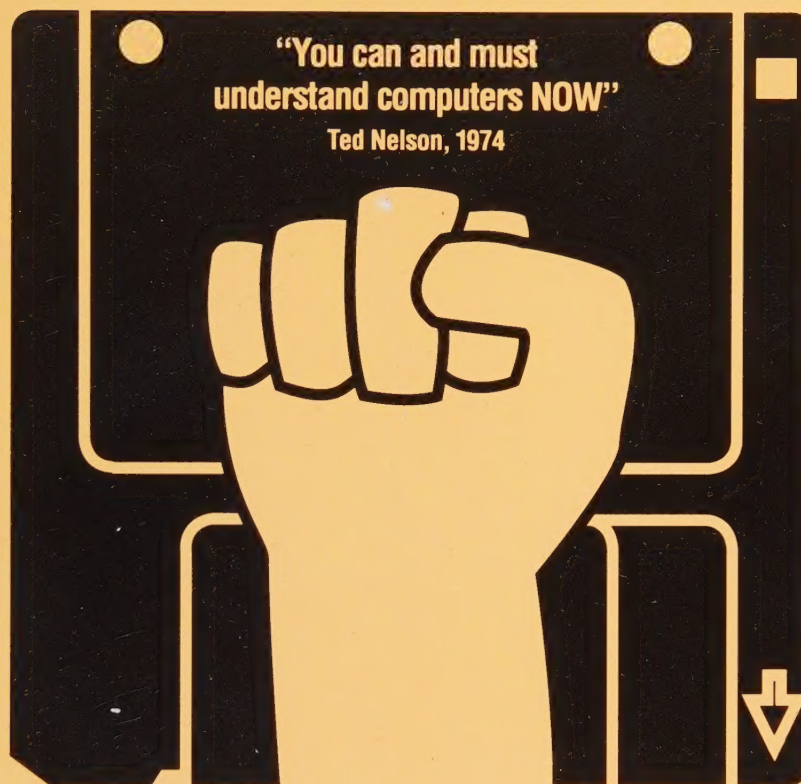


*Ted Nelson*


REVISED  
& UPDATED!

# COMPUTER LIB



## *DREAM MACHINES*

*Introduction by Stewart Brand,  
originator of The Whole Earth Catalog  
and author of The Media Lab*



Digitized by the Internet Archive  
in 2023 with funding from  
Kahle/Austin Foundation

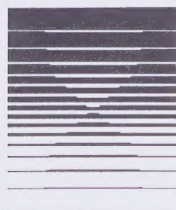
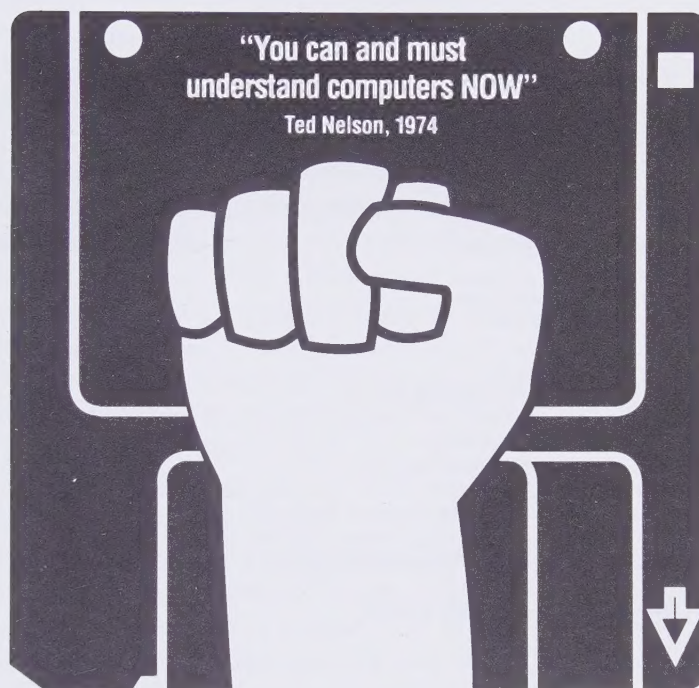
<https://archive.org/details/computerlibdream0000nels>



*Ted Nelson*

FRED C MOULTON  
FCMOULTON@GMAIL

# COMPUTER LIB



T E M P U S™

PUBLISHED BY

Tempus Books of Microsoft Press  
A Division of Microsoft Corporation  
16011 NE 36th Way, Box 97017  
Redmond, Washington 98073-9717

Copyright ©1974, 1975, 1987 by Theodor H. Nelson  
Published 1974. Revised Edition 1987.

All rights reserved. No part of the contents of this book may  
be reproduced or transmitted in any form or by any means without  
the written permission of the publisher.

Library of Congress Cataloging in Publication Data  
Nelson, Theodor H.

Computer lib/dream machines / Theodor Nelson. — Rev. ed.  
p. cm.

Rev. ed. of: Computer lib. 1974.

ISBN 0-914845-49-7

I. Computers. I. Nelson, Theodor H. Computer lib. II. Title.  
QA76.N39 1987 87-19706 CIP  
004—dc19

Printed and bound in the United States of America.

1 2 3 4 5 6 7 8 9 HCHC 8 9 0 9 8 7

Distributed to the book trade in the  
United States by Harper & Row.

Distributed to the book trade in  
Canada by General Publishing Company, Ltd.

Distributed to the book trade outside the  
United States and Canada by Penguin Books Ltd.

Penguin Books Ltd., Harmondsworth, Middlesex, England  
Penguin Books Australia Ltd., Ringwood, Victoria, Australia  
Penguin Books N. Z. Ltd., 182-190 Wairau Road, Auckland 10, New Zealand

British Cataloging in Publication Data available

Note that the author's original copyright also applies to certain interactive designs as presented here,  
under both statutory copyright law and such judicial interpretations as the current look-and-feel  
doctrine.

THIS BOOK IS ALSO AVAILABLE ON-LINE ON THE EXPERIMENTAL XANADU™  
HYPERTEXT NETWORK. Contact Project Xanadu, 8480 Fredericksburg, Suite 138,  
San Antonio, TX 78229, for details.

# F O R E W O R D

---

“Revolution” is an overused word these days, especially around new technology, most especially around computer technology. For the coming of personal computers, however, it’s the right word, because the people who made the event happen saw themselves as revolutionaries and acted accordingly. In motivation, style, and global effect, they fought and won a second American Revolution. Oh, was it American.

It was all quite ironic. At the very time in the late ’60s through the mid-’70s that the New Left was noisily advocating populist political revolution and conspicuously failing, a tiny sub-subculture of the Counter-Culture was quietly, invisibly fomenting a populist computer revolution, and the shock of *their* success was heard ’round the world.

The one fomenter who was not quiet was Ted Nelson, accurately depicted as the Tom Paine of the personal-computer revolution. His 1974 tract, *Computer Lib/Dream Machines*, had the same effect as Paine’s *Common Sense*—it captivated readers, informed them, and set them debating and eventually marching, rallying around a common cause many of them hadn’t realized was so worthy or even a cause before. These troops, however, marched with their fingers (on keyboards) rather than bloody feet, and they were even less responsive to command than the rabble-in-arms that embarrassed the British Empire. The Hackers’ Rebellion was a guerrilla war, fought mostly at night while the enemy slept.

The enemy was Central Processing, in all its commercial, philosophical, political, and socio-economic manifestations. Big Nurse.

*Computer Lib* has been called “the first personal-computer book” —I think incorrectly. I’d give that honor to Nicholas Negroponte’s 1970 book from MIT Press, *The Architecture Machine*. What is accurate is the designation of *Computer Lib* as “the first computer cult book,” with interesting linkage to another self-published cult book of the time, the *Whole Earth Catalog*, which I was responsible for. Nelson says he blatantly imitated the *Catalog*’s large, loose format crammed with eclectic information, as well as its basic function of “access to tools.” Well, a lot of people did that—more than a hundred publications that I know of. No one did it to better effect than Nelson.

That’s part of why it’s a particular delight for me to introduce the new edition of *Computer Lib*. But I’m primarily interested in helping here for exactly the same reason we would welcome a 1987 rewrite by Tom Paine called *Common Sense—The Update*, laying into the excesses and blindnesses of the 20th century American Empire and tracing new avenues of liberation.

Ted Nelson in 1987 declares: “COMPUTERS ARE JUST AS OPPRESSIVE AS BEFORE, BUT SMALLER AND CHEAPER AND MORE WIDESPREAD. NOW YOU CAN BE OPPRESSED BY COMPUTERS IN YOUR LIVING ROOM.”



---

The Revolution may seem to be over, but it's not won. It's like building a house. The sky-filling framing is heroic and dramatic and quick. Most of the heroes take their bows and leave. Then the finishing—making the house livable—drags on for months and years, over-budget, lost in detail, discouraging. *Real* heroes linger, reminding the finishers of the grand original scheme, helping every refinement serve that vision, driving the experiments closer and closer to the meat of human living.

Finishing is never finished.

That's in a technology that holds relatively still, like house carpentry. Computer technology will probably not hold still during the lifetime of any reader of this book. Every new tool needs to be explored not only as an instrument of finer finishing, but as a potential reviser of the fundamental scheme of possibilities. Computer technology is an ultra-fluid medium in which constant reframing, as well as refinishing, is possible—therefore imperative.

Problem: You don't get enough of the requisite originality just from young specialists—the myriad Computer Science majors coming out of the colleges—and specialists tend to close off a discipline to outsiders. Sometimes specialist self-defense discourages deep innovation. The broad band of originality suitable to this medium has to come from young generalists, and reaching *them* takes a technology far more convivial, self-evident, affordable, powerful, and user-malleable than what we have so far.

A revolution of this constant a potency can afford to leave no prospective revolutionary unreached and uninvited. Free the World Six Billion.

STEWART BRAND



## HOWto. Read this book.

Much of the specialness of this book is not what it says, but when it said it (1974 and 1975). We've added to it and spruced it up, but the original materials from the 1974 book and the 1975 supplement have been kept distinct from newer stuff.

The publisher has taken great care to indicate which material is old and which is new, in the most discrete and elegant fashion. (A lot of computer people don't know the difference between discrete, meaning separate, and discreet, meaning tasteful, but in this case we've tried to make it both anyhow.) Paragraphs in plain type are old; paragraphs in italics are new. (Headings, however, proved uncontrollable in this regard.)

What we've marked as "old" includes both material from the 1974 original and the 1975 supplement—in other words, from just before and just after the Altair.

I write to be read aloud, which confuses editors; the punctuation and capitalization are a guide to the intonation of the reader.

### Books by **TED NELSON**

Life, Love, College, Etc. 1959 (originally entitled *We Need a Sociology Department*).

Computer Lib, 1974; supplement 1975; second edition, 1987.

The Home Computer Revolution, 1977.

Literary Machines, 1981; major revision 1987 (Edition 87.1).

Biostrategy and Polymind: A New Theory of Human Life. In preparation.

### AUTHOR'S CREDENTIALS

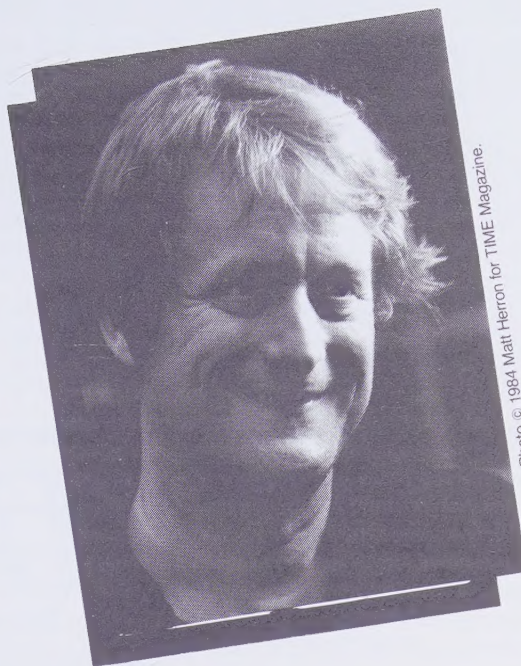


Photo © 1984 Matt Herron for TIME Magazine.

Designer; author of unusual and influential books. Occasional academic, lecturer, etc., in U.S. and abroad.

Discovered and named hypertext, 1965. Inventor of hypergrid interactive software, 1980s.

Founding designer and director of Project Xanadu. B.A., philosophy, Swarthmore; graduate study U. of Chicago.

M.A., sociology, Harvard. Mostly self-taught in computers.

Research assistant, Communication Research Institute, 1962-63.

Instructor in sociology, Vassar College, 1964-66.

Senior staff researcher, Harcourt, Brace & World Publishers, 1966-67.

Consultant to Bell and CBS Laboratories, 1967-70. Proprietor of The Nelson Organization, Inc., New York City, 1969-72.

Lecturer, U. Illinois at Chicago Circle, 1973-76.

Co-founder of the Itty Bitty Machine Co. computer store, Evanston, Illinois, 1976.

Editor, Creative Computing magazine, 1980-81.

Software designer and senior technical writer, Datapoint, 1981-84.

Selected as one of America's most eligible bachelors by the editors of Playgirl (Celeste Fremon, The American Bachelors' Register. Simon & Schuster, 1982).

There are sections on Nelson and the Xanadu system in two recent books: Howard Rheingold, *Tools for Thought* (Simon & Schuster, 1985) and K. Eric Drexler, *Engines of Creation* (Anchor/Doubleday, 1986).

### A NOTE ON TERMINOLOGY.

The author is in some ways an Old Fashioned Guy, and a few of the words used here have an antique ring. In order to maintain the text of the original book, these terms have been retained. New remarks have been kept consistent with the older parts.

Be prepared to see the word minicomputer for anything smaller than a mainframe. This book came out before the term "microcomputer" became popular, and so the author uses the words minicomputer and dinky computer to mean small machines you can fit in your apartment or on your desk. (The author insists that the term "microcomputer" means nothing and never has, and refuses to use it.)

Similar strictures affect the term core memory, the term in use back in the seventies. Again, while it has been commonly replaced by the term "RAM" (for Random Access Memory), I consider that also to be stupid. The memory is still in the middle, and core means middle, right? Never use an acronym when a lousy Anglo-Saxon word can be used instead, say Strunk and White and the Author, and a number of computer old-timers agree with them.

Also, in 1974, computer screens were called displays. And indeed this book was intended to fight the good fight in favor of terminals, and against the computer centers where they wanted you to put things in on punch cards. So "terminal" is occasionally used in this book where desktop computer would be more appropriate now.

The term "word processing" is sometimes in quotes because the author has called it "text handling" since 1960, and he considers the term "word processing" to have been a political maneuver by IBM around 1967. You don't process the words (like numbers); you just store them and move them around.

Gender reference, as well, is slightly oldfashioned. At the time this book was written, it was okay to say "mankind" and use the masculine when you intended to refer to persons of all sexes. Since then the language has had its Consciousness Raised, but we're keeping old parts of this book intact, so some of it sounds sexist now.



# INTRODUCTION TO THE 1987 EDITION

Hi again. To newcomers, welcome; to old friends of this book, welcome back.

Here is the new edition of a quirky book that has been fairly influential and made some rather good predictions, especially about the personal computer era that had not yet begun (but began immediately after the book appeared). (The rather good predictions are discussed on p. 9.)

This book is frequently referred to as a classic. It has a certain historic significance, for when it appeared (it is apparently the first "personal computer book") and for the influence it had in the seventies. It first came out in 1974, only a few months before the Altair announcement in December of that year and the subsequent personal computer explosion. It be-

came a cult book among the hackers of the seventies, typically those who were in their teens and twenties, who savored its emphasis and vision. Many of them are now leaders in the computer industry of the eighties.

This was, and is, intended as a layman's introduction to the computer field. In practice it has also been a book of

browsings for the would-be, will-be computer person and even professional. Many professional Deep Thinkers, though, resent some of the things said here because they are not being presented through the "proper forum," i.e., professional conferences and refereed publications. Still, these thoughts reached a lot of people, right?

I wish we could have run the WHOLE original 1974 Computer Lib. However, the original weighed in at 300,000 words. It contained ancient concerns like where to rent Teletypes, and there's been a lot of new stuff to cover, even with the good old Computer Lib slapdashery.

There was little time to go through the copious notes I've made for the update to these thirteen years, but then there was too much material anyway. (Some day, too, it might be fun to publish the extensive pieces that got left out of the original, but they're in storage someplace.)

For those few but precious readers who care, here are some

## NOTES ON THE CHANGES

### 1. THE BELOVED DOUBLE COVER:

After much discussion, and despite the sage misgivings of some of Microsoft Press' best marketing people, these two independent books, joined at the navel, have kept their identity within two front covers. I'm sure this is the right thing to do.

### 2. REARRANGEMENT:

Things have been moved all around, so lovers of the old book may be disoriented at first.

### 3. THE NEW:

Various updated, heartfelt, miscellaneous and random new chunks are scattered throughout the book. They, like this one, are italicized.

### 4. THE CUTS:

#### A. Cuts In The 1974 Edition

Half the manuscript, and the glossary, had to be kicked aside; including sections on movies, "multi-media," microfilm, training simulators, augmented stage productions of the future, and goodness knows what.

#### B. The New Cuts

A few big chunks have been thrown out. This includes introductions to three computer languages, stuff on microcode and programming in binary, some historical material, too much about certain individuals, lists of 1974 products, and extraneous ibmology. And I've dropped various personal reminiscences, with the intent of republishing them elsewhere, someday.

I have also made a few unmarked cuts within sentences, irrelevancies, words like "recently," or stuff not worth explaining.

However, a few obsolete prices and addresses have been left in because they're amusing or enlightening.

I've also had to make various cuts because Microsoft Press' standards of Permission are strict; the permissions I got in '73 worked fine for me, but big companies, by necessity, live by different standards.

## UNIFIED COMPUTER FIELD THEORY:

### There Is Only One Computer Field

*Don't let them fool you, don't believe in those lines they're trying to draw! There is no separate "personal" computer world.*

*There is no distinction between personal computers and the old kind in what they will do, how they are designed, or how they are built. The difference is in the MARKETPLACE—how they are sold.*

*To talk about the personal field as separate only sows confusion, like the idiotic word "microcomputer" (see p. 45). Anyone who tells you that the personal field is truly separate from the rest is either ignorant, or trying to sell you something you should consider not buying.*

*By pretending it's a different field, and that all you need to know is what he's selling, the salesman is saying implicitly that you don't have to really understand things. Maybe some people find this reassuring. But if the computer field has scared you, isn't it better to know you shouldn't have been scared than to have those apprehensions catered to?*

*Marketing boundary or no, there are no boundaries in the technicalities of the real computer world.*

*The principles are the same for big and little machines. Just as all roads are connected, all the ideas of the computer field are interconnected.*

*Computerdom is an immense intertwined tapestry: its ideas are one mass, its breakthroughs have possible consequences in every branch, and for everybody. An idea may go from one area to a completely different one—that is, an area someone previously thought was completely different. Every technique of computerdom is, or may someday be, at your disposal.*

*It all trades off: big machines may be substituted for little ones, and (preferably) little for big. Different computer languages may, with thought and difficulty, be substituted one for the other. All the different data-structure techniques can more or less be interchanged: arrays may be substituted for linked lists, or trees for either, or hashing, or whatever; in various mixed switcheroos. But then, algorithms may in some ways be substituted for data structures.*

*It's all one tangle, one world of excitement, disappointment, confusion, and excitement again.*



There's quite a lot in here that people missed the first time, presumably because there just was so much of it, and on first impression the book's main thrust was so distinctive as to blur some of the subtleties. Now there is more and more debate about many of the issues: freedom, access and privacy; and interactive systems and their design, about which I said a great deal that was overlooked at that time because people were not thinking very much about these things. So old friends may find things they didn't notice before.

And, indeed, there are a number of parts people haven't gotten yet, especially on computer graphics and the media of the future (the Dream Machines side).

Computer Lib has been updated only slightly, because its main messages are just as important now as when it came out. (See related comments, "Notes on the Changes," nearby.)

A lot of people are under the impression that the computer field is constantly changing. In fact some things don't change at all, or change only glacially: data structures, languages, computer design, glorious predictions for artificial intelligence.

But the big issues I pointed out in the original book were the issues of personal freedom and access to information and its tools. I said the world is in an extremely precarious position, and all that can save us is to give tomorrow's idea hackers the

This side of the book, *Computer Lib* proper (whose title is nevertheless the simplest way to refer to both halves), is an attempt to explain simply and concisely why computers are marvelous and wonderful, and what some main things are in the field.

The second half of the book, *Dream Machines*, is specifically about fantasy and imagination, and new techniques for it. That half is related to this half, but can be read first; I wanted to separate them as distinctly as possible.

*The two sides are really two separate books with two thrusts:*

*On this side, PERSONAL FREEDOM THROUGH UNDERSTANDING COMPUTERS;*

*On the Dream Machines side, NEW UNDERSTANDING AND NEW ARTS THROUGH GRAPHICS AND INTERACTION.*

## COMPUTING HAS ALWAYS BEEN PERSONAL

By this I mean that if you weren't intensely involved in it, sometimes with every fiber in your mind atwiltch, you weren't doing computers, you were just a user.

If you get involved, it involves all of you: your heart and mind and way of doing things and your image of yourself. A whole way of life.

## ASPECTS of this book ...

The explanations — not yet fully debugged — are intended for anybody. The listings of expensive products and services are intended not only as corroborative detail, for a general sense of what's available, but also for business people who might find them helpful, for affluent individuals and clubs who want to try their hand, and finally as a box score of how the prices are coming down. Because we are all going to be able to afford these things pretty soon.

Many people could have written *Computer Lib*. That is, there are many people who both know computers and write interestingly. The late Lee Revens, who was an editor for the ACM; Jonathan Post, poet and physicist; Jim Warren, founder of the West Coast Computer Faire and what is now InfoWorld; Dave Ahl, publisher emeritus of the late lamented *Creative Computing*. And probably many more. Except it wouldn't have been as vehement. "This book was written by somebody who cares too much," somebody said.

THERE IS ONLY ONE COMPUTER WORLD; but some people lock themselves in compartments.

## ... SUMMARY of this book

Man has created the myth of "the computer" in his own image, or one of them: cold, immaculate, sterile, "scientific," oppressive.

Some people flee this image. Others, drawn toward it, have joined the cold-sterile-oppressive cult, and propagate it like a faith. Many are still about this mischief, making people do things rigidly and saying it is the computer's fault.

Still others see computers for what they really are: versatile gizmos which may be turned to any purpose, in any style. And so a wealth of new styles and human purposes are being proposed and tried, each proponent propounding his own dream in his own very personal way.

This book presents a panoply of things and dreams. Perhaps some will appeal to the reader...



very best facilities they can have. And new visualizations, and well-designed interactive systems, will be vital to the world of the future.

Now here we are in 1987, and the point is, **NOTHING HAS CHANGED.** Everything has proceeded as foreordained, except that certain details have worked themselves out.

### THE REVOLUTION SO FAR

Many people look in amazement at the changes wrought by the personal computer, and think it's a revolution.

You ain't seen nothin' yet.

There is a new world called the world of "personal computers"—except that computers have *always* been personal.

And there is really no personal computer world, distinct from the rest. **There is only one computer world;** all parts of it interconnect. (You may only see one corner of it and not realize.) The same data structures are used in the space shuttle and accounting programs. The same people go from programming text systems to programming video games.

It could be argued in 1974, when this book first came out, that the computer world was not the *real* world. But now the computer world is no longer in any sense distinct from "the real world;" the computer world is the real world.

Actually things are still quite terrible in the computer field: balkanized, incompatible, often stressing exactly the wrong things (as in video games and "computer literacy" courses).

But there is a spirit abroad of exuberant and intelligent change—not judicious and solemn, but wild, experimental, and hopeful in every direction.

The future may give us a new freedom of information in heaven and earth, with new abilities to predict, coordinate and agree on a future beneficial to all.

Or it may bring the dank horrors so well foreseen by Orwell.

But the innovation, and the cause for hope, are coming from you out there. The people who have bought computers, figured them out, and started building the new possibilities. May this book continue to inspire in all these directions.

The future is too important to be left to the pompous and stuffy people.

COMPUTERS BELONG TO ALL  
MANKIND.

## INTRODUCTION TO THE 1974 EDITION

Any nitwit can understand computers, and many do. Unfortunately, due to ridiculous historical circumstances, computers have been made a mystery to most of the world. And this situation does not seem to be improving. You hear more and more about computers, but to most people it's just one big blur. The people who know about computers often seem unwilling to explain things or answer your questions. Stereotyped notions develop about computers operating in fixed ways—and so confusion increases. The chasm between laymen and computer people widens fast and dangerously.

This book is a measure of desperation, so serious and abysmal is the public sense of confusion and ignorance. Anything with buttons or lights can be palmed off on the layman as a computer. There are so many different things, and their differences are so important; yet to the lay public they are lumped together as "computer stuff," indistinct and beyond understanding or criticism. It's as if people couldn't tell apart camera from exposure meter or tripod, or car from truck or tollbooth. This book is therefore devoted to the premise that

**EVERYBODY SHOULD UNDERSTAND  
COMPUTERS.**

It is intended to fill a crying need. Lots of everyday people have asked me where they can learn about computers, and I have had to say *nowhere*. Most of what is written about computers for the layman is either unreadable or silly. (Some exceptions are listed nearby; you can go to them

instead of this if you want.) But virtually nowhere is the big picture simply enough explained. Nowhere can one get a simple, soup-to-nuts overview of what computers are really about, without technical or mathematical mumbo-jumbo, complicated examples, or talking down. This book is an attempt.

(And nowhere have I seen a simple book explaining to the layman the fabulous wonderland of computer graphics which awaits us all, a matter which means a great deal to me personally, as well as a lot to all of us in general. That's discussed on the flip side.)

**Computers are simply a necessary and enjoyable part of life, like food and books. Computers are not everything, they are just an aspect of everything, and not to know this is computer illiteracy, a silly and dangerous ignorance.**

Computers are as easy to understand as cameras. I have tried to make this book like a photography magazine—breezy, forceful and as vivid as possible. This book will explain how to tell apples from oranges and which way is up. If you want to make cider, or help get things right side up, you will have to go on from here.

I am not a skillful programmer, hands-on person or eminent professional; I am just a computer fan, computer fanatic if you will. But if Dr. David Reuben can write about sex I can certainly write about computers. I have written this like a letter to a nephew, chatty and personal. This is perhaps less boring for the reader, and certainly less boring for the writer, who is doing this in a hurry. Like a photo

magazine, it throws at you some rudiments in a merry setting. Other things are thrown in so you'll get the sound of them, even if the details are elusive. (We learn most everyday things by beginning with vague impressions, but somehow encouraging these is not usually felt to be respectable.) What I have chosen for inclusion here has been arbitrary, based on what might amuse and give quick insight. Any bright highschool kid, or anyone else who can stumble through the details of a photography magazine, should be able to understand this book, or get the main ideas. This will not make you a programmer or a computer person, though it may help you talk that talk, and perhaps make you feel more comfortable (or at least able to cope) when new machines encroach on your life. If you can get a chance to learn programming—see the suggestions on p. 40—it's awfully good experience for anybody above fourth grade. But the main idea of this book is to help you tell apples from oranges, and which way is up. I hope you go on from here, and have made a few suggestions.

I am "publishing" this book myself, in this first draft form, to test its viability, to see how mad the computer people get, and to see if there is as much hunger to understand computers, among all you Folks Out There, as I think. I will be interested to receive corrections and suggestions for subsequent editions, if any. (The computer field is its own exploding universe, so I'll worry about up-to-dateness at that time.)

### THE COMPUTER PRIESTHOOD

Knowledge is power and so it tends to be hoarded. Experts in any field rarely want people to understand what they do, and generally enjoy putting people down.

Thus if we say that the use of computers is dominated by a priesthood, people who spatter you with unintelligible answers and seem unwilling to give you straight ones, it is not that they are different in this respect from any other profession. Doctors, lawyers and construction engineers are the same way.

But computers are very special, and we have to deal with them everywhere, and this effectively gives the computer priesthood a stranglehold on the operation of all large organizations, of government bureaux, and anything else that

**People talk about the "depersonalization" of computers.**

**I want to emphasize the personalization of computers—that they, their programs and languages, are designed by individuals, each with his or her own obsessions.**

**So more than any book for beginners, this one stresses the personal contributions of individuals, and the wide-ranging disagreements of a field which to many in the outside world seemed "objective" and "scientific."**



they run. Members of Congress are now complaining about control of information by the computer people, that they cannot get the information even though it's on computers. Next to this it seems a small matter that in ordinary companies "un-trained" personnel can't get straight questions answered by computer people; but it's the same phenomenon.

It is imperative for many reasons that the appalling gap between public and computer insider be closed. As the saying goes, war is too important to be left to the generals. Guardianship of the computer can no longer be left to a priesthood. I see this as just one example of the creeping evil of Professionalism,\* the control of aspects of society by cliques of insiders. There may be some chance, though, that Professionalism can be turned around. Doctors, for example, are being told that they no longer own people's bodies. And this book may suggest to some computer professionals that their position should not be as sacrosanct as they have thought, either.

This is not to say that computer people are trying to louse everybody up on purpose. Like anyone trying to do a complex job as he sees fit, they don't want to be bothered with idle questions and complaints. Indeed, probably any group of insiders would have hoarded computers just as much. If the computer had evolved from the telegraph (which it just might have), perhaps the librarians would have hoarded it conceptually as much as the math and engineering people have. But things have gone too far. People have legitimate complaints about the way computers are used, and legitimate ideas for ways they should be used, which should no longer be shunted aside.

In no way do I mean to condemn computer people in general. (Only the ones who don't want you to know what's going on.) The field is full of fine, imaginative people. Indeed, the number of creative and brilliant people known within the field for their clever and creative contributions is considerable. They deserve to be known as widely as, say, good photographers or writers.

## FANDOM

With this book I am no longer calling myself a computer professional. I'm a computer *fan*, and I'm out to make you one. (All computer professionals were

fans once, but people get crabrier as they get older, and more professional.) A generation of computer fans and hobbyists is well on its way, but for the most part these are people who have had some sort of an In. This is meant to be an In for those who didn't get one earlier.

The computer fan is someone who appreciates the options, fun, excitement, and fiendish fascination of computers. Not only is the computer fun in itself, like electric trains; but it also extends to you a wide variety of possible personal uses. (In case you don't know it, the price of computers and using them is going down as fast as every other price is going up. So in the next few decades we may be reduced to eating soybeans and carrots, but we'll certainly have computers.)

Somehow the idea is abroad that computer activities are *uncreative*, as compared, say, with rotating clay against your fingers until it becomes a pot. This is categorically false. Computers involve imagination and creation at the highest level. Computers are an involvement you can really get into, regardless of your trip or your karma. They are toys, they are tools, they are glorious abstractions. So if you like mental creation, toy trains, or abstractions, computers are for you. If you are interested in democracy and its future, you'd better understand computers. And if you are concerned about power and the way it is being used, and aren't we all right now, the same thing goes.

## THE SOCIETY

Which brings us to our next topic.

There is no question of *whether* the computer will remake society; it has. You deal with computers perhaps many times a day—or worse, computers deal with you, though you may not know it. Computers are going into everything, are intertwined with everything, and it's going to get more and more so. The reader should have a sense of the dance of options, the remarkably different ways that computers may be used; by extension, he should come to see the extraordinary range of options which confront us as a society in our future use of them. Indeed, computers have with a swoop expanded the options of everything.

But a variety of inconvenient systems already touch on our lives, nuisances we must deal with all the time; and I fear that worse is to come. I would like to alert the reader, in no uncertain terms, that the

time has come to be openly attentive and critical in observing and dealing with computer systems; and to transform criticism into action. If systems are bad, annoying and demeaning, these matters should be brought to the attention of the perpetrators. Politely at first. But just as the atmospheric pollution fostered by GM has become a matter for citizen concern and attack through legitimate channels of protest, so too should the procedural pollution of inconsiderate computer systems become a matter for the same kinds of concern. The reader should realize he can criticize and demand;

THE PUBLIC DOES NOT HAVE TO  
TAKE WHAT'S BEING DISHED OUT.

There is already a backlash against computers, and the spirit of this anti-computer backlash is correct, but should be directed against very specific kinds of things. The public should stop being mad at "computers" in the abstract, and start being mad at the people who make inconvenient systems. It is not "the computer," which has no intrinsic style or character, which is at fault; it is people who use "the computer" as an excuse to inconvenience you, who are at fault. The mechanisms of legitimate public protest—sit-ins and so on—should perhaps soon be turned to complaint over bad and inhuman computer systems. 🐼

**\* This is a side point. I see Professionalism as a spreading disease of the present-day world, a sort of poly-oligarchy by which various groups (subway conductors, social workers, bricklayers) can bring things to a halt if their particular new increased demands are not met. (Meanwhile, the irrelevance of each profession increases, in proportion to its increasing rigidity.) Such lucky groups demand more in each go-round—but meantime, the number who are permanently unemployed grows and grows.**

This book is a multi-user, high-resolution, demand-paged, hardware-software read-mostly memory, retrieval and display system with real-time interaction, tactile interface, audio and video feedback.

See?

Don't be afraid of technical terms.

**I want very much to encourage skepticism about simple solutions proposed under technological slogans. Some may be excellent (like water fluoridation), some may be preposterous (see "the Ray-Gun Initiative," p. 171). But an informed public must understand something of the details in order to assess their plausibility. Today's public seems to trust technologists far too much.**



☛ The question is, will the crummier trends continue? Or can the public learn, in time, what good and beautiful things are possible, and translate this realization into an effective demand? I do not believe this is an obscure or specialized issue. Its shadow falls across the future of mankind, if any, like a giant sequoia. Either computer systems are going to go on inconveniencing our lives, or they are going to be turned around to make life better. This is one of the directions that consumerism should turn.

I have an axe to grind: I want to see computers useful to individuals, and the sooner the better, without necessary complication or human servility being required. Anyone who agrees with these principles is on my side, and anyone who does not, is not.

#### THIS BOOK IS FOR PERSONAL FREEDOM, AND AGAINST RESTRICTION AND COERCION.

That's really all it's about. Many people, for many reasons of their own, enjoy and believe in restricting and coercing people; the reader may decide whether he is for or against this principle. A chant you can take to the streets:

COMPUTER POWER TO THE PEOPLE!  
DOWN WITH CYBERCRUD!

#### THE FUTURE, IF ANY

Simply as a matter of citizenship, it is essential to understand the impact and uses of computers in the world of the future, if any; and to have a sense of the issues *about* computers that confront us as a people—especially privacy and data banks, but also strange new additions to our economic system (“the checkless society”), our political system (half-baked vote-at-home proposals), and so on. I regret that there is not room to cover these here.

Various companies are seeking wide public support for the sorts of things they are trying to bring about. Legislation will be proposed on which the views of the public should have a bearing. It is important that these be understood sensibly by some part of the electorate before they are made too permanent, rather than made matters of dumb assent.

Finally, and most solemnly, computers are helping us understand the unprecedented danger of our future (see “The Club of Rome,” p. 172). The human race may have only a short time left on earth, even if there is no war. These studies must

be seen and understood by as many intelligent men of good will as possible.

#### THEREFORE

Welcome to the computer world, the damndest and craziest thing that has ever happened. But we, the computer people, are not crazy. It is you others who are crazy to let us have all this fun and power to ourselves.

COMPUTERS BELONG TO  
ALL MANKIND.

**C**omputers are catching hell from growing multitudes who see them uniformly as the tools of the regulation and suffocation of all things warm, moist, and human. The charges, of course, are not totally unfounded, but in their most sweeping form they are ineffective and therefore actually an acquiescence to the dehumanization which they decry. We clearly need a much more discerning evaluation in order to clarify the ethics of various roles of machines in human affairs.”

Ken Knowlton in “Collaborations with Artists—a Programmer’s Reflections” in *Nake & Rosenfeld*, eds., *Graphic Languages* (North-Holland Pub. Co.), p. 399.

*I talk a lot about freedom. Many other people in the computer world are concerned with this, and it is their work with codes and so forth that will give us what freedom and privacy we may retain in the future. However, it is my job to keep this issue in front of you. Freedom has to be mentioned all the time. Don't you forget it.*

# HISTORY OF THIS BOOK

The present product is not the book I had meant to write. Most is first-draft; how the sentences do run on. (Believe it or not, I do not like underlining things—a first-draft expedient.) Fact-checking and bibliographies had to be largely abandoned. Better planning could have increased type size; and so on. Sorry for all that.

I wanted to write an introduction to computers, and a separate book on Fantics, for years. But the idea of binding them back-to-back in a Whole Earth format, with lots of mischievous Enrichment material, didn't hit me till Jan. '73. I have tried to add all the stimulating and exhilarating stuff I could find, especially personalizations; computers are deeply personal machines, contrary to legend, and so are showing-systems. I regret having to throw so many of my concerns into comic relief, but I hope that some readers will sense the seriousness below.

The final inspiration for this book came from something called the *Domebook*, that tells you straightforwardly how to make Geodesic Domes. And of course I'm blatantly imitating, in a way, the wonderful *Whole Earth Catalog* of Stewart Brand. As I think back, though, the tone also comes in part from Pete Seeger's wonderful banjo book, and Tom McCahill's automobile reviews in *Mechanix Illustrated*. As to the last aspect, that of taking my case to the public because the experts won't listen, the only precedent I can think of is Maj. Alexander de Seversky's *Victory Through Air Power*, telling the country how he thought we should win World War II.

*I had been hired by Joe Lipson, vice chancellor of the University of Illinois, and told to “do something creative about education.” I looked at Project Plato, which was big on campus there, concluded that it was a hopeless mess, and decided to write this.*

*I first thought of calling this The Counterculture Computer Book. That would've worked, but it was kind of obtuse.*

*Then one night in February, out for dinner with my friend Sheila, the title Computer Lib came to me like a thunderbolt. The right word, the right title, can make an enormous difference; at once the tone of the book was clear.*

*While I wrote it I was teaching and serving as a staff photographer at the University of Illinois in Chicago.*

## CHRONOLOGY

*I started writing Computer Lib in January 1973; I finished it about July of 1974, and had copies to distribute around the end of August. (I added a large supplement in the summer of '75, after the Altair had come out; so the materials from the original Computer Lib straddle that important date.)*

## THE TIMES WHEN IT WAS WRITTEN

*It is hard to remember the atmosphere of those days.*

*Computer books up to that time had tables of contents with lots of numerals and subheadings, and were full of block diagrams. But I make no claims to stylistic originality: computer magazines were already light and clever, sometimes. (Datamation had published some wonderfully breezy and terribly funny stuff as early as the sixties, and Bob Albrecht's PCC, a playful computer tabloid from the People's Computer Company in the Bay Area, was already being published in '73. Another playful magazine, Dave Ahl's Creative Computing, began publishing just about the time Computer Lib appeared.)*

*At that time, most people thought computers had to be mainframes; Unix was far less well known.*

*Everybody knew that chip prices were going down. What exactly this was going to mean was not clear to people; they thought that processor chips would be used in soft-drink machines and typewriters, but no one had any sense of a Computer Hunger out there.*

*Computer recreations were just starting. There were a couple of video games, "Pong" and a few more like it, but the heyday of videogames lay several years ahead.* ■

# TARGETING

**One approach to the world is to do highly-targeted projects, meant to appeal to a very specific audience or market. This is valid, but I find it personally uninteresting.**

**To me the real design challenge—whether of a book or of software—is a broad-spectrum product that meets many desiderata and serves many kinds of reader or user. That was the approach in this book and that is the approach in my major software designs.**

## MAIN POINTS OF THIS BOOK

**The book is of interest not just for what it says but for when it said it—1974. You hear a lot of this stuff on every streetcorner now.**

**It was a conscious imitation of the Whole Earth Catalog, attempting to bring the same informality to a layman's introduction to the computer field and electronic media.**

### 1. THE LIB SIDE

**It urged people toward a new era of personal computing, which I saw as an important new fad, excitement and ideology; and it made various predictions about the world. (I tried to capture the ideas and attitudes of computer people as I knew them, and to talk about the field as I saw it coming—once computers got into the hands of The People.)**

**It was a crusade for small computers, which were at that time called "minicomputers." (The inane term "microcomputer," which means nothing different, has since then muddled the issue.)**

**It was to tell people of the evil of IBM, especially in the way they had suppressed the development of interactive systems and computer freedom—creating computer centers, concentration camps for information, where people were regimented and oppressed; hence the large and carefully scholarly section about that firm.**

**It was an invitation to farble and to hack, making computerdom's animated concepts and devious realities your own.**

**It was an invitation and exhortation to a new era of freedom and understanding, which has not happened, but is not yet impossible—though the clumsiness and muddiness of the personal computer field as we see it today are discouraging indeed.**

**A new point in this edition (symmetrical with the Dream Machines side) is that computers are a Balkanized, incompatible mess, and that has to be fixed.**

**It was a declaration that only computers in the hands of the people, and their new visualizing tools as well, can save the world—or anything decent—from the horrors that are to come.**

### 2. THE DREAM MACHINES SIDE

**I tried to gather, here, many people's grand and revolutionary ideas for graphics, retrieval, future paradises. (I ask credit for only a few of the ideas here: hypertext, hypermedia, the Xanadu system, and a few interactive designs.)**

**It was an announcement of services and artificial realities yet to come, some of which are now resplendent, some of which have arrived in twisted form.**

**New showing-systems—automatic and branching presentational systems of all types—are being continually announced by different corporations, sometimes with claims of miraculous powers.**

**Details of these systems vary deeply, with all kinds of audio-visual-textual-cinematic branching; but the rules of showmanship and conceptual presentation do not change. But they are not well understood, especially by people who want to sell systems.**

**Not understanding these eternal verities leads people in some wrongful directions, as in some computer-assisted instruction.**

**But these systems will naturally gravitate into hypermedia.**

**The details of the hypermedia systems of the future are vital; for these will be the principal intellectual environment of the human race hereafter.**

**The unification of all these systems is necessary: artistic, presentational, conceptual.**

**This is not understood because the technoids are still in the saddle. But when the artists and filmmakers finally get control, the glorious potentialities of hypermedia will finally be revealed.**

**In the long future, the generalized unification of our digital hypermedia archives and presentational systems is vital. We need a unity like that of paper. And freedom of access is our ultimate concern throughout the future.**

**Dream Machines was an advertisement for my own services as a software designer, with free samples. And it was an invitation for smart and idealistic souls to join me in a great and brave undertaking, project Xanadu.**



Interactive systems were new; the concept of **highly**-interactive computer systems, unlike line-feed terminal programs, was largely unknown. There were pocket calculators, but today's toy-store variety of talking bears and programmable games were just not around. The idea that interactive systems had to be **designed**, rather than "built by engineers," was practically unheard.

## THE ORDEAL

This project, simple in principle, has been infinitely bothersome. Self-publication was necessary because no publisher could have comprehended the concept of this book.

Running through in my mind what had to be said, I thought originally that I could bat the project out in about two weeks, and that it would be forty pages of 8 1/2x11. Sigh.

I used a typewriter, after finding out that the IBM Mag Card Selectric to which I had access was awful to use.

As always I had too much to say. The weeks and then the months dragged on and on. My friends, especially Sheila McKenzie and Wade Freeman, helped me lay it out. It took a year and a half to do the book, night and day, in an unending state of emergency.

### OUT THE DOOR IN '74

The first SIGGRAPH, practically the first computer graphics conference (see p. DM 80), was held in Boulder in the summer of '74. Tom DeFanti, Dan Sandin and I had coauthored a paper, so we drove out there together. In that first electric gathering of computer-graphics visionaries I passed out posters with sample pages from the still-unfinished book. That was the only advertising I did that had any effect.

And when it had been printed and bound, I think in August of '74, I schlepped home hundreds of cartons—load after load in my smallish car.

But then, as I started to send out free copies to reviewers and friends, I found to my horror that about a third of the books had the pages messed up. It was perfectly simple to get them right (just follow the page numbers that didn't say DM), but there had been a screwup. So I had to check every page of every copy, and the printer unbound those copies and redid them.

The book had what some found a rather alarming format: 11x16 (Big), with two front covers, one for each of the two separate books. The first front cover had a revolutionary-type fist (left over from the sixties) poking up inside a computer with the teaser:

You can and must understand computers **now**.

## TWO SIDES, TWO BOOKS

That was the main front cover. The **other** front cover had the title Dream Machines, and showed a longhaired youth flying, with a cape, finger outstretched to touch some sort of screen.

The book was two-sided, you see, because it was really two separate books joined at the navel. Dream Machines did not come before Computer Lib any more than Computer Lib came before Dream Machines. They were two separate, sweeping books on two important subjects. But the subjects were deeply intertwined and so the books had to be deeply intertwined. (I advocated refer-

ring to them both as "Computer Lib" just for convenience.)

Lots of other books have double front covers: various science-fiction books from Ace, for instance, and The Italian/Polish Joke Book. But having two front covers apparently makes stuffy and conventional people uncomfortable, they act as if it violates a law of nature or something. Well, I figured that was okay, since the book wouldn't be for them anyway.

## THE SMALL TYPE

It was only a hundred and twenty-eight pages. Unfortunately, those pages were packed with tiny type. How much no one knew until it was typed up by Microsoft Press to create this edition. It contained almost **three hundred thousand words**.

That is a big book. And it was very small type. This was a great mistake on my part, but I could not afford to have more pages printed. In hindsight, of course, I would certainly have laid it out on at least twice as many pages.

I wrote the book and published it myself. Since many people ask me what that means, let me explain. I got the idea, thought up each word, typed up the text, pasted it on sheets of cardboard, took it to a nice printer and said Here is Some Money Mr. Nice Printer, and I brought home a lot of cartons of books (made photographically from my sheets with no modifications) and proceeded to sell them.

That is what publishers do.

However, like people in most fields, some of them try to make a mystery out of it.

People keep asking me why I didn't go to a "real publisher." This is annoying on various counts.

First, I did sell fifty thousand copies. I think that qualifies me as a real publisher.

However, that is an evasive, weaselly line to take, because it should be stressed that we have freedom of the press in this country, and it is not hypothetical. If you make a book or a pamphlet and pass it out, that is by definition publishing it—MAKING IT AVAILABLE TO THE PUBLIC—and you are by definition a publisher. To exercise this sacred right of freedom of the press is in fact a sacred rite, the most sacred that we secular humanists know.

(I must admit, however, that conventional publishers wouldn't have run it with such tiny type. But then they would never have published it in the first, second or third place.)

The small type greatly limited its success; in a dozen years I sold a prodigious number of copies for a book published by its author. However, given the market and the book's positioning—the first personal computer book, already available when personal computers arrived—it was also a flop.

A more global way of stating the problem is that I never understood how much some people value **packaging** above **content**. Which is why fancy suits sell equipment, I suppose.

## SALES

The sudden sales were a shock, especially the unexpected order of a dozen copies by one of my great heroes, Ken Knowlton. Orders poured in by the dozen, then by the hundred. My friend Commander Hugo McCauley (USN, Ret.) handled the distribution through his business, Hugo's Book Service.

No magazines reviewed it. (Indeed, when a friend of mine presented it to the editors of Ms., the response was that it would be "of no interest to women.")

I had expected conventional publishers to fall over themselves making offers for it. Not a one.

But the word of mouth kept spreading. Colleges adopted it for courses. Many copies went to **Canada**, which bought far more copies per capita than the United States. A few counterculture bookstores picked up on it.

It has sold at least a hundred copies a month, rain or shine. Events in the outside world have seemed to have little effect. It just has a following, that's all. All in all, it sold about fifty thousand copies over thirteen years.

Hugo got tired of distributing, and I made some bad business arrangements. I remained the official publisher, but others had increasing say. How I lost control of

## All Publishing is Vanity Publishing

I heartily recommend Bill Henderson (ed.)'s **The Publish-It-Yourself Handbook**, \$4 from the Pushcart Book Press, Box 845, Yonkers NY 10701.



the book is like all other stories of loss of control (e.g., Buster Keaton and his studio). Given my strong preference for self-publishing, this very cooperative arrangement with Microsoft Press has been the best possible.

## GENERAL RESULTS

This book has been very widely quoted.

Many people—fifty or more—have told me that this book—or sometimes my other writings and speeches—changed their lives. A lot of these people are now in their thirties and forties and heading up companies of their own.

Many people bought copies by the dozen to give away. (Many people have also reported that their copies were stolen (see nearby).)

I've been told that the book is often used as an ice-breaker by people new in a company. To carry it makes some sort of a statement. They'll sit in the cafeteria with a Lib on the table, and potential friends will arrive.

The book was adopted for many classes, especially in Canada in the late seventies. It has adorned many coffee-tables and dwelt in many Johns (I heard of one company that had an official copy in the rest room.)

And overall, it seems to have been quite influential (as Adam Osborne put it, "ver-y influential"). Although so far I think it may have only hastened the inevitable burgeoning of various developments.

Anyway, its influence has waned: the combination of new stuff and new people in the field, more emphasis on packaging, booksellers' bewilderment (is it about hardware or software? What category is it in?) and its 1970s, hundred-flowers style, have rendered the original antique. So here is Computer Lib II, newly waxed, polished and packaged.

## THE REVISIONS

This was the fourth or fifth major attempt to redo the book, depending on what you count. There was a deal with a Madison Avenue publisher, which fell through. There was a "large type edition," re-laid out, now a collector's item.

There was also a version where I was expected to put in 50% new material, because some people mistakenly thought the book was out of date. (A few product names were outdated, but the ideas have

not changed. I am fascinated by the faddishness of people newly arrived in the computer field. Lacking exposure to the basic ideas, they get the notion that every new chip announcement changes the whole world.)

The size of the project was enormous. What finally worked was Microsoft Press keying the old edition all in, my printing it out, cutting the printout into paper scraps, re-sorting the scraps into the right sequence, and then copying the scraps into a new series of files with PC-Write, numbered so that Microsoft Press would know how to lay it out. Hundreds of thousands of keystrokes were required for this mechanical process (see "decent text systems," below). Much of this was done by the wonderful Lauren Sarno.

Writing the new material was easy by comparison with these inane mechanics, dictated by the present inane state of available software.

The sad thing about it all is that 90% of these efforts are unnecessary.

A decent computer text system would have obviated all the finding-and-retyping problems. I feel deeply for everyone who has trouble writing by conventional means, and who wouldn't if only decent systems were available.

I should stress that decent text systems do not exist yet; and the fact that everybody is used to "word processing" in its present forms is a sign of how backward the field is still. You still can rearrange slips of paper faster than you can rearrange text pieces on-screen!

By "a decent computer text system" I mean

1. A system that allows you to sort thousands of manuscripts, notes and scraps on-screen just as fast as you can sort their paper versions on the living-room rug—a 2<sup>1/2</sup>d array with partial overlap, modal overlap and flagging. Unlike the rug system, however, the computer system would also allow instances (not copying), so that an item could go to as many places as you want.

And, furthermore,

2. A system that allows detailed inter-comparison between any two documents or versions, with rapid scrolling in either (see p. DM 41).

ANYWAY

Enjoy.

# AMNESTY FOR YOUTHIEVES

Computer Lib is apparently one of the most-stolen books in history. Judging from what people have told me, ten thousand of you out there must feel very, very guilty.

But I am prepared to grant amnesty, provided you send the victim from whom you stole it a copy of this edition. And if you don't know where that person is, you must instead send it to a needy person, office, department or company head.

## SELF-DEFENSE AGAINST HAVING COPIES STOLEN

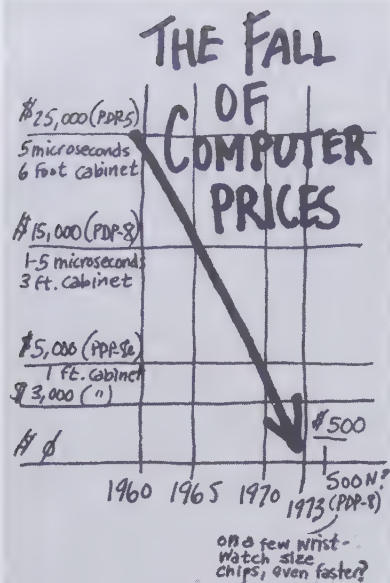
Whether this edition will be as much stolen is unclear, but we must assume the worst. I have three suggestions for making sure you always have a copy handy for inspiration and reference, to settle arguments or start them.

1. Drill a hole in your book half an inch in from the spine, and attach a stout cable through this hole to a brick. Use high-quality epoxy on the brick.

2. Or, attach it to your computer. That way to take the book they'll have to take the computer. Hmm.

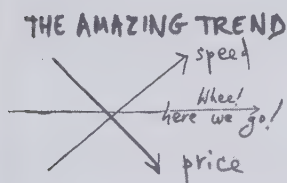
3. Best, buy this book by the carton (at your best discount price), and leave only one copy out. Put your name in it. It will disappear, as will the others one by one. More and more people will start to act as though they owe you favors.

David Hon says his life was changed by finding my book The Home Computer Revolution in the pocket of an airplane seat. He now heads a firm called Ixxion, which makes interactive medical dummies.



This diagram shows the amazing and unique way prices drop in the computer field. The prices shown are for the first minicomputer, the PDP-5 (and its hugely popular offspring, the PDP-8); but the principle has held throughout the field, and the downward trend will probably accelerate due to the new big integrated circuits.

Another example: an IBM 7090, a very decent million-dollar computer in 1960, was put up for sale at a modish Parke-Bernet "used computer auction" in 1970. If I remember aright, they could not get a \$1000 bid, because today's machines are so much smaller, faster and more dependable.



## FANCY WRIST-WATCHES

Tricky wrist-watches that keep track of more than one time zone are now available (as suggested in "Bucky's Wristwatch," p. 68).

*"Nelson's books...ventured dozens of predictions about the future of personal computers, many of which turned out to be strikingly accurate, a few of which turned out to be bad guesses. As a forecaster in a notoriously unpredictable field, Ted Nelson has done better than most..."*

—Howard Rheingold, *Tools for Thought* (Simon & Schuster), p. 229.

"Some of us minicomputer freaks see little need for big computers. Minicomputers are splendid for interactive and "good-guy" systems; as personal machines, to handle typing and bookkeeping; even for business systems..." ("The Minicomputer," p. 79)

# DID I SAY THAT?

"Gee," I keep hearing, "I just looked at Computer Lib again and it's amazingly up to date!"

This book is being republished as a Classic, which is very nice, but now in this section I have to tell you *why* it is a Classic, which takes even extra chutzpah.

## WHAT THIS BOOK PREDICTED

Hundreds of books, millions of people now say many of the same things; so much of the point is not so much what it says, but when it said it. (Written mostly in '73, with a big supplement in 1975.)

The book emphasizes personal uses and personal ownership of computers; as well as text systems, image synthesis and interactive design. Far-fetched and obscure then, these have proven to be central aspects of our world today.

The emphasis on *easy-to-use* systems is also to be found throughout the book. This too has proven to be central. (I dislike the term "user-friendly" — it's not the *friendliness* of a system that counts, it's the ease and clarity—but the term "friendly" does appear here several times.)

A few noteworthy passages on various subjects were eerily exact; some of them are highlighted here.

In going through this material again, I am amazed to see most of the important world issues of today clearly foreseen.

Starvation-filled lands, pie-in-the-sky defensive systems, and freedom of information will remain issues, unfortunately, for the indefinite future.

## COMPUTER OWNERSHIP & FAD

I made some very unusual predictions that came true; most notably, so far, the explosion of personal computing. Computer enthusiasm was then the province of only a few thousand zanies. That we computer fans have grown to millions now is a matter of history.

This book predicted a spectacular market in personal computers before there were any at all, which I expected to become a national fad on the scale that it did. It was obvious! I just considered how interesting these ideas were to me and my friends, and thought about how they would grab the American psyche.

I knew that when computer enthusiasm hit it would be an explosive fad, exciting, idealistic and full of energy.

Many others foresaw the widespread coming of little computers, but in an abstract kind of way—expecting them to be introduced surreptitiously in appliances, and to be gradually introduced through Market Research and Product Planning. Nope. Had we waited for big companies to bring out personal computers, we would be waiting still (see "How Computer Stuff is Financed," p. 116).

Predicting the coming popular computer magazines, and their function:

## ► Popular Computers

That the field has not been popularized by its better writers may simply come from an honest doubt that ordinary people can understand computers.

I dispute that. Through magazines, millions of Americans have learned about photography. Through the popular science-and-mechanics type magazine, and more recently the electronics magazines, various other technical subjects have become widely understood.

So far nobody has opened up computers. This is a first attempt. If this book won't do it another one will. And you better believe that Popular Computers magazine is not very far away.

Woops, here it is. *Popular Computing*, \$15 a year (\$12 if prepaid), Box 272, Calabasas, CA 91302.

It's now a different mag and it's on the newsstands, and there are dozens more, explaining away as suggested.



It was also obvious that most work in the future would move to the computer screen, as I said in the Dream Machines side. But this was an obscure and futuristic issue; I said we should all care passionately **how** our future work was to be designed. People debate this now in every office. (This passion was first bestowed, I believe, on my wonderful and immortal friend Douglas Engelbart, who thought of most of the things we would need for screenwork way ahead of anybody (see p. DM 16).)

Computer text systems will be, not merely the basis of office systems and newfangled systems for writers, but eventually the entire structure of tomorrow's publishing and libraries, and the battleground of tomorrow's liberties. This has scarcely begun yet.

The interactive computer screen has become the center of much work and play; eventually most. But *THE INTERACTIVE SCREEN, TOO, IS A NEW KIND OF MOTION PICTURE*. The design of interactive systems—from video games to office systems—is a form of movie-making, and everyone who's thought it was a branch of engineering tended to make awful systems. Interactive system *design* is now obviously—obviously to everyone, that is, except engineers—the center of our new world. But that doesn't mean anybody knows how to **do** it. (See p. DM 67 and most of Dream Machines.)

## Dinosaur

Many computer people, the author included, entertain certain doubts about the long-term usefulness of big computers, since minicomputers are cheaper, especially in the long run, and can actually be in the offices and homes where people create and use the information. Big computers are necessary for time-sharing (see p. 103) and huge "number-crunching" jobs (see "Grosch's Law," p. 83). However, it will soon be cheaper to put standardized number-crunching jobs in stand-alone or accessory hardware; see "Microprocessors," p. 96. Fans of big computers also argue that they are necessary for business programming, but that only means traditional business programming—non-interactive and batch-oriented. For tomorrow's friendly and clear business systems, networks of minis may be preferable. But makers of big computers may be unwilling to admit this possibility.

**Was this the first time the phrase "home computer" appeared in a book? (Fall 1975.) A very nice version of the 8 has appeared, an excellent home computer, with 8k of core, two floppy disks, key-scope, and wet-printer option; this is the "Classic," at \$12,000.**



## Private Ownership of Computers

Allusions to owning your own computer were shot through the book. It was then a new concept.

"It makes sense to own your own." (No longer elsewhere in this edition.)

"The idea of owning a computer may seem strange to some people, but with prices falling as they are it makes perfect sense." (See "The Minicomputer," p. 79.)

"But these prices too will come way down as the consumer market opens." (See "The Minicomputer," p. 79.)

"Soon a fully-loaded minicomputer will cost less than the best hi-fi sets. In a couple of years, thousands of individuals will own computers, and millions more will want to. Look out, here we go." (No longer elsewhere in this edition.)

"This is still a field where individuals can have a profound influence. But the wrong way to try it is through conventional corporate financing. Get your own computer, do it in a garret, and *then* talk about ways of getting it out to the world." ("How Financed," p. 116.)

"By reading this book in some detail, especially that difficult machine-language stuff (see "Rock Bottom" and "Bucky's Wristwatch," p. 68), or the pieces on specific computer languages (p. 56), you really should be mentally prepared to get into programming, if you dig it.

"Maybe you should consider buying your own minicomputer, for a couple of thousand. Or (if you're a parent), chipping in with several families to get one. Or a terminal, and buying (or cadging as cadge can) time on a time-sharing system. Maybe you should start a computer club, which makes it easier to get cast-off equipment. If you have a chance, maybe you *should* take computer courses, but remember the slant these are likely to have. Or perhaps you prefer just to sit and wait, and be prepared to speak up sharply if the computer people arrive ready to push you around." ("What Next," p. 177.)

### PERSONAL SOFTWARE

Computers could do all kinds of things for individuals, if only the programs were available. For instance: help you calculate your tax interactively till it comes out best; help the harried credit-card holder with bill-paying by allowing him to try out different payments to different creditors till he settles on the month's best mix, *then* typing the checks; *WRITING ANGRY LETTERS BACK* to those companies that write *you* nasty letters by computer; helping with letter-writing in general. You'll have to write the programs.

Yes, computers could do all kinds of things for individuals, and some of the programs are available. Limited and troublesome, but available.

Now you can try your tax interactively till it comes out best. (With tax programs or spreadsheets or various other means.)

You can do "letter-writing in general," of course, through text systems and "word processors" (see p. DM 27).

Writing nasty letters back: form letters are available on disk. They may not have the verve you want for dealing with nasty accounting departments, though.

The bill-juggling program, as suggested here, is damned important—and, so far as I know, still not available.

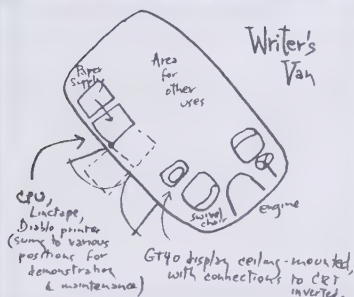


I always figured that the day of Computer Hobbyism would arrive when the folks at Heathkit offered a build-it-yourself computer. But you know what they came out with instead last year? A general interface for hooking things to the PDP-8.

"...better and better quack letters that look as though they've really been personally typed to you by a real human being." (JUNK MAIL, p. 152.)

*Yup. But who could foresee Ed McMabon's visage on the outside?*

## WRONG



Next in computer hobbyism will obviously be the Computer Van. Already vans come with swivel thrones, four-track stereo, color TV; so this next step is obvious. But most important, recreational vehicles can be purchased on very long time-plans, sometimes seven years. (MITS has a demo van with Altair, floppy disk, lineprinter. It drives around showing off. But they'll sell you one like it for a trifling \$29,000.) Now for mobile operation we redo the power supply...

The computer-in-a-van turned out not to be practical (though it would have had the benefit of providing long-term automotive fi-

nancing for your computer setup). So much junk you need around to make computers work, still.

Videodiscs were first announced in 1949, and I got really excited about them as a boy. In a first draft of this book, embittered by those early hopes, I predicted that they would never come out.

I tried to draw a moralistic line on body electronics, and said we shouldn't puncture people's skins (pp. DM 138-139). This Rubicon has been thoroughly crossed.

I also thought that the TRAC® language was going to have a big impact (see p. 61). It didn't. (I even tried to argue Bill Gates out of BASIC, I think it may have been in '76. That's funny enough to admit.)

I also thought the Apple II, with no lower case, had no future.

"So many programs exist for the PDP-8, and so much sentimental fondness, that it will be with us for the foreseeable future. We may assume that a PDP-8 on one or two wristwatch-sized chips is only a year or so away."

Right and wrong. The PDP-8 was put on a chip by Intersil, but it wasn't a hit. And sentimental fondness or no, the 8 is now gone.

## HYPERTEXT AND HYPERMEDIA

The coming of hypertext, foreseen in various of my articles of the sixties, is discussed on the Dream Machines side. Curiously, that only started coming true in 1986, when suddenly the word "hypertext" came into common use. A notable hypertext program called GUIDE is available for the Macintosh, and, at the high end, a hypertext system called Note-cards is available in LISP on Xerox's D-machines (see p. 92).

Explorable systems—hypermedia—are cropping up everywhere, from shopping malls to art museums. The Smithsonian Institute has installed a hypermedia laboratory (the National Demonstration Laboratory for Interactive Media, under Glen Hoptman).

Tomorrow's hypershows will be increasingly based on branching cinema (hypermovies). The time is finally ripe for these to hit the consumer market (see CDI p. DM 65, DVI p. DM 99, the Playstation p. DM 23, writable videodisc or WORM drive p. DM 51). But they could have happened sooner, based on film, had American industry had the vision.

## IMAGE SYNTHESIS AND SPECIAL EFFECTS

The world of show biz, including audio and motion pictures, is being transformed as I said—through audio and image synthesis (explained throughout Dream Machines), computer-based movie editing (p. DM 49), and other methods. By now we've seen image synthesis used in major films (notably "Tron," "The Last Starfighter" and "2010,") and in thousands of TV commercials and station breaks.

Techniques of computer-controlled puppetry (see p. DM 56) bring us the beloved E.T. and the whole new boy-meets-alien genre.

## WORDS

Several words that I coined have caught on: hypertext, hypermedia, hyperfilm and hypermovie, hypershow; also cybercrud and softcopy.

**Fantics**—my term for the combination of explanation and showbiz (p. DM 74) has not been picked up by anyone, although my more recent term **virtuality** (p. DM 68) has.

"Compound document" is now popular; the term arose in the Xanadu work in the mid-seventies.

Recently, someone surprised me, however, by saying they thought the terms "image synthesis," "interactive system" and "interactive design" first appeared in this book. Those are steep claims which I don't actually **make**, but I thought it would be nice to **mention** them here.

## PREDICTIONS ELSEWHERE

My best predictions were in a book I wrote during the nineteen-sixties, but that was declared Company Secret and I can't publish it. Much of it came true exactly. Again, it's not what I said but when I said it.

On a more mundane level, two predictions from my 1977 book *The Home Computer Revolution* have proven themselves interestingly.

For one thing, I predicted that ten million personal computers would be sold by 1980—the most radical number so predicted, I believe. While that ten-million figure was not reached until 1981 or 1982 (depending on what you count as a personal computer), still I believe my estimate may have come closer than any other.

I also predicted in that book a ferocious struggle between the existing computer centers in large organizations (companies, universities)—oppressive monopolies generally run by organizational politicians—and their imprisoned clients in those organizations who would relish the new freedom of having their own computer. This has happened spectacularly, and a large part of IBM's current strategies are built around trying to keep its big mainframe computers necessary, if not wanted.

## NEW PREDICTIONS

My last predictions are these:

Computer "basics" as we have known them will go away. New software designs will make it possible for everyone to use computers casually in every area. Such casual usership will begin at the age of one-and-a-half or two.

Tomorrow's computer systems for home and business will be quite different from those of today: more visual, understandable and enjoyable, **far** easier to use, and **far** more powerful. There **will** be an easing of the personal complications of life, the clarification of all the schedules, payments, correspondence, accounts, and addresses and personal things to be kept track of. 🐾



"For historical reasons computers have been used mostly with numbers up to now; but that is going to be thoroughly turned around." ("Data Structure," p. 70.)

Vast quantities of text are on line. Note the quantities of on-line documents proffered by IBM for the Sander suit, p. 137. And see text networks, p. DM 39. What offices keep on disk is mostly text.

GRAPHICS: the data structures and outputs of picture processing (p. DM 81) and image synthesis (pp. DM 100-119) occupy immense amounts of storage. Not to mention NASA's great libraries of planetary pictures and the Landsat photos.

On the other hand, humongously large streams of numbers are acquired and saved by such techniques as oil exploration. So whether text and graphics will soon outweigh "numbers" is not yet clear, nor are we likely ever to find out.

## Products ▼

A few software products exactly fit specs that were laid down here. One is Filevision, a program for the Mac that lets you poke part of a picture and branch to something else; this is what you needed for "hypercomics," pp. DM 135-136. (GUIDE, mentioned elsewhere, also offers branching diagrams like Filevision.)

However, one design from this book has appeared on the market just in the last year, practically as I presented it: MacSpin, a data analysis program for visualizing and rotating points in multiple changing dimensions. (See p. DM 49.)

## THE MOST TERRIFYING PRESCIENT REMARK

"Imagine if the Watergate mob had had control over national data banks. Enough said."

*This was all too prescient. The current gang of White House crooks started hatcheting our freedoms just as these words suggest. Admiral Poindexter, of the Irangate mess, issued a secret directive saying it was necessary to restrict access to non-classified material in data banks. Effectively he sought to destroy the most precious American freedom with a stroke of the pen. See "Issues," p. 167.)*

*I said that we might artificially create "the virus that may wipe out mankind." ("The Mitiest Computer," p. 94.) At that time there was no plausible candidate for such a virus. Now there is. See p. 158.*

On one of the big predictions, the collapse of IBM, some think I goofed. (Steven Levy in Hackers makes much of how wrong he thought I was.)

IBM's marketing system in 1974 was mainframe-centered, supporting the computer centers in suppressing interactive systems. I did not imagine that IBM could move into the personal marketplace, and expected them to be the enemy of computer freedom forever. That's why I put such careful emphasis in the original Computer Lib on the mechanics of IBM's repression.

But the IBM leopard changed its spots. IBM brought out the PC, a personal computer that could be highly interactive and used by everybody for everything (actually an upgraded and nastier Apple II). And in doing so, IBM regained its virtue, because the PC was a contribution to human freedom. (But this virtue was only temporary, of course. The contortions IBM went through to cripple the PCjr and make its monitor incompatible showed that basically their strategies remain the same.)

But in my opinion the PC has only postponed the crash of IBM. By gallantly seizing the initiative in the personal-computer market, they got temporary profit—but they have been unable to hold that market with their traditional Controlled Upgrade Path to larger machines, and the copies of the PC from other manufacturers—the "clones"—have taken much of the market.

Meanwhile, because IBM made their communication programs so insanely complicated (many think to lock out the competition), they are now themselves being deservedly locked out—as companies who need communications buy whole systems from DEC.

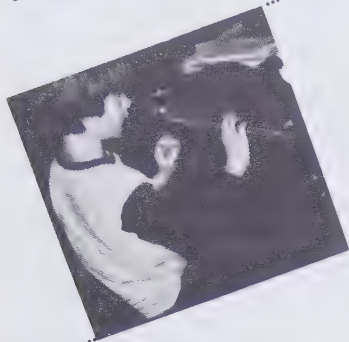
IBM is on the defensive on every front, trying to stop the users from leaving their big machines.

Their profits have gone down, their stock has gone down, and they are early-retiring all the people they can. So I believe I was right about the crash of IBM. It took longer to start, and is happening in slow motion, but it is happening indeed.

*The ABM system was a boondoggle to shoot down incoming missiles in the nineteen-sixties. It has been dandied up in lasers as the Strategic Defense Initiative, and is more idiotic than ever. See "The Ray-Gun Initiative," p. 171.*



## Kids and Computers



Kids love computers. They belong together. This lad flips panel switches on a Nova.



☛ The computer screens of the 1990s will not be like today's clumsy screens, but will be of the flight simulator class, such as the IRIS (see p. DM 113). Graphics and sound coming from disk and repository network will blend with graphics and sound generated in the user's computer. This will permit new, spectacular, intricate, extremely vivid multimodal interactive media. You will swoop through, magnify and interactively manipulate throbbing, amusement-park-like visualizations of everything—in color, three dimensions and sound. I call this the *Playstation*; see p. DM 23. And so the *design of new visualizations* will become the forefront (as already discussed on the Dream Machines side.)

This will blow education sky-high, combining simulation and TV in materials deliverable from computer networks, and *finally* breaking away from the funda-

mental problem of the school system—the Schedule.

A new era of comprehension will begin.

The *real* office of the future will be as clear and simple as Pac-Man, and have comparable sound effects (which the user may mute as desired).

"Word processing," spreadsheet and database will no longer be distinct.

Today's intricate languages and queries will be replaced by far more visual (and audible) environments.

Today's inane "business graphics" will be replaced by far deeper responding visualizations.

"Files" as we know them will go away; you will store entities with multiple names or no names, hierarchical or overlapping, and interconnected any whichway. "Backups" will be forgotten; correspond-

ing safety measures will have become automatic.

It is plain and manifest destiny that electronic publishing will replace most of paper publishing in the next decade or so. From your reading machine (whether or not it's *called* a computer) you will summon up text and graphics from all over, stored on digital servers (whether or not they're *called* computers) around the world—a universal repository network. This is both good and bad. You will be able to get at anything rapidly in vivid new ways, but this will also concentrate archival danger, and issues of "freedom of the press" in entirely new ways.

A world of information plenty will open its doors—and all of civilization will experience a new Renaissance. Just barely in time to save humanity from destruction.

Here the predictions rest.

## NOT YET

The term "program follower," and the pointing hand for representing it, seemed to me to be a lucid and clarifying way to look at computer mechanics. To the best of my knowledge only one person has picked up on this, but that person is the remarkable graphic artist and Artful Intelligencer Scott Kim, so that kind of makes up for it.

Ordinarily the term "logic" connotes, not just rigor and exactitude, but a stylistic squareness. I thought it a nice turnaround to show "logic" as a squiggly rosette in my computer diagrams. This too has not caught on yet.

### ANIMATED AND SEGMENTED ICONS AND CONTROLS

Screen icons, developed at Xerox PARC and popularized on the Macintosh computer, seem to be everywhere these days. I had come up with the same idea independently in the sixties, but in my designs, different parts of the icon could be manipulated with useful meaning. This has not caught on yet. (See pp. DM 41-43.)

Another design that I consider obvious for controlling complex interactive graphics is the Walking Net, presented here (p. DM 47), which keeps the user's choices simple and unconfusing but still rich. That's still to come.

"The way computer prices are coming down, through integrated circuits ever more powerful and cheap, that PDP-10 can be sold for \$500 in... (check your choice) \_1978 \_1979 \_1980 \_1981 \_1982." (No longer elsewhere in this edition.)

Not quite. The processor of the PDP-10 was about the equivalent of the Motorola 68020 or the Intel 386—chips that are just now getting boxed up and still in the \$2000 range, without peripherals. Also, the 10 had a lot of interfacing and output channels. So the year for that price is about 1988.

"Computers should make things easier in both our work and our private lives, and should help lighten our loads and enlighten our minds, clarifying the complexities of everything." (IBM section, p. 144)

"USING A COMPUTER SHOULD ALWAYS BE EASIER THAN NOT USING A COMPUTER. If it isn't, you (or your company, or your state) may have been sold a bill of goods.

"OR, they may have decided your inconvenience is less important than something else.

"In any case, you have a right to ask sharp questions." (No longer elsewhere in this edition.)

Unfortunately, in most areas we're not there yet.

"In such a world, computers are not a tool but a way of life. The computer is toy, pet, checkerboard, music box, and TV. Computers are for making music, computers are for getting people together via community memory, computers are for letter-writing, computers are for art and movie-making and the animated decoration of the home."

("The Greening of Computerdom," p. 60)

"I've finally given up on trying to reach professionals, who evidently need a thick gravy of technicalities to make the obvious palatable; with this booky I am taking my case to the People. It is there, anyway, out in Consumerdom, that the real action is going to occur. So the important thing is for everybody to know what's really possible, and what they could have."

("Flip Out," p. DM 149)



# WHERE ARE THEY NOW?

*It needs explaining. I originally planned to be a writer and independent film-maker, but when I took a computer course, in 1960, I said WOW, this will be the way to write and make films—and started to design software for writing and movie-making.*

*In a personal decision with many consequences, I decided to let my manuscripts wait until the tools were ready, and simply **accumulate** notes toward that day. My reasoning was simple: using conventional tools, I could only expect to do a small number of projects during my working life. With the more powerful tools I envisioned, I would be able to accomplish far more. (This has been a long-running irony, since the tools are still not ready and the notes sit and seethe. A big part of the work set aside is a philosophical sys-*

*tem I now call General Schematics, which combines elements of Korzybski and various social sciences, including a combinatorial theory of strategy. None of this is published. The material is waiting still, ripening in storerooms.)*

## SIXTIES

*I designed interactive systems, in imagination and on paper, all through the sixties—movie-making systems, office systems, text-handling systems, personal life-management systems.*

*I wrote a lot of articles in hopes of backing. I published the hypertext idea around 1965, and enjoyed a brief celebrity; but then people lost interest. I worked on the first visual word processor at Brown University.*

## THE FIRST COMPUTER SINGING COMMERCIAL (and talkin' blues), by the author.

(As heard at the National Joint Computer Conference, 1973.)

How are we going to sell the Home Computer?

Well if you want to sell computers, let me tell you what to do:

You've got to talk to the housewives, and the children, too;

No one wants to program, they want something they can view...

It's got to offer fun, and it's got to offer truth;

It's got to give you something that'll lift you from the booth;

It's got to be uplifting to the Lady from Duluth.

You've got to have a vision; you've got to have an angle;

You should maybe sing a jingle (in a way that doesn't jangle);

It's got to have a tingle, in a way their minds can't tangle—

(Continued on p. DM 145.)

*Project Xanadu gradually took shape: first as a console for authors and decision-makers, allowing extremely complex alternatives to be intercompared; then as something much bigger (see p. DM 141).*

## SEVENTIES

*I applied for a patent on an image synthesis system. An appointment to the University of Illinois gave me time to write this book.*

## AFTER THE BOOK

*Again it was briefly the Celebrity Trail, making lots of speeches. Many were the colorful pioneers I tripped and toked with.*

*Orders for Computer Lib poured in, by the dozen and later the hundred. Financially helpful, but never enough to free me for my real work.*

*So I still had to keep getting jobs I didn't want. I gave speeches, taught here and there. Precarious scrapes, many moves, lost cartons.*

*I backed some wrong horses. I committed to the Sol computer, the TRAC language, and a column in ROM magazine. Bingo. May they all rest in peace.*

*But then the era of the Suits began, and more conventional audiences were discomfited by my outlook. (Columnists like Adam Osborne, Peter McWilliams and Jerry Pournelle took over pundit slots, writing niches I'd begun.)*

*I had studied and long prepared for the coming of personal computing, and no one was better positioned when it came. But there was an attitude mismatch, a paradigm problem.*

## SOFTWORLD AND ITTY BITTY

*Here I had been designing personal software all these years, knowing personal computing was coming; but I could find no backing for my software, because it wasn't like anybody else's.*

*I became a partner in the Itty Bitty Machine Company, a computer store in Northern Chicago, partly so that the store would back integrated software. (Somewhere there is actually a business textbook in which the question of whether to continue with Itty Bitty's software development is presented as a case study for discussion.)*

*I announced an integrated software package for home and small business at (I believe) the West Coast Computer Faire in spring of '78. It was to use the TRAC language and work on cassettes (disks not being widely available on the personal market). I was going to call it Softworld, and its constituents had nice names too—Planorama, Pictrola, Ledgerdomain, Bill Juggler.*

*It was too soon. (The term "integrated software" had not yet been invented, and people thought the idea was crazy.) For this and that reason it did not reach demo.*

## XANADU 80

*Computer Lib was a disappointment in many ways. It did not free me financially for other work; it did not bring in backing. But astonishingly, it succeeded in one of its most important purposes. At the end of the book, which was hidden in the middle, was an implied invitation to*

*I have trained one world-class interactive designer, Stuart Greene (presently at Apple), and one world-class generalist and internals designer, Mark Miller. Their talents were their own, but I know I helped shape their ideas.*

*I'm still hiring apprentices, at the same rate of pay (how much have you got?) But so far that policy has brought in some of the best and the brightest.*

Apprentices  
and  
Students



appropriately clever people to come join the Xanadu project. And, of all things, they did. The system was redesigned by a new and remarkable group in 1979 and 1980; the current system embodies that new structure. (See p. DM 141.)

In 1980 and 1981 I was the editor of Creative Computing, a successful magazine that strove for a counterculture image. In late '81 I was hired by Datapoint as a software designer—at last!—but after two months they called my designs “too futuristic” and made me a technical writer.

### BLOODIED BUT UNBOWED

Since leaving Datapoint in 1984, I've been speaking and writing and waiting. I have consolidated all my old designs for interactive software into a new single unified design.

It is intended to improve significantly what people can do with computers; help with my own day-to-day work in a fashion I consider acceptable; and provide a concrete lesson in software design and unity.

In 1986, due to the sudden ascendance of hypertext, I found myself abruptly promoted from Lunatic to Visionary, with new interest in my work. I hope you will share with me the excitement to come as we sweep aside old methods with tomorrow's interactive systems; and bring forth the new literature, art educational system and way of life of the hypertext future.

I have been pleased to meet, and otherwise enjoy, the variety of clever, charming and/or lubricious persons who have sought me out since the first book appeared; as well as all the speaking engagements, soirees and whatnot.

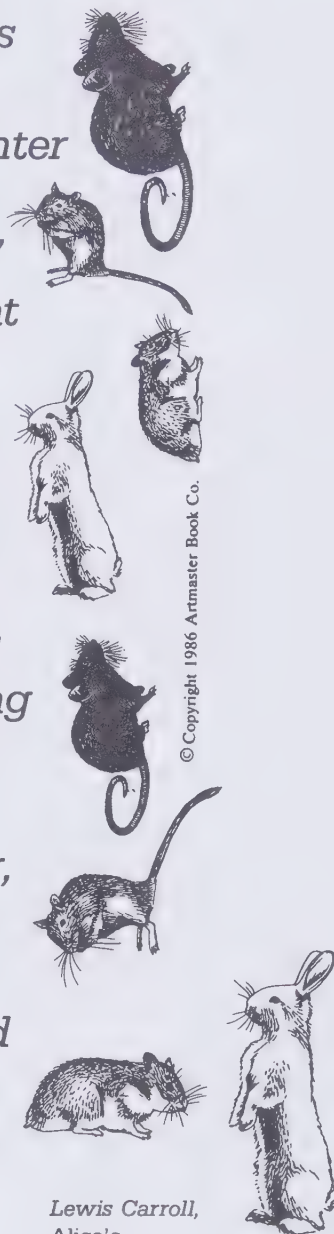
I am delighted to receive relevant material and communications of any kind, although problems of time, disorganization and mood often preclude a Personal Type Reply.

It has been a real lift for my morale to share some of these ideas and enthusiasms with a wider public at last. It is you, finally, who have to care; and I am very glad you do.

vy

**I have been talking revolution all these years—not political revolution, but intellectual and social revolution—fixing education, enticing all children to be smart (and making the world safe for smart children), and giving us all access to the panoply of thought and knowledge. The human race has to get very smart very fast. This is crazy?**

“It was much pleasanter at home,” thought poor Alice, “when one wasn’t always growing larger and smaller, and being ordered about by mice and rabbits.”



© Copyright 1986 Artmaster Book Co.

Lewis Carroll,  
Alice's  
Adventures in  
Wonderland,  
Chapter 4.

*It is the best of times, it is the worst of times. Since this book first came out, everything has changed, nothing has changed. The hardware is different, the issues are not.*

*In 1974 there were no personal computers. The order was maintained by a collusion of administrators, systems people, pompous academics and frantic users. Not to mention IBM.*

*(From the 1975 supplement):*

Dig It, All:

## THE GREENING OF COMPUTERDOM

1975 may be thought of as the year in which the computer underground suddenly appeared in full force. The Altair was probably the big crystallizing event.

Not that there wasn't a counterculture before. There were the games-players at every university, the prank programmers (see pp. 112-114), and, wherever computers are the center of things, a shared experience of mischief and breakthrough. There was Computer People for Peace, a cliquey and unapproachable group with booths at the conferences (at least, their backs were always turned when you wanted to ask questions). There was the hobby fringe.

But now it's gone different. Instead of pretentious company names meant to appeal to obtuse businessmen, like Performance Measurement Systems Consultants Group and Bottom-Line-Tronics, the new companies have rock-group names like General Turtle, Inc., The Sphere, and Loving Grace Cybernetics. In this new computer counterculture, the main computer companies are not IBM and Honeywell and Univac, but DEC and MITS and General Turtle; the standard computer is not the 370, but the 11 (or possibly the Altair or the 8). The standard language is not FORTRAN or ALGOL or PL/I, but BASIC. Instead of the big color TV that middle America wants, the underground computernik dreams of his own graphic setup forever running The Game of Life in color (see p. 115). (Of course, that'll also require the color TV; see “Bit Maps,” p. DM 92.)



( From the 1975 Supplement )

# THE ALTAIR STORY

It began with a bang last Christmas: the cover of *Popular Electronics* showed 'a computer you can build yourself for only \$400!'

It was real. A young firm in Albuquerque called Micro Instrumentation and Telemetry Systems, or MITS, had finally done it: a computer for well under \$1000. In a box not much bigger than a typewriter, a machine comparable to the Univac I. They called it the Altair 8800.

Of course, in a way this was an obvious step. The MITS computer was simply the packaging, as a computer, of a specific integrated circuit chip that had been on the market for some months. This chip, the Intel 8080, is a micro-processor, (see p. 96), generally employed for fixed purposes in cash registers, pinball machines, and the like. However, to make it a "general" computer—with the engineering, hookups, and accessories that entailed—would be no small matter if taken seriously.

But MITS took it seriously, and offered with the Altair a small but complete line of terminals, disks, printers, interfaces, and, most important, service facilities.

The firm had innovated before, notably when they brought out the first handheld calculator several years before. Just as they correctly anticipated that demand, they foresaw this one.

They also chose unerringly the right market to begin on: electronic hobbyists and kitbuilders. The kit-maker enjoys the challenge of building a machine from only a diagram and a box of parts; and to be far from a repairman holds no terrors for him, for he is the repairman.

The price drop was not as dramatic as it might seem to the general public; nor is the computer quite as cheap as it seems at first glance. Contrary to a public impression, created by IBM and a muddled press over the years, that computers are huge and cost millions of dollars, very good computers have been available lately for a couple of thousand, not counting accessories.

But the accessories present a problem. On that score, the apparent rock-bottom price of the Altair may have been misleading, especially to kit-builders. A computer itself is a limp dishrag without memory, terminals, and programs—all of which pad the cost of the package. By the time you've added 8K memory, a terminal, and BASIC software to your kit-built Altair, a thousand dollars has flown (\$1400 if you buy it already assembled). Then if you want the disk (and who doesn't), that's at least fifteen hundred more.

Now, kit-builders just starting may not see the point of all these fripperies; they aren't used to powers like that of a full computer, so coming to realize the immensity of it all may be a gradual awakening, with many happy soldering experiences on the way. Others may be brought up short as they sense what they're getting into.

This is partly a problem of MITS' trying to reach two consumer groups at once: the kit-builder, who may have thought a computer was a fancy switch-

box, and now must enter a world he doesn't know, and the computer sophisticate, who looks at the bottom line for the cost of a complete package.

MITS prices are quite reasonable. If you buy a kit for anything in the Altair line, it's generally about 25% less than the assembled and fully-checked-out version.

The basic computer kit costs \$439 (\$621 assembled), but ignore that; it's like a car without an engine, seats, or wheel. A complete package (their "BASIC 1" set), with the computer, 8K of memory, terminal, and 8K BASIC language is \$1391. A more high-powered system with 12K of fast memory and *double floppy disk* is \$6650, complete with their Extended BASIC. There are many separate items, plans, and options; it is possible, of course, to buy a packaged system from them for as much as you want to spend.

Indeed, MITS' low prices aren't that low. When it comes to price, they are about 50% ahead of the conventional competition. For instance, their \$5000 setup (with terminal and disk) might be taken as roughly equivalent to the DEC Classic at around \$10,000. (See p. 11.)

But what you usually pay for in this field is service and fringe benefits. The fundamental test, it seems to me, is whether you can come back to the company with your problems. (They even answer correspondence about their customers' computer troubles.) MITS' principal contribution is really in the thought they have given to their market, and the depth with which they are serving it. They no doubt anticipated competitors who would supply accessories and undersell them. But they see the advantage in this: they even give out their mailing list to competitors who sell Altair memory boards cheaper! They were not out just for a quick buck; they appear to be thoroughly committed to full-spectrum computer service.

In eight months, MITS has gone from twenty-five to a hundred employees and sold OVER FOUR THOUSAND COMPUTERS, which is something like two or three percent of the computers in America. Today, the electronic nuts; tomorrow, the world.

Bob Albrecht, caliph of counterculture computerdom, highly endorses Altair Extended BASIC. Says it's terrific.

## WHERE ARE THEY NOW?

*MITS got bought by Pertec or Percom, I forget which, and disappeared without a trace. Ed Roberts, who was its president, got an M.D. And David Bunnell, MITS' very effective head of publicity, is founder and publisher of PC World, MacWorld and Publish! magazines.*

*Albrecht is still a counterculture computer figure in the Bay Area, but his high profile has gone down somewhat. He does claim to have been one of the first to put fantasy adventure-gaming on computers.*

*And Altair BASIC became the first product of a company called Microsoft.*

IN ONE ERA  
AND OUT  
THE OTHER

*Many things which represented blows for computer freedom in one era, when they first appeared, became oppressive as time went on. Yesterday's freedom can be today's drag.*

*Marxists would call it Dialectical Progression. I prefer to call it Paradigm Shift.*

## INNOVATIONS ONCE, BUT NO LONGER, LIBERATING

Computer Languages  
Operating Systems  
DEC-style operating systems  
"Word Processing"  
The Apple II  
Spreadsheet Programs  
The Macintosh  
Wizzywig  
Inanimate Icons  
Disconnected Windows



☛ In such a world, computers are not a tool but a way of life. The computer is toy, pet, checkerboard, music box, and TV. Computers are for making music, computers are for getting people together via community memory, computers are for letter-writing, computers are for art and movie-making and the animated decoration of the home.

Computers are for games; a vast number of interactive game-programs are published and swapped around. Almost all are in the BASIC language. (Bob Albrecht's *WHAT TO DO AFTER YOU HIT RETURN* is said to be definitive—\$7.50 from People's Computer Company, 1919 Menalto Ave., Menlo Park CA 94025. See also their magazine *PCC*.) PLATO games, a somewhat different subspecies, are discussed on p. DM 20.

The underground computer magazines have become a blizzard. Albrecht's sprightly and successful *PCC*, originally oriented toward high and grade schools, has now branched into hobbyism as well. On the hardware side there is *The Computer Hobbyist*, and now a slick new hobby magazine, *Byte*, with a first printing of 50,000. On the educational side there is a swell new magazine called *Creative Computing*.

Even for those coming anew into the field—the radio hams and amateur telescope-makers who've laid their Master-Charge cards on the line for the Altair—computers represent a whole new social life. Amateur computer clubs have drawn startling numbers; for instance, the Los Angeles and San Francisco groups are currently pulling 100 members to their weekly meetings.

This book and its surprise success probably rate mention of some sort in the world of underground computerdom, '74-'75; although my underground status may be in jeopardy. I had intended to bypass the computer establishment, and certainly not expected to become assimilated therein; so the dozens of university class adoptions have come as a considerable shock, as have the acceptance and legitimation I had long since given up on. My heartfelt thanks for this response and I'll try to live up to it.

### THE HUNDRED-FLOWERS PERIOD

*It's been a wild dozen years since Computer Lib came out in August of*

*1974; in December came the Altair announcement, and the rest—the era of personal computing—is history.*

*The famous Altair cover of Popular Electronics announced personal computers to the American public; and thousands of orders poured into the would-have-been-bankrupt firm in Albuquerque, people drove all night to buy their kits, and a new dream exploded into American consciousness.*

*It's impossible to recapture the feeling of that earlier time, the personal excitement and joy many of us felt.*

*IT WAS HAPPENING AT LAST*  
*Most people thought it unimaginable that there would be personal computers. Academicians had said so, but in an abstracted sort of way. But now here they were, and it was about as abstract as the circus coming to town. It was a SCRAMBLE, a GOLD RUSH. And it was like any other revolution. Dreamers, misfits and malcontents (such as the author) streamed out of their garrets and cabins. They bought, they studied, they built, they programmed, they read lots of magazines.*

*Everybody thought they could do everything.*

*Your computer would cook your dinner, lullabye you to sleep, babysit your*

*children and make epic films, just as soon as you got the software ready.*

*The focal, crystallizing moment for the personal computer field was the First World Altair Computer Conference, in the summer of 1975. (There was no second World Altair Computer Conference.) To MITS' headquarters, and an Albuquerque hotel, descended an assortment of crazed and determined individuals, about a thousand I think, who sat late into the night listening to folks like Adam Osborne (not yet terminally pompous) and Dave Ahl and Les Solomon and me, talking about how it was all going to be.*

*The people who bought dinky computers then were very different, different both from previous computer users and from users now. They were more like radio hams—computer hams, you might say. Quite a few, it seems, were very fat; why, and why no longer, I cannot say.*

*And the ordeals of users! When the first Radio Shack computer came out, users had to run a special program to "debounce" the keyboard. And users of certain boards kept burning out something called the Data Delay Chip, which could only be gotten in the middle of New Jersey; you had to be put on a waiting list.*

### CAREER PATHS THROUGH A UNIFIED (and very strange) FIELD

*At a conference recently, people compared careers.*

*One guy said he'd started out doing simulations of the Space Shuttle, and more recently had gotten into hyphenation for text systems.*

*Another guy had started in text systems, working for Douglas Engelbart (see p. DM 16), and more recently become famous for creating a video game called WORMS.*

*We were all watching to see if someone else was infringing on our specialty or our little twist. It was difficult to get people to work on the same thing. We all just had great plans with no one else to listen to but other people with their own great plans.*

*Lee Felsenstein on Life in the Homebrew Club (quoted in Moritz, The Little Kingdom, p. 111)*

*"...we developed a priesthood of our own. Sure, it was an open priesthood, easy to join for those who'd take the trouble..."*

*Jerry Pournelle (Infoworld, 20 April '87)*

### THE CLUBS

*The Clubs! Biggest were Boston (the Boston Area Computer Society) and the Bay area (the famous Homebrew Computer Club, which grew out of potluck dinners at PCC). They sprouted everywhere, to swap and trade and exchange information on how actually to make those things work.*

### THE MAGAZINES

*The Magazines! First came PCC, Byte and Creative Computing, then ROM and Dr. Dobb's Journal of Computer Calisthenics and Orthodontia, and Intelligent Machines Journal and Appletalk, and on and on. At first you bought every personal computer magazine that came out. That was possible for about two years, or four if you were very dogged about it. But they began to pile up unread, and there was no place to put them, and it gradually became clear that it was a whole new field every six months.*

## The Altair Factory



aturally it had to be in Los Angeles. The first "computer store," it seems, is at 11656 Pico (at Barrington), West L.A., 1/4 mile west of the San Diego Freeway; 213/478-3168. Hours are 2 to 8 Wednesday to Friday, 10 to 6 on Saturday and Sunday. It's called the Arrowhead Computer Company, and they stock a line of Altairs.

## WHERE ARE THEY NOW?

*Dick Heiser never wanted to be a tycoon, and sold the store when it stopped being fun. Now he teaches and writes. He and his wife Loretta recently took off nine months to go on the Great Peace March from L.A. to D.C.*



## THE STORES

The world's first computer store, opened by Dick and Lois Heiser in L.A., was a shock to everybody. So was the land-office business they did. (They kept a big quote from Computer Lib in the window.) And suddenly there sprang up computer stores everywhere.

The early stores were a merry madness—bare feet, jollification and gossip. Paul Terrell started and franchised the Byte Shops, where he stocked a lot of Computer Libs and launched public sales of the first Apple computer, somewhat to the surprise of Steve Jobs. And Portia Isaacson, at her computer store in Texas, delivered the famous dictum to free the corporate users: "I'll sell you a computer under any name you want!" (See Computer Center Politics, p. 161.)

## THE COMPANIES

The Companies! Pickles and Trout. Beagle Brothers Software. Imsai, who made the red-and-blue Altair copy that worked **well**. Software came in polyethylene bags, a cassette and photocopied instructions with very bad artwork.

We kept track of strange developments with care. A firm called Processor Technology started up by selling circuit boards for the Altair that would hold A WHOLE 4 K! One long soap opera concerned a Dr. Suding, somewhere in the west, who was building a computer with **both** an 8080 and a 6800, plus cassette drives! Processor Technology brought out a beautiful computer, the Sol, which had wooden sides and a fine big keyboard, and came out well ahead of the Apple, and should've been the big one. But they floundered.

And then there was my favorite product, which was announced but I don't think ever delivered: Alfred the Office.

## ARRIVAL OF THE SUITS

Some of us thought it would go on forever, that spirit. No, it was not to be. Sensing an industry aborning, the conventional business types began to move in. People arrived who judged your ideas by your clothing. The Suits didn't love computers; they were taking over our field for intrigues and power plays of their own.

Creeping respectability set in.

The Apple started as a one-board computer that a couple of guys were showing in a booth. But they had a good, complete product, and its impetus was billowed with large infusions of money from the standard venture-capital world. Scarcely changing their good product, they went for Image and left the counter-culture behind. (See p. 124.)

Apple's third wind came from Visi-Cal, something never seen before, an "electronic spreadsheet." At first it ran only on the Apple, and hundreds of thousands of companies bought the Apple to run this indispensable new tool.

The second wave of computer stores began, with carpets, run by carpetbaggers and staffed by greasers. Hiring as fast as they could, they filled the floor with ignorant fast talkers who could pick up phrases quickly and use them as if they understood them.

Then the Empire struck back.

The IBM PC was big and ungainly. (When I saw that preposterous keyboard I practically fell over laughing.) But it caught on. And it was marketed as no personal computer had been marketed before.

## THE PEAK

In 1982, video games were a major center of American culture. (Pac-Man alone grossed 500 million dollars.) Around 1983, **every tenth book sold in America was a computer book**. But that was the peak. By the end of 1984, computer-book sales were way down. Discouragement, probably. People realized personal computers **weren't** going to simplify and clarify their lives, but the opposite; they would have to occupy one small corner of affairs and be kowtowed to. The dream had, for many, guttered out.

## THE NEW ORDER

It's big business now, because the PC (and its copies) captured the corporate desktop. The Suits are in the saddle. Beginners are now cowed by "the way it has to be." Across the land, harried secretaries and middle managers are losing their files and making twenty printouts to correct small errors in outgoing letters. And we have Designer computer stores, with carpets and painted Motifs on the wall, staffed by cunning survivors.

Order has been restored; the hundred-flowers period is over; pinstripes talk only to pinstripes once again. You have to be a High Roller to start a software company, or so they say. Only certain favored developers, Big Players, get an advanced look at the new computers.

The Home Computer Revolution has settled into the Desktop Computer Industry.

Jim Warren's Intelligent Machines Journal has become an industry tabloid, Infoworld. And a magazine that started in the People's Computer Company as Dr. Dobbs' Journal of Computer Calisthenics and Orthodontia ("Running Light without Overbyte") is now on your local stand as Software Tools. (But for the old-timers, they still call it Dr. Dobbs Journal of Software Tools in the finer print.)

I **don't** say a lot of guys from the old days have lost their integrity by making a big business of it. But I keep my integrity by pointing out that the **real** problems are right where they always were.

Millions of people have learned to live with not-very-nice machines and awful operating systems—what we may call the Industrial Desktop. Indeed, PC clones are being sold by Amway.

Not only that, but the PC clone has been picked as the main desktop computer for use in the Soviet Union, while (an

ocean away) the U.S. Army has designated the PC clone as its principal battle computer—supposedly to keep track of current strength, with "standard commercial word processing, spreadsheet and database software."

We are reconciled to a SoftBiz whose incompatibilities are like those of cars and cameras—but worse; where programs are incompatible not just between different hardware lines, but incompatible for cooperation on the same computer.

In hardware, now, incredible things are happening. The Atari 520, a serious and fast computer (but not a standard one) can be bought for as little as **three hundred dollars** in a serious configuration. The new Mac II can address 4,294,967,304 bytes (32 gigabytes). So if CPU hardware were all that mattered we'd be in great shape.

But in software things are pretty bad. And the general level of understanding is pretty low. (See "A Field of Rubble," p. 20). People are talking Fifth Generation when they can't even absorb UNIX (see p. 104). "Power Users" can accomplish wonders, but only by spending much of their lives messing with the system.

At this writing (first quarter '87) there are some 4 million Apple IIs, 1 million Macintoshes, and maybe ten million PCs,

The media talk as if there were no others but Jobs'n'Woz, who just happened to be standing underneath when the idea of personal computers dropped out of a clear blue sky.

Those of us who were fighting for personal computers in a previous decade can be forgiven our annoyance.

Likewise the guys before Jobs'n'Woz who tried to sell personal computers but didn't pull it off quite as well. Especially MITS, Sphere, SWTP and Processor Tech, and Truong Trong Thi, who actually announced the first personal computer for sale (see Steve Ditlea, ed., The Digital Deli, Workman Press).

Hurray for Apple's success and its good products. But the media should lay off rewriting history.



around half being from IBM. As a guess, probably a quarter or less of these desktop computers are in homes.

That's pretty impressive unless you put it in its larger context: there are something like twenty times more VCRs in the United States than computers. For good reason. **Computers are not yet easy, and for most people they are no longer fun.** Millions of small computers are out there in homes and offices, but most subsist in a private gridlock of complex file maintenance, tangled applications, abandoned hopes. It is a mess, a snarl and a tragedy.

## BIBLIOGRAPHY

"Those Unforgettable Next Two Years."

Speech by Ted Nelson at the Spring 1977 West Coast Computer Faire. Appears in Ted Nelson, *The Home Computer Revolution, 1977* (published by the author).

Steven Levy, *Hackers: Heroes of the Computer Revolution*. Anchor/Doubleday.

This details the intimate relationship between the frontier computer crazies and the key developments of the field.

Ernest M. Rogers and Judith K. Larsen, *Silicon Valley Fever: Growth of High-Technology Culture*. Basic Books, 1984.

Another history.

Paul Freiberger and Michael Swaine, *Fire in the Valley: The Making of the Personal Computer*. Osborne/McGraw-Hill, 1984.

Michael Moritz, *The Little Kingdom: The Private Story of Apple Computer*. William Morrow and Company, 1984. *The story of Apple.*

Scott Cohen, *Zap!: The Rise and Fall of Atari*. McGraw-Hill, 1984.

Michael S. Tomczyk, *The Home Computer Wars: An Insider's Account of Commodore and Jack Tramiel*. Compute! Publications, Inc., 1984.

Benn Ross Schneider, Jr., *Travels in Computerland*. Addison-Wesley.

*The computer world before the Altair*. Thomas A. Ross, *The Eudaemonic Pie*. Mifflin.

*True story of hackers versus Vegas.*

Adam Osborne, *Running Wild: The Next Industrial Revolution*. Osborne/McGraw-Hill, 1979.

## Interactive Goods

At lunch  
I heard  
salesmen  
discussing  
"interactive  
goods."  
It's come  
to that.

## THE REVOLUTION BETRAYED

After every big revolution—American, French, Russian, Chinese—there are those who are disappointed, who were expecting a deeper social revolution, who hate to see it become stodgy and middle-class. "We had something beautiful going, what happened to it?"

These guys then turn around and preach another revolution. Trotsky. Mao. Tom Paine. Even Tom Jefferson.

That's how I feel about the personal computer revolution. I meet guys from the early days and we shake our heads and say, "It isn't fun any more."

I saw personal computing become industrialized, lose its ideals, and turn to crap—incompatible products for the Industrial Desktop. But the possibilities are lurking still: under the dingy crust of complication and incompatibility there pulses the glowing heart of a better world to come.

I spoke a few pages above about the conceptual excitement, fun and power of computers in general, and the world of computer ideas.

In grim contrast, unfortunately, is the unnecessary complication of day-to-day work and the routine danger to your files, your work and your organization:

# A FIELD OF RUBBLE

I run into people from years ago, and they say with amazement:

"I would never have believed it! Personal computers everywhere! So this is what Ted Nelson was talking about twenty-five years ago!"

No, friend, it is not. It is very much the opposite.

I was preaching the simplification of life, and a new world of unity and clarity and understanding. Instead, today's computer world is an abhorrent tangle, a labyrinth, a city of chaos with many blighted suburbs; a twisted landscape of rubble, junk and insane complication; messy, Balkanized and tragic.

It is all snaggles and catches, a gridlock of mutual obstruction, a catechism of memorizations, a continuing emergency of irrelevant complications to which your close attention is vital. To make things work you have to know the equivalent of all the connections on the New York subway system.

Computerization is still *invasive*, not gently adapting to methods that are already in place, but hacking them to fit the convenience of today's programming environments the way a hog is made to fit a deepfreeze.

And so much still **just doesn't work**, or has drawbacks and exceptions so glaring as to get in the way of any sensible use.

And as far as I know, **no one's life has yet been simplified by a computer.**

(If you do not yet know the terms used below, you should nevertheless understand the general drift without much difficulty.)

## INCONSISTENCY

Things still are typically incompatible. The parts do not fit together.

Your work is not cumulative. Often when you change programs you have to discard all your previous working methods and work out new ones.

Almost nothing is easy the way it should be; only a few things fit together.

The frontier is still, as I said in 1974, the issue of simplicity and clarity. And all too few clarifying ideas have been tried yet.

Until now only the top 10%—in patience and technological smarts—could use computers. The comparatively vivid Macintosh ("the first computer worth criticizing"—Alan Kay) extended this proportion perhaps 10% more.

**The computer field is like a great big toy store.**

**But all the toys are broken.**

—Steve Witham

be.....  
market  
continues  
to  
support  
chaos.

Aaron  
Marcus

COMPUTERS ARE JUST  
AS OPPRESSIVE AS BEFORE,

BUT SMALLER AND  
CHEAPER AND MORE  
WIDESPREAD.

NOW YOU CAN BE  
OPPRESSED BY COMPUTERS  
IN YOUR LIVING ROOM.



Beginners are all too often hit with complexities they shouldn't have to face, perhaps ever. Not only do the things you deal with, such as the MS-DOS operating system, have glaring inconsistencies; but every program has its own set of rules, and you can't **customize or bring them under a common regime**. It is like having to deal with a separate bureaucracy for each.

Different lines of computers are, of course, incompatible. It's as if there were six different kinds of roads and six different kinds of cars for them, and you had to keep watching the signs to know whether to drive on the right or the left, and having to plot out the overpasses like the secret service.

This is not to say the machines aren't wonderful, considered in isolation. But try to bring them together or feed them programs, and towering complications ensue.

And installing computers is a complicated mess.

Keeping your files and their interrelationships straight is still an enormous task

Users must deal with grim tricky control programs—operating systems like CP/M and MS/DOS—which make it all too easy to wipe out their files or bring about other dreaded conditions.

Figuring out how to map the things you want to save into the inane and bizarre hierarchical structures demanded by today's operating systems takes a long time, is highly inappropriate for most purposes, and leaves your disks **strewn with debris**—mutilated bodies of information gasping in the dust. I would venture to say that the computer files of many if not most users and most offices are an unholy mess. Bernice or Sam knows where things are, but when they leave it's all effectively destroyed. Whoever is not extremely diligent or extremely simpleminded has this problem. The process of evolving your own file structure may so swamp you with leftover chunks that you'll never be able to get that same material in order again.

Among the filing problems: lots of different things have similar (or the same) names. You have to put things in a hierarchy, which for most purposes is not sensible. People disagree about how to organize the files, and fight about it. You run out of physical places nearby to put the disks, and gradually the farther-away disks become forgotten.

And then there's the backup problem. (For many people, the backup problem is the guilty secret that they haven't been backing up.)

And the other programs (the "application software") are worse. The beginner can do almost nothing. Software is generally incompatible. (Programs that connect are usually from the same manufacturer, and even those typically connect badly.) Programs do not fit together, their aspects cannot be combined, and their results typically cannot go back and forth between them. Rather than being easy-to-get-into and self-revealing, the typical program is a grotesque labyrinth of complication, assuming (but not explaining) a complex set of ideas. Learning each new program is typically a big investment of time, requiring weeks of effort to get the first small nugget back.

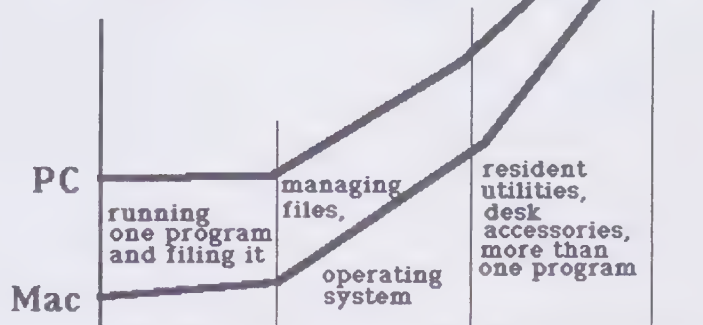
Existing software still is not for casual users, those who just want to do something quickly, let alone for people who are frantic. Unlike paper, computers cannot be safely used yet by people who are

**People think computers are rigid and invariant. This (as stated elsewhere in this book) is due to the systems which people have imposed, and then blamed, on the computer.**

The technoids, Lords of Complication, do not see or understand the problems other people have with the systems they design. This is the problem.

The real problems of the computer field are design problems.

## ESCALATING DIFFICULTIES

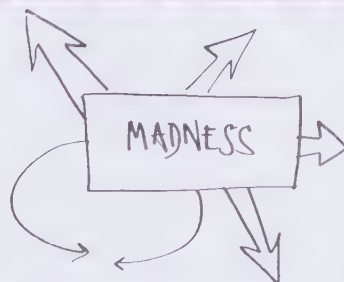


I hope that this book will help people who are inconvenienced by computer systems to understand and pinpoint what they think is wrong with the systems—in their data structure, interactive properties, or other design features—and that they will try to express their discontents intelligently and constructively to those responsible. Including, where appropriate, International Business Machines Corporation, Armonk, NY.



The Sinclair 1000 was advertised with the slogan:

"Why spend thousands to find out you hate computing when you can find out for \$49.95?"



**The two main worlds present a study in contrasts.**

**The PC is authoritarian, the Macintosh is bumpkinly and obtuse.**

**Using the PC may be compared to juggling with straight razors. Using the Macintosh is like shaving with a bowling pin.**

There are many fine things about the Macintosh, and I generally recommend it to beginners.

However, its simplicity is only skin deep.

I had to give a talk at Vassar College last year, and I got inspired and stayed up late writing a new outline on my son's Macintosh, using MacDraw, my favorite outlining product.

The next day, an hour before my talk, I took my diskette to the office at Vassar and plunked it in their Macintosh, and we ordered it to print out my outline on the laser printer.

Lo! My headline font had been changed from "New York" to a silly Computerish Font (see p. 148); and all the actual text had turned into straight lines. Not letters at all, just lines on the page.

The solution: we carried the Macintosh across the campus and strapped it to the lectern. I had to read from it with constant mousing. And most ignominious of all, I had to read it in that idiotic font.

The font problem is just one of the nasties behind that jolly interface.



The  
good  
news:  
ANYTHING  
IS  
POSSIBLE  
on your  
computer.  
The bad  
news:  
NOTHING  
IS EASY.

"...It is very easy to be blinded to the essential uselessness of them by the sense of achievement you get from getting them to work at all."

Douglas Adams  
So Long, and Thanks  
for All the Fish,  
Harmony Books, 1985,  
p. 177

As a rule, each piece of software creates a neat little world inside which things are orderly—and pushes out the disorder, pushes it outside.

Someday someone will come up with large-scale programs that don't have an "outside" to push problems into. That will be the day.

flustered, frantic or stoned. Whereas (regrettably though the necessity may be), cars can.

The parts of the overpriced software package cannot be mixed and matched; they must be taken whole, replacing whatever else you were using—something that might have been better in some ways.

Installing programs is a complicated mess.

But the superficial complication of software hides its deeper wrongfulness.

There are rigid and artificial dividing lines between "separate applications" that have nothing to do with the real connections of the matters in your mind. These separate applications were fostered, historically, by many things—especially file structures. The conventional types of programs—"word processing," spreadsheet, database—are misguided. (See T. Nelson, "The Tyranny of the File," Datamation, 15 December 1987.)

Oh, it is possible in principle to do some mixing and matching. By keystroke macros, for instance, elaborately worked out for programs like ProKey, you can change the controls of other programs. But this is a full-time job.

It's a mix n' match new world. The time and trouble for mixing and matching, however, can be very great. It is a tangle of setups and desktop accessories, pop-up and resident programs and keyboard macros, of programs that can't work together, of partial compatibilities and (more commonly) total incompatibilities; of software you cannot customize. It is a jumble of programmer tools and incompatible, narrow, special-purpose programs. Everything has a different conceptual structure.

So-called "integrated software" was intended to solve this problem, but it doesn't; it just produces local tangles of a new kind of complication.

Even the Macintosh, ballyhooed as "the computer for the rest of us," is only

superficially simple. To use it well involves just as much complication as the PC. (They say it's easier on the Macintosh, but not if you want to do anything serious.)

Setting up a computer the way you want it is like customizing a car: it can take all your evenings for two years. And after two years you want something else anyway.

Keeping your computer system together, hardware and software, is a full-time job; and that has to stop. Unless you buy the standard store packages, it's still all these damn pieces a tekkie has to tie together. And tying together all these wonderful thingsamabob is a big job. Now, what you buy as a package may be moderately easy to set up, may only need a tekkie by your side. But what is "theoretically" possible is usually difficult, and could take a salaried tekkie several months.

Setting up your own system in software can be incredibly dangerous in subtle ways. You may be making commitments for a lifetime. Your tekkie friends can really help you or can really screw you up, even though they intend to do just what you want—or what they think you REALLY want. The more you know about computers the better: for you cannot trust anyone else to know what you want. It's that messy.

And it all can be easy, it CAN! (Video games point the way.)

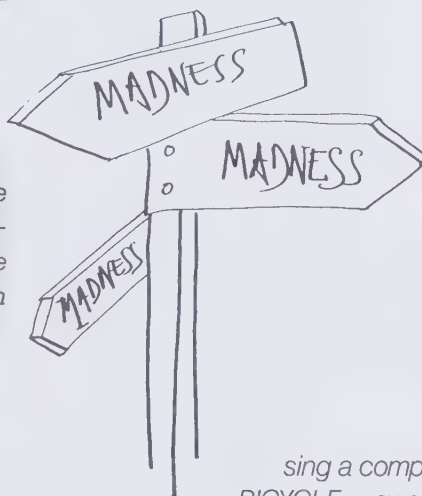
The pundits and the companies have made it clear that they don't understand interaction. (Apple, better at interaction than most, has had to throw an immense amount of weight behind its trivial Macintosh interface, as if it were a fabulous and ultimate achievement.)

The levels of animated presentation in video games—let alone those of the high-power graphic displays and simulators—have not reached the office market; office systems are static and hardly interactive at all. (Today's "business graphics"—pie charts and histograms—are absurdly shallow.)

The reaction of indignation and horror that many have to the complexity of computers is **entirely appropriate**. So many people have been messed up by computers as they are presently being foisted.

## FREEDOM

I have been astounded to learn how little concern there is with ideals at all in the computer field; and for the ideal of freedom, a matter which is always close at hand, almost none.



People talk as if "setting up a computer" is like buying a car. No, it's like buying a car, building garages and driveways and grease pits, hiring chauffeurs and mechanics, learning to drive, and learning to pick up handkerchieves with your teeth as you do wheelies in a parking lot.

sing a computer SHOULD BE LIKE RIDING A BICYCLE—swerve, spin, loop, bank, zip around on impulse. INSTEAD, IT'S LIKE RIDING A BICYCLE ON A TIGHTROPE.

And, having to learn computing on the IBM personal computer is like having to learn to drive in a shift car. In the Himalayas.



WE ARE  
TRAINED  
IN AN  
UNFIT  
FITNESS

What is called "computer literacy" generally means facility with (and acquiescence to) today's hideous operating systems—and repudiating any earlier, intuitive notions of how computers work. Intuitive notions which probably made more sense than how things are, and which are what we will get to in future computer systems at last.



We see in the new interactive systems—the hypermedia themselves—new potentials for imprisonment, if the user is not given absolute option to escape, to seek a different view.

New and exclusive prisons—closed interactive systems—are everywhere.

### TURNING US TO SHEEP

People are still being told this is how things have to be, **AND THEY ARE STILL BELIEVING IT.**

Much of the computer field is a foolish rabble, surging this way and that, unable to imagine anything they have not yet seen.

The psychology of the consumer: **UTTER FAILURE TO GRASP HOW CONFOUNDED THINGS ARE**—people are afraid to express the naive hopes of compatibilities and convertibilities that make the most sense.

If you don't know the program is arbitrary, you believe that's how it has to be. Turning us to sheep.

What bothers me is the moral twerpitude of so many people in the field, looking only at what is, not imagining what **COULD BE.**

Many of today's generation of software watchers are extremely shallow; their minds corrupted by what they've seen; they don't realize what can be.

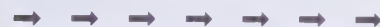
**COULDN'T IT BE DIFFERENT?**

**COULDN'T IT BE BETTER?**

Yes.

You're on the right track.

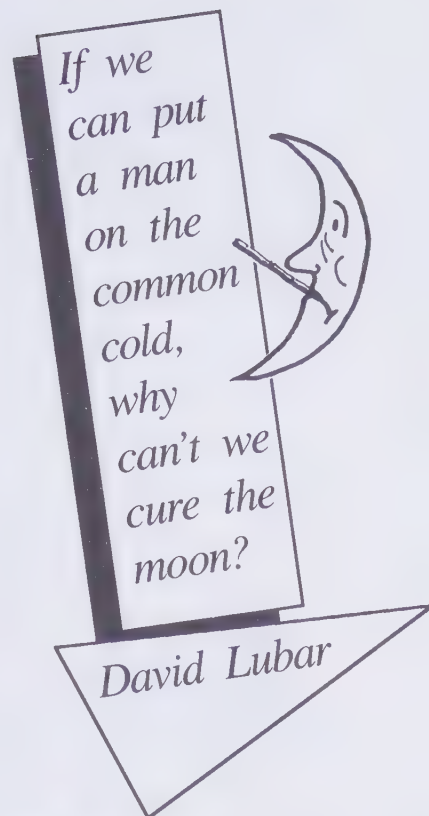
Today's sheep have lost their imaginations, lost the capacity to dream, and think what they can see is how it has to be.



**Nothing has been right in the computer field, nothing has been cumulative.**

**People have to keep changing systems, converting files, moving along to other machines, other software, trying to set up something stable.**

**And spending far more money than should be necessary, on machine after machine, software package after software package.**



### THE DECLINE OF MIND

ARE YOU AWARE THAT OUR LEVEL OF LITERACY IS CONTINUING TO DECLINE?

I DUNNO... I THINK I SAW SUMP' N ON TV ABOUT IT.

### KEEPING THE DREAM

**“As great visions move out, all the best parts get stripped away.”**

—Alan Kay

## HOPE FOR THE FIELD

You can get personal computers now with virtually unlimited crunch, unlimited memory. Yet the same old clumsy ways of thinking and working continue to prevail. **No one's life has yet been simplified by a computer;** and the unification of literature, visualization, education, art and science that the personal computer seemed to promise still seem as far away as ever.

It is not the hardware holding us back. It is **ideas**, the old ones; it is **design**, tired designs that were appropriate when machines were small and expensive and unreliable and unconnected.

Everybody is taking a subset of the problem, and some people are doing a very good job, sophisticated and diligent, with the subsets they have chosen. Realists have built many fine, interlocking products.

There is just one thing wrong with that: and that is that the most important problems cannot be done piecemeal. Flying to the moon is an example. Such problems can be broken into chunks, yes, but the chunks are not obvious ones, nor can they be solved independently. **YOU'VE GOT TO EAT THE WHOLE BOWLING BALL.**

Unfortunately, people who have solved a problem in some way they chose to narrow down the field don't want to hear about the **real** problem.

The real problems of manipulating information are right where they always were. They have to do with conceptual structure, interconnectedness, mental clarity and presentation; and they are not technical, but conceptual and artistic. The clumsy fads of today's software are only gestures at them. The problems are **overall design.**

But there will be revolutionary changes in the next generation of interactive system, coming from the next generation of designers—the filmmakers and video-game veterans and philosophers. It is time for a new world of designs, a new system of thinking, and a new level of visualization and conceptual artistry worthy of today's equipment and tomorrow's possibilities and hopes. ■



■ We must leave paper behind at last if we are to move to the new worlds of tomorrow. And that includes leaving behind such last-ditch anachronisms as "Wiz-zywig" (see p. DM 28) and "modelessness." These all have their eye on the past.

It is manifest destiny. Computers will be everywhere, better and different. We will be reading and writing principally at screens. There will be crib computers for infants, providing multidimensional sight-and-sound interaction; **wearable** computers (similar to the Sony Walkman) with a keyboard on your knee; new multi-function devices that combine dictation, photography and digital record-keeping—dictacams?; new meeting rooms and conversation pits with aerial blackboards we can share; a new structure of

documents that more truly represent our ideas, in electronic libraries spanning the world (see p. 141).

The local tangles and snafus so familiar now will disappear; it will be a simplified, unified and clarified world of information and ideas. We will have forgotten today's sharp divisions between "applications." The global organization and maintenance of all your stored materials will be automatic. Such issues as "backup" and "maintaining database integrity" will have become resolved into different issues with new solutions.

The irony, of course, is that the naive expectations, the beginners' hopes, were right all along. Just what everybody wanted **will** become available: the easy-to-use, organize-your-life-or-company, top-to-bottom system: being able freely to combine what are now "separate applica-

tions," being able to create with ease a sweeping personal environment, "keep track of everything," "have all the answers." The all-enveloping vision of a changed and better life—that vision that was vaporized by computer literacy classes—will come true. Today's "computer literacy" and styles of operation will seem to us then like poking the cat's whisker on the crystal in the radio set of 1920. "It's what we had to do **then**," we'll say.

The possibilities for individuals' lives, creativity and intellect will change the world. There will be a total and revolutionary change in daily human life. There will be simplification of personal and business life, and easy access to a world of ideas; **real** education, sweeping and generalized understandings, access to knowl-

edge on every subject; the opportunity for more honest negotiations and subtle, complex solutions involving many parties.

The only ones who can be opposed to this vision are the information monopolists, those who benefit from others' ignorance. But that's a lot of people. (There even are those who consciously wish to entrap us in a world of **less** access to information than we now have; see p. 167.) Be alert to, be amused by, the many euphemisms and pretexts and excuses they will find.

The eternal verities stand tall. The problems are what they always were: freedom, pluralism, access. In all of this there will emerge a new humanism; in all of this we will find the new strengths and powers that we deserve, and freedoms that we require.

A lot of people, especially the press, think "ease of use" and "computer power" are opposites. Usually this is because they've seen the Macintosh, which appears to beginners to be easy to use, and the PC, which is complex but allows more control to the "power user."

But these are just stereotypes.

It is my belief that the computer system of tomorrow can be ten times simpler, ten times more powerful, and a hundred times clearer and more vivid.

When the real Office of the Future appears, it will be as easy as Pac-Man.

*The storm of computer progress moves continually on, leaving behind it many stagnant puddles containing interesting life. Every stage, every invention, lives on someplace, a leftover setup or way of thinking that people live and work in. But the storm center moves on. Tomorrow's center will be very different.*

*So the residue of today's clumsy, screwed-up computer world will continue, in some form, indefinitely; but I believe most of it can arise from the current shambles.*

● The  
● next  
● generation  
● of  
● computers  
● should be,  
● like—for  
● our  
● parents. "  
● Artie Manley,  
● Electronic Arts

## The Buck Stops Here

Everywhere in the world people can pretend that your ignorance, or position, or credentials, or poverty, or general unworthiness, are the reasons you are being pushed around or made to feel small. And because you can't tell, you have to take it.

And of course we can do the same thing with computers. Yes, we can do it in spades. (See Cybercrud, p. 27.) But many of us do not want to. There has to be a better way. There has to be a better world.

## Logic

People cannot imagine greater ease of use until they experience it. Doug Engelbart has the cleverest way to explain the concept of ease of use: he ties a pencil to a brick, and asks you to write with it.

"Now imagine," says Engelbart, "that you untie the pencil from the brick. That is how much improvement you will get from the new tools." (See p. DM 16.)

At a conference in March 1987, Alan Kay showed a videotape of a 22-month-old girl using a Macintosh at home. She had no trouble using the mouse to open files and play with MacPaint, a drawing program.

Kay had worked at PARC for years on the design of computers for kids. But, he explained ruefully, it never occurred to them that such programs could be used by children that young, so they had never tried it.

Tomorrow's world will involve many new environments and decision options. The technoid's dream: each will be complex, learned individually, manned by expert-bureaucrats. The Jeffersonian dream: they will be clear and simple, and unified as one, and we will be able to move among them with ease. The casual user will be able to step up to most systems and begin using them as easily as she now uses a vending machine.

"Look and feel" is a big issue right now, and the two most popular look-and-feel paradigms are the Macintosh (or Maclike) environment, with windowshade menus, and the Lotus paradigm, with a stepping cursor and a help line.

But ALL THE LOOKS AND FEELS HAVE NOT BEEN SEEN AND FELT.



# WHERE IT'S AT

Computers are where it's at.

Recently a bank employee was accused of embezzling a million and a half dollars by clever computer programming. His programs shifted funds from hundreds of people's accounts to his own, but apparently kept things looking innocent by clever programming tricks. According to the papers, the program kept up appearances by redepositing the stolen amount in each account just as interest payments were about to be calculated, then withdrawing it again just after. ("Chief Teller Is Accused of Theft of \$1.5 Million at a Bank Here." *New York Times*, 23 March '73, p. 1.) The alleged embezzlement was discovered, not by bank audit, but by records found on the premises of a raided bookmaker.

In a recent scandal that has rocked the insurance world, an insurance company appears to have generated thousands of fictitious customers and accounts by computer, then bilked other insurance companies—those who re-insured the original fictitious policies—by fictitious claims on the fictitious misfortunes of the fictitious policy-holders.

In April of 1973, according to the Chicago radio, a burglary ring had a "computerized" list of a thousand prospective victims.

There have been instances where dishonest university students, nevertheless able programmers, were able to change their course grades, stored on a central university computer.

It is not unheard of for ace programmers to create grand incomprehensible systems that run whole companies, systems they can personally play like a piano, and then blackmail their firms.

A friend of a friend of the author is an ace programmer at the Pentagon, supposedly a private supervising colonel. On days he is mad at his boss, he says, the army cannot find out its strength within 300,000 men. Or three million if he so chooses.

This awkward state of affairs, obviously spanning both the American continent and most realms of endeavor, has come about for various reasons.

First, the climate of uncomprehension leads men in management to treat computer matters as "mere technicalities"—

a myth as sinister as the public notion that computers are "scientific"—and abandon the kind of scrutiny they sensibly apply to any other company activities.

Second, most of today's computer systems are inherently leaky and insecure—and likely to stay that way awhile. Getting things to work on them involves giving people extraordinary and invisible powers. (Eventually this will change, but watch out for the meantime.)

The obvious consequence is simply for the computer people to be allowed to take over altogether. It may indeed be that computer people—the more well-informed and visionary ones, anyway—can see the farthest, and appreciate most deeply the better ways things can go, and the steps that have to be taken to get there. (And Boards of Managers can at least be partially assured that hanky-panky at the lower levels will be prevented, if men in charge know where the bodies are buried.)

That seems to be how it's going. Examples:

The president of Dartmouth College, John Kemeny, is a respected computer-man and a developer of one of the important computing languages, BASIC (see p. 59).

## WHERE ARE THEY NOW?

*John Kemeny served on the Three Mile Island inquest committee, and has more recently brought out another version of BASIC, called True BASIC (see p. 59).*

The new president of the Russell Sage Foundation, Hugh Cline, used to teach computing at Columbia.

It's probably the same in industry. In other words, more and more, for better and for worse, things are being run by people who know how to use computers, and this trend is probably irreversible.

In some ways, of course, this is a sinister portent. In private industry it's not so bad, since the danger is more of embezzlement and botch-up than of public menace. But then there's the problem of the

## CHUTZPAH DEPARTMENT

A college student broke through the security of the Pacific Telephone computer system from a terminal and, according to *Computerworld* (6 June '73), stole over a million dollars worth of equipment by ordering it delivered to him! (*Penthouse*, December '73, claims he was in high school and it was only nine hundred thousand, but you get the idea.)

After serving a few weeks in jail, he has formed his own computer-security consulting company.

More power to him.

The new breed has got to be watched.

This is the urgency of this book. Remember that the man who writes the payroll program can write himself some pretty amazing checks—perhaps to be mailed out to Switzerland, next year.

From here on it's computer politics, computer dirty tricks, computer wonderlands, computer everything.

For anyone concerned to be where it's at, then, this book will provide a few suggestions. Now is the time you either know or you don't.

Enough power talk. Knowledge is power. Here you go. Dig in.

## IT'S ALL COMPUTERS

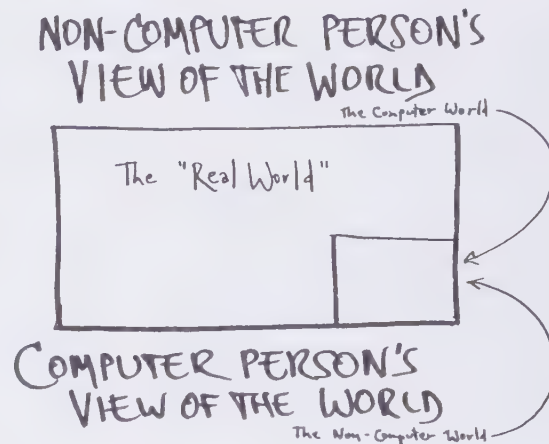
Somewhere inside almost every object, these days, is a little processor. Elevators, drink machines, gas pumps, the jammed cash register that just happens to be holding up your line—all have peculiarities based on how some programmer thought about the problem.

Quite recently, as I was awaiting takeoff in a recently built airplane, there came a disquieting announcement from the Captain. There would be a delay, he said, "while we get center of gravity figures"—the computer's acting up.

This did not make anyone feel secure.

Quite the old joke about the takeoff announcement in the fully automated plane, where "Nothing can possibly go wrong. Go wrong. Go wrong."

There is considerable agreement between the two views:



Many Humanists avoid the computer field. Over the last two decades, I've had a dozen acquaintances who told me they wanted no part of computers, then (several years later) ruefully accosted me to say they've gotten into it.

I had a student who showed talent as a designer of interactive systems, but he wanted to go into ornithology instead.

I recently heard, however, that he's now programming bird calls.



government. The men who manage the information tools are more and more in charge of government, too. And if we can have a Watergate *without* computers, just wait. (See "Great Issues," p. 157.)

The way to defend ourselves against computer people is to become computer people ourselves. Which of course is the point. We must all become computer people, at least to the extent that we have already become Automobile People and Camera People—that is, informed enough to tell when one goes by or when someone points one at you.

## MANY MANSIONS

The future is going to be full of computers, for good or ill. Many computer systems are being prepared by a variety of lunatics, idealists and dreamers, as well as profit-hungry companies and unimaginative clods, all for the benefit of mankind. Which ones will work and which ones we will like is another matter. The grand and dreamy ones bid fair to reorganize drastically the lives of mankind.

(These grander and dreamier applications are discussed on the other side of this book.)

But the plain computer visions are grand enough.

The great world of time-sharing, for instance. ("Time-sharing" means that the computer's time is shared by a variety of users simultaneously. See p. 103.) If you have an account on a time-sharing computer, you can sign on from your terminal (see p. DM 11) over any telephone, no matter where you are, and at once do anything that particular computer allows—calling up programs in a variety of computer languages, dipping into data on a variety of subjects as easily as one now consults a chart.

For instance, at Dartmouth College—where time-sharing is perhaps farthest advanced as a way of life—the user (any Dartmouth student, for instance) can just sit down at a terminal and write a simple program (in Dartmouth's BASIC language, for instance) to analyze census data. Since Dartmouth has a complete file on its time-sharing system of the detailed

sample from the 1970 census, the program can buzz through that and report almost immediately the numbers of divorced Aleuts or boy millionaires in the sample, or (more significantly) the relative incomes of different ethnic groups when categorized according to the questioner's interests.

But simple time-sharing is only the beginning. *Networks* of computers are now coming into being. Most significant of these is the ARPANET (financed by ARPA, the Defense Department's Advanced Research Projects Agency, it is nonetheless non-military in character). Dozens of large time-sharing computers around the country are being tied into the Arpanet, and a user of any of these can reach directly into the *other* computers of the network—using their programs, data or other facilities. Arpanet enthusiasts see this as the wave of the future.

## MINI MANSIONS

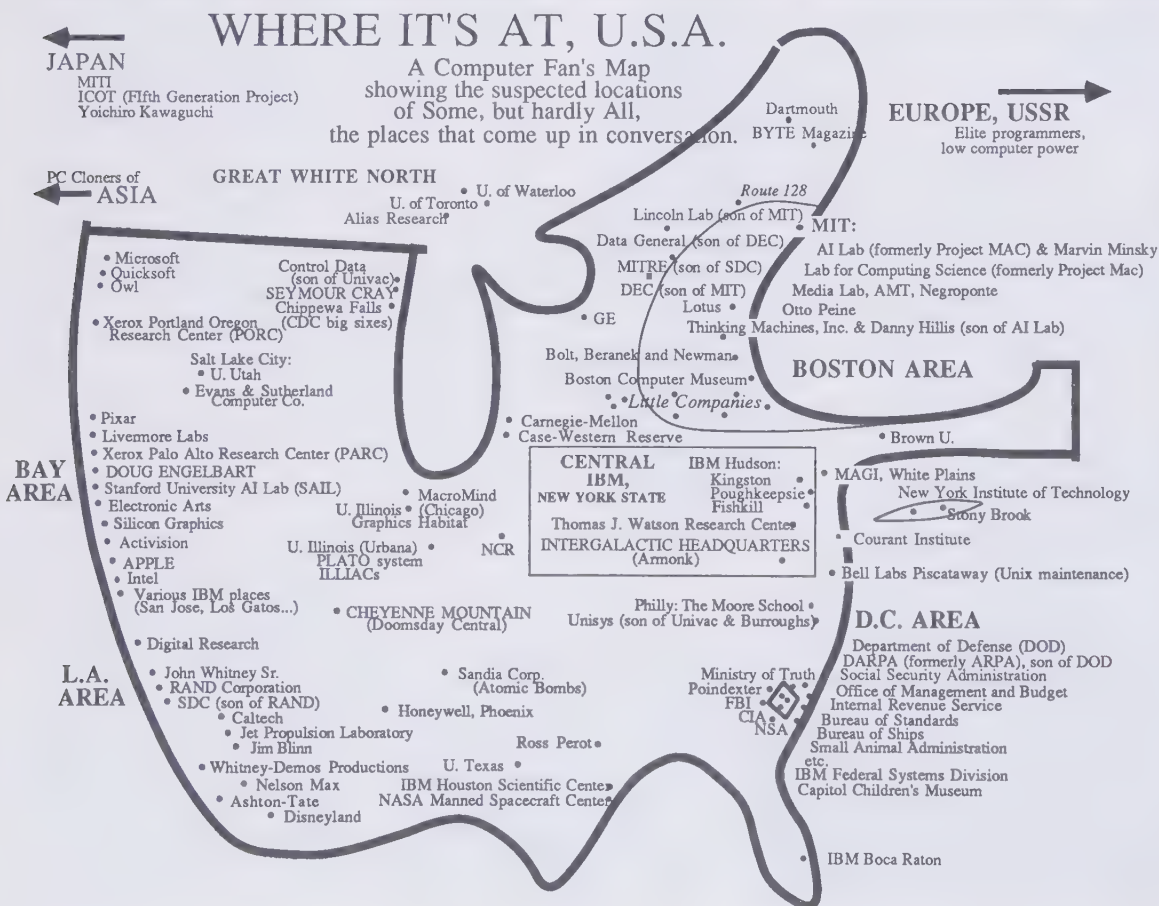
But while computers and their combinations grow bigger and bigger, they

also grow smaller and smaller. A complete computer the size of an Oreo™ cookie is now available, guaranteed for twenty-five years (and very expensive). But its actual heart, the Intel microprocessor, is only sixty bucks *now*, and just wait (see "Microprocessors," p. 96.) By 1980 there should be as many programmed and programmable objects in your house as you now have TVs, radios and typewriters; that's a conservative estimate. But just *what* these devices will all be doing—ah, there's the question that has many people talking to themselves.

## OTHER COMING THINGS?

There are a lot of tall stories about what computers will do for the world. Among the most threatening, I think, are glowing reports of "scientific" politics (don't you believe it). We hear how computers will bring "science" to government, helping, for example, to redraw the lines of election districts. (See "Cybercrud," p. 27.)

Then you may also have heard that computers are going to be our new mentors and companions, tutoring us, chatting with us and perhaps lulling us to sleep—like Hal in "2001." Worried? Good. (See "The God-Builders," p. DM 120.)



*That sinking Feeling*  
as you begin to realize just  
how big the computer field  
really is, can be bested by

*THAT SWIMMING FEELING,*  
as you learn how computers  
work in the area that interests  
you.

**Y**our fate and freedom as a working individual are tied to computers, either through an occupation where you can choose and affect their use, or where that use is chosen for you.

So is your fate and freedom as a political animal—now herded and branded and carefully supervised, but still able—at least for now—to break free. See pp. 164-170.



# CYBERCRUD

A number of people have gotten mad at me for coining the term "cybercrud," which I define as "putting things over on people using computers." But as long as it goes on we'll need the word. At every corner of our society, people are issuing pronouncements and making other people do things and saying it's *because of the computer*. The function of cybercrud is thus to confuse, intimidate or pressure. We have all got to get wise to this if it is going to be curtailed.

Cybercrud takes numerous forms. All of them, however, share the patina of "science" that computers have for the layman.

## 1a) COMPUTER AS MAGIC WORD

The most delicate, and seemingly innocent, technique is the practice of naming things so as spuriously to suggest that they involve computers. Thus there is a manufacturer of pot-pipes with "Data" in its name, and apparently a pornography house with "Cyber-".

## 1b) COMPUTER AS MAGIC INGREDIENT

The above seems silly, but it is no less silly than talking about "computer predictions" and "computer studies" of things. *The mere fact that a computer is involved in something has no bearing on its character or validity.* The way things are done with computers affects their character and validity, just like the way things are done without computers. (Indeed, merely using a computer often has no bearing on the way things are done.)

This same technique is easily magnified to suggest, not merely that something involves computers, but is wholly done by computers. The word "computerize" performs this fatal function. When used specifically, as in *computerize the billing operation*, it can be fairly clear; but make it vague, as in *computerize the office*, and it can mean anything.

"Fully computerize" is worse. Thus we hear about a "fully computerized" print shop, which turns out to be one whose computers do the typesetting; but they could also run the presses, pay the bills and work the coffee machine. For practical purposes, there is no such thing

as "fully" computerized. There is always one more thing computers could do.

## 2) WHITE LIES: THE COMPUTER MADE ME DO IT

Next come all the leetle white lies about how such-and-such is the computer's fault and not your decision. Thus the computer is made a General Scapegoat at the same time it's covering up for what somebody wants to do anyway.

"It has to be this way."

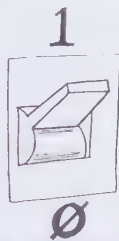
"There's nothing we can do; this is all handled by computer."

"The computer will not allow this."

"The computer won't let us."

The translation is, of course, **THE STINKY LOUSY PROGRAM DOES NOT PERMIT IT.** Which means in turn: **WE DO NOT CHOOSE TO PROVIDE, IN OUR PROGRAMS AND EQUIPMENT, ANY ALTERNATIVES.**

Now, it is often the case that good and sufficient reason exists for the way things are done. But it is also often the case that companies and the public are inconvenienced, or worse, by decisions the computer people make and then hide with their claim of technical necessity.



*They have really stupid switches now on computers that look Scientific. Zero means Off, see, because that's no current, and 1 means On, because it's, uh, a number, right?*

*Actually, this was just a compromise so they wouldn't have to print switch-plates in different languages. But it sure is stupid. Zero could also mean On. These codes are arbitrary.*

## "COMPUTERS AND THEIR PRIESTS"

"First get it through your head that computers are big, expensive, fast, dumb adding-machine-typewriters. Then realize that most of the computer technicians that you're likely to meet or hire are complicators, not simplifiers. They're trying to make it look tough. Not easy. They're building a mystique, a priesthood, their own mumbo-jumbo ritual to keep you from knowing what they—and you—are doing."

—Robert Townsend,  
*Up The Organization* (Knopf), p. 36.

In the movie "Fail-Safe," they showed you lots of fake tape drives with the reels constantly turning in one direction. This they called a "computer." Calling any sinister box "a computer" is a widespread trick. Gives people the willies. Keeps 'em in line.

## Balderdash!

Dear Charter Subscriber,  
Because AVANT-GARDE's subscription roll is maintained by electronic computer, it is necessary for us to assign a common expiration date to all subscriptions. This enables us to distribute copies and mail renewal notices to all subscribers at the same time. Therefore, we are writing to inform you that your AVANT-GARDE Charter Subscription must be renewed now.

## A BASIC REJOINER

**we should all practice and have ready at the tip of our tongues: WHY THE HELL NOT? YOU'RE THE ONES WITH THE COMPUTERS, NOT ME!**

**Let's froth up a little citizen indignation here.**

*Except maybe now that we do have computers, we can find an even better way (and fair way) to get back at inconsiderate companies.*

## REASONS FOR CYBERCRUD (ALL BAD)

- 1) to manipulate situations.
- 2) to control others.
- 3) to fool.
- 4) to look like hot stuff.
- 5) to keep outsiders from seeing through something.
- 6) to sell something.
- 7) to put someone down.
- 8) to conceal.
- 9) general secretiveness.
- 10) low expectation of others' mentality.
- 11) seeking to be the broker and middleman for all relations with the computer.
- 12) vagueness sounds profound.
- 13) you don't have to show what you're not sure of.
- 14) your public image is monolithic.
- 15) you really don't know.



### 3) YAGOTTAS: COMPUTER AS COERCER

More aggressively, cybercrud is a technique for making people do what you want. "The computer requires it," you say, and so people can be made to hand over personal information, secretaries can be intimidated into scouring the files, payment schedules can be artificially enforced.

### THE GENERAL STATUS TRICK

Status tricks, combining the putdown and the self-boost, date back to times immemorial. But today they take new forms. The biggest trick is to elevate yourself and demean the listener at the same time, or, more generally, the technique is

making people feel stupid while acting like a big cheese. Thus someone might say,

"People must begin to get used to the objective scientific ways of doing things that computers now make necessary."

But the translation seems to be:

"People must get used to the inflexible, badly thought out, inconvenient and unkind systems that I and other self-righteous individuals and companies are inflicting on the world."

### YOU DON'T ALWAYS GOTTA

The uninformed are bulldozed, and even the informed are pressured, by the foolish myths of the clever, implacable

and scientific computer to which they must adapt. People are told they have to "relate to the computer." But actually they are being made to relate to systems humans have designed around it, in much the same way a sword dance is designed around the sword.

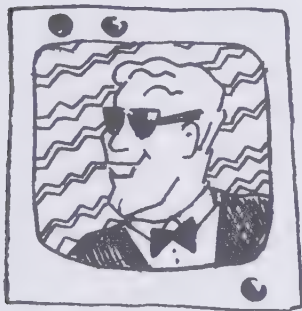
When establishment computer people say that the computer requires you to be systematic, they generally mean you have to learn *their* system. But anyone who tells you a method "has to be changed for the computer" is usually fibbing. He *prefers* to change the method for the computer. The reason may be bad or good. Often the computer salesman or indoctrinator will present as "scientific" techniques which were doped out or whomped up by a couple of guys in the back room.

Here is an example, as told to me. A friend of mine worked in a dress factory where they had a perfectly good system for billing and bookkeeping. Customers were listed by name and kept in alphabetical order. The fast pace of the garment industry meant that companies often changed names, and so various companies had a number of different names in the file. This bothered nobody because the people understood the system.

Then management bought a small computer, never mind what brand, and hired a couple of guys to come in and put the bookkeeping system on it.

Still okay. Indeed, small programming firms can sometimes do this sort of thing very well, because they can work flexibly with the people and don't necessarily feel

## Cybercrud for the Eighties



### MAX HEADROOM

Supposedly a jerky, wisecracking figure named Max Headroom is "the first computer-generated talk-show host." But it's just actor Matt Frewer with rubber make-up, slowed-down video and staccato sound processing. (People, 1 December 1986.)

### BEAUTIFUL BUNNY BOOTIES

Cybercrud is not aimed only at laymen. It can work even among insiders.

The operations manager of a national time-sharing service, for example, was fanatical about cleanliness. In order to assure a Clean Computer Room, he said, and hence no dangerous dust near the tapes or disks, he made a rule requiring that anyone entering the computer room *had to wear cloth booties over his shoes*.

Booties were hung outside for those who had to enter.

"And I had the greatest time making *his*," says his wife, laughing. "With the cutest little bunny faces on them. The buttons were the hardest part to get—you know, the ones with eyes that *roll*!" She laughs very hard as she tells this.

"Of course there was no need for it," he now chortles, "but it sure kept people out of the computer room."

(That's applied logic for you.)

### WHERE ARE THEY NOW?

Applied Logic, the Princeton-based time-sharing service where (as I hint above) this actually happened, is now defunct. Their services were excellent—they kept more backups than anybody—but their prices were high. Friends of mine once had six million dollars worth of stock. It turned to waste paper.

TIME,  
15 April  
1985  
shows a  
classroom  
of Soviet  
computer  
students  
wearing  
white  
coats!

### THE CARGO-CULT ASPECT

Outsiders are often prey to cybercrud they dream up themselves. I once knew a college registrar's office where they had been getting along fine for years with paper forms. The year before the computer was slated to arrive, they started using file cards filled out by hand, instead. Why?

"Well, we thought that would make it easier for the computer. Computers use cards, don't they?"

### The DAMNED LIE:

"Computers are rigid and inhuman."

A BETTER APPROXIMATION:

People are sometimes (all too often) rigid and inhuman. (Machines and animals are *nonhuman*—the term "inhuman" applies only to people.)

"Rigid and inhuman" computer systems are the creation of rigid and inhuman people.

committed to making it work a certain way.

Well, this was a nice instance where the existing system could have been exactly transferred to the computer. The fact that some customers had several names would certainly have been no problem; a program could have been written that allowed users to type any acceptable customer name, causing the computer to look up the correct account (and if desired, print its usual name and ask for verification).

But no. The guys did not answer employees' questions comprehensibly, nor did they want suggestions. They immediately decreed that since computers only worked with numbers (a fib, but a convenience to them), every customer would thenceforth have to be referred to by number.

After that the firm had nothing but trouble, through confusion over the multiple names, and my friend predicted that this would destroy the company. I haven't heard the outcome.

This story is not necessarily very interesting; it merely happened. It's not a made-up example.

Moral: until we overthrow the myth that people always have to adapt to computers, rather than the other way around, things will never go right. Adaptations should take place on both sides, darn it.

### EVERYBODY DOES IT

Cybercrud is by no means the province of computer people alone. Business manipulators and bureaucrats have quickly learned the tricks. Companies do it to the public. The *press*, indeed, contributes (see "Suggestions for Writers and Spokesmen," p. 44). But the computer people are best at it because they have more technicalities to shuffle around magically; they can put anybody down.

Now, computer people do deserve respect. So many things that people do with computers are *hard*. It can be understood that they want to be appreciated, and if not for the particulars, for the *machismo* (machinismo?) of coping with intricacy. But that is no excuse for keeping others in controlled ignorance. No man has a right to be proud that he is preserving and manipulating the ignorance of others.

**You call up the bank and ask your balance and they say,**

**"I'm afraid I can't get that information. You see, it's on a computer."**

**(See Basic Rejoinder, p. 39.)**

**Well, the reason it's that way is that they're handling things in Batch (see p. 102) and they aren't storing your account on disk, or if they are they don't have a terminal they can query it with.**

**But to say that they can't get the information because it's on a computer is a typical use of the computer as an excuse; and second, if the person believes this to be an explanation, it's a sign of the intimidation and obfuscation that have been sewn up among the clerks who don't understand computers.**

**Write them a letter. Change banks. Let's get the banks to put on more and more citizen services. Rah!**

# DAMN

→That Computer!

Everybody blames the computer.

People are *encouraged* to blame the computer. The employees of a firm, by telling outside people that it's the computer's fault, are encouraging public apathy through private deceit. The pretense is that this thing, the computer, is rigid and inhuman (see "The Myth of the Machine," p. 31) and makes all kinds of stupid mistakes.

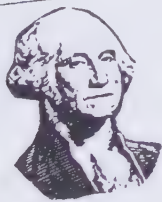
Computers rarely make mistakes. If the computing hardware makes a hardware error in a billion operations, it may be noticed and a repairman called. (Of course, once in a billion operations is once in a thousand seconds, or perhaps every ten minutes. That ought to be mentioned.) Anyhow, innocent gadgetry is not what forces you to make stupid multiple choices on bureaucratic forms; mere equipment isn't what loses your subscription records;

IT'S THE SYSTEM.

By the system we mean the whole setup: the computer, the accessories that have been chosen for it, its *plan of operation* or program, and the way files are kept and complaints handled.

Don't blame the computer.

Blame the system; blame the programmer; blame the procedures; best of all, *blame the company*. Let them know you will take your business to wherever they have human beings. Same for governmental agencies: write your congressman. And so on.



Dear Depositor:

Your bank is now utilizing a computer to provide you with better banking service. This new computer requires the use of a three part deposit slip. Enclosed you will find a supply of these new deposit slips. Please compare the account number on the deposit slips with the one imprinted on your checks to be sure the numbers are identical. If they agree, please start using them immediately. We recommend that you carry a few of these deposit slips in the cover of your checkbook.

If there are any questions about this new procedure, any one of our officers will be glad to help you.

PIFFLE. (The program requires it.)



**THE BIG LIE** (Cybercrud '75, Antitrust Division) The backbone of IBM's defense in antitrust is this whopper:

**MAJOR PREMISE** "Computers are so complicated only a company as large as IBM can put together the technical teams necessary to make them work."

**THE COROLLARY** "Computers are so complicated that there's just no way to make it possible for competitors to hook up their equipment to ours."

**THE TRUTH** almost anybody can make sensible computers that work and tie together sensibly. Only IBM can do it wrong and make it stick.

Artificial Intelligence enthusiasts unfortunately tend to have a magician's outlook: to make clear how their things work would spoil the show.

Thus, for a rather peculiar art show held at New York's Jewish Museum in 1970, a group from MIT built a large device that stacked blocks under control of a minicomputer (Interdata brand). Now, the fact that it could stack and re-stack blocks with just a minicomputer was really quite an accomplishment, but this was not explained.

Instead, the block-stacking machine was enclosed in a large glass pen, in which numerous gerbils—hoppy little rodents—were free to wander about. When a gerbil saw that a block was about to be stacked on him, he would sensibly move. Now, it is fairly humorous, and not cruel, to put gerbils into a block-stacking machine. But this was offered to the public as a device partaking of a far more global mission, the experimental interaction of living creatures and a dynamic self-improving environment, blah blah blah.

Passersby were awed. "Why are those animals in there?" one would say, and the more informed one would usually say, "It's some kind of scientific experiment."

Well, this is a twilight area, between science and whimsical hokum, but one cannot help wishing simple and humorous things could be presented with their simplicity and humor laid bare.

I remember watching one gerbil who stood motionless on his little kangaroo matchstick legs, watching the Great Grappler rearranging his world. Gerbils are somewhat inscrutable, but I had a sense that he was worshipping it. He did not move until the block started coming down on top of him.

I take this as an allegory.

WHAT SECRET  
POWERS DOES  
THIS MAN  
POSSESS?



Practice saying them loudly and firmly to yourself. That way you won't freeze when they're pulled on you.

THAT'S NOT HOW YOU DO IT

THAT'S NOT HOW YOU USE COMPUTERS

THAT'S NOT WHAT YOU DO WITH COMPUTERS

THAT'S NOT HOW IT'S DONE

THAT'S NOT PRACTICAL

HOW MUCH DO YOU KNOW ABOUT COMPUTERS?

WITH YOUR BACKGROUND, YOU COULDN'T UNDERSTAND IT

LET'S CALL IN SOMEONE WHO KNOWS THIS APPLICATION (generally a shill)

IT ISN'T DONE (you know the answer to that one)

and the one I've been waiting to hear,

IF GOD HAD INTENDED COMPUTERS TO BE USED THAT WAY, HE WOULD HAVE DESIGNED THEM DIFFERENTLY.

Unfortunately there is no room here to coach you on how to reply to all these. Be assured that there is always a reply. The brute-force brazen comeback, equally dirty, is just to say something like

DIDN'T YOU SEE THE LAST JOINT PROCEEDINGS?

or

OH YEAH? WHAT ABOUT THE x WORK USING A y?

(where x is anyplace on the map on p. 26, and y is any current computer, such as PDP-10.)

◀ Cybercrud is, of course, just one branch of  
◀ THE GREAT GAME OF TECHNO-LOGICAL  
◀ PRETENSE that has the whole world in its grasp.

◀ "Man, woman, child—  
◀ all is up against the Wall of Science."

◀ —Firesign Theater

# THE MYTH OF THE MACHINE: A DEEP CULTURAL ENGRAM

Public thinking about computers is heavily tinged by a peculiar image which we may call the Myth of the Machine. It goes as follows: there is something called the Machine, which is Taking Over The World. According to this point of view The Machine is a relentless, peremptory, repetitive, invariable, monotonous, inexorable, implacable, ruthless, inhuman, dehumanizing, impersonal Juggernaut, brainlessly carrying out repetitive (and often violent) actions. Symbolic of this is of course Charlie Chaplin, dodging the relentless, repetitive, monotonous, implacable, dehumanizing gears of a machine he must deal with in the film *Modern Times*.

Ordinarily this view of The Machine is contrasted with an idea of a Warm Human Being, usually an idealized version of the person thinking these thoughts.



But consider something. The model often goes further than this. The Machine is cold, the Human Being emotional and warm. Yet there is such a thing as being too emotional and warm. There is in fact a third type in the schema, the being who goes too far on the same scale. Strangely, he has at least three different names, though the picture of him is abstractly the same:



Now, "bums," "niggers" and "hippies" are not real people. The words are derogatory slang for the destitute, for persons with any African ancestry, and for people dressing in certain styles. But the remarkable thing about the slang is that all three of these derogatory terms seem to have the same connotation in our culture: someone who is *dirty, lazy, and lascivious*. In other words, whatever distinguishes The Machine from the Warm Human Being is carried too far by the bunch at the other end.

In other words, this conceptual continuum is a single, fundamental scale in our culture; why is unclear. Since most people consider themselves—naturally!—to be in the middle category, it acts as a sort of reference continuum of two bad things on either side.

It also has another effect: it supplies a derogatory way of seeing. On the right-hand side, it allows many Americans not to see, or to see only with disgust, the destitute and those with African ancestry and those dressing in hippie style.

But this book isn't about that.

The left side of the continuum is our present concern.

There, too, people refuse to see. What people mainly refuse to see is that *machines in general aren't like that*, relentless, repetitive, monotonous, implacable, dehumanizing. Oh, there are some machines like that, particularly the automobile assembly line. But the assembly line was designed the way it is because it gets the most work out of people. It gets the work it does out of people by the way it exerts pressure.

So here we see the same old trick: people building a system and saying it has

to work that way because it's a machine, rather than because that's how I designed it.

To make the point clearer, let's consider some other machines.

The *automobile* is a machine, but it is hardly the repetitive, "dehumanized" thing we usually hear about. It goes uphill, downhill, left and right, fast and slow. It may be decorated. It is the scene of many warm human activities. And most importantly, *automobiles are very much the extension of their owners*, exemplifying life-style, personality, and ideology. Consider the Baja Buggy Volkswagen and the ostentatious cushy Cadillac. Consider the dashboard ornament and the bumper sticker. The Machine, indeed.

The *camera* is a machine, but one that allows its user to freeze and preserve the views and images of the world he wants.

The *bicycle* is a machine, but one that brings you into personal and non-polluting contact with nature, or at least that stylized kind of nature accessible to bicycle paths.

To sum up, then, The Machine is a myth. The bad things in our society are the products of bad systems, bad decisions and conceivably bad people, in various combinations. Machines per se are essentially neutral, though some machines can be built which are bad indeed, such as bombs, guns and death-camps.

The myth of The Machine is a curious aspect of our ideology. Is it especially American, or world-wide?

If we ignore this myth we can see each possible machine or system for what it is, and study how it ties in with human life for good or ill, fostering or lousing up

such things as the good life, preservation of species, love and self-respect.

## THE MYTH AND THE RORSCHACH

"The computer is the ultimate Rorschach test," Fred Bales said to me twelve years ago. Dr. Bales, a Harvard psychologist, was somewhat perturbed by the papers he was getting in his seminar on computer modeling in the social sciences. Somewhat nutty people in the seminar were writing somewhat nutty papers for him.

And truer words were never spoken. On this point I find Bales has been terribly, terribly right. The computer is an incredible projective test: what you see in the computer comes right off the back wall of your psyche. In over a decade in the field I have not ceased to marvel at the way people's personalities entwine with the computer, each making it his own—or rejecting it—in his own, often unique and peculiar way, deeply reflecting his concerns and what is in his heart. Yes, odd people are attracted to the computer, and the bonds that hold them are not those of casual interest.

In fact, people tend to *identify* with it.

In this light we may consider the often-heard remarks about computers being rigid, narrow, and inflexible. This is of course true in a sense, but the fact that some people stress it over and over is an important clue to something about them. My own impression is that the people who stress this aspect are the comparatively rigid, narrow and inflexible people.

Other computer experts, no less worthy, tell us the computer is a supertoy,

**The myth of the computer is based on some rather deep cultural patterns.**

**Computers have been branded as "cold" and "impersonal"—reflecting the style in which they have been used—and taking the heat off the cold and impersonal people who have set up their use in the past.**

**The myth has prevented people from seeing computers as they really are: flexible, functionless, tricky machines you can put to any purpose.**

## The B·I·N·A·R·Y Myth

*Just because computers use off-on doodads internally, that doesn't mean they can "deal with only two-way decisions."*

*Just as twenty-six letters of the alphabet can give you all poetry, science and holy books, the two states (on and off) can be combined and interwoven to give you every possible form of data. Including all twenty-six letters of the alphabet, poetry, science and holy books. (See "Data Structure," p. 70.)*



the grandest play machine ever to be discovered. These people tend to be the more outgoing, generous and playful types.

In a classic study, psychiatrist Bruno Bettelheim examined a child who *thought* he was a machine, who talked in staccato monosyllables, walked jerkily and decorated the side of his bed with gears. We will not discuss here the probable origins and cure of this complex; but we must consider that identifying with machines is a crucial cultural theme in American society, an available theme for all of us. And it well may be that computer people are partaking of this same self-image: in a more benign form, perhaps, a shift of gears (as it were) from Bettelheim's mechanical child, but still on the same track.

Some of the computer high-school kids I've known, because of their youth, have been even more up-front about this than adults.

I know one boy, for instance, whose dream was to put a 33ASR Teletype on wheels under radio control, and alarm people at the computer conference by having it roll up to them and clatter out questions impersonally. (If you knew the kid—aloof and haughty-seeming—you might think that's how he approaches people in real life.)

I know a high school boy (not a computer expert) who programmed a computer to type out a love story, using the BASIC "print" command, the only one he knew. He could not bring himself to write the love story on paper.

The best example I can think of, though, took place at the kids' booth (see p. 110) at a computer conference. One of the more withdrawn girls was sitting at an off-line video terminal, idly typing things onto the screen. When she had gone, a sentence remained. It said:

I love you all, but at a distance.

(On the other side of this book, *Dream Machines*, we will carry this matter further. The most exciting things in the computer field are coming from people trying to realize their wildest dreams by computer: artificial intelligence, computer music, computer picture-making, and so on.)

## I Hate Computers

**Yes, perhaps you do. But what you think you hate may be the narrow, oppressive systems designed by narrow, oppressive people.**

**IF COMPUTERS HAVE SEEMED IMPERSONAL—that is, unsympathetic and unresponsive—it is because they represented the administrative decisions of unsympathetic bureaucracy and/or unsympathetic programmers. And because responsive computers were not available.**

## YOUR INFORMATION SOURCES

There are several major places you get information in the computer field: friends, magazines, bingo cards, conferences and conference proceedings.

### FRIENDS

Friends we can't help with. But you might make some at conferences. Or join a computer club?

### COMPUTER CLUBS

are now in all cities. You join the club that goes with the computer you own. Ask at your local computer store.

### FIND A MENTOR

If you can find someone who gives good advice and explanations—even if younger than you—stick with him or her.

### THE QUAG OF MAGS

Many excellent and informative computer magazines are available. Like Playboy and Penthouse, they help you daydream; but also like Playboy and Penthouse, if you spend too much time on them you'll never have time for the real thing.

BYTE (on the newsstand), the tech-noids' monthly Bible, offers stupefying details on every computer subject, from chip details to higher-level programming. It is easy, reading BYTE, to get the impression that hardware is the important thing. No.

ACM Computing Surveys. Excellent, clearly written introductory articles on a variety of subjects. Any serious beginner should definitely subscribe to Computing Surveys. (See ACM, p. 35.)

Communications of the ACM ("CACM"). High class journal about theoretical matters and events on the intellectual side of the field. (See ACM, p. 35)

Doctor Dobb's Journal of SOFTWARE TOOLS. (See p. 18.)

InfoWorld. P.O. Box 5994, Pasadena, CA 91107, (818) 577-7233. This is the weekly tabloid of the desktop computer industry. Rife with unimportant announcements and arguments, but useful nonetheless. Free or \$100 per year, depending on your buying power in the desktop industry.

### BEST GENERAL BOOKS FOR BEGINNERS (BESIDES THIS ONE)

Peter Laurie, *The Joy of Computers*. Little, Brown, 1983, \$20.

Excellent, handsome, encyclopedia-style book on all aspects of computerdom, written to British standards of literacy.

Steve Ditlea, *The Digital Deli*. Workman Press.

A wonderful anthology of reminiscences, anecdotes and ideas covering the whole computer field.

### SOME GOOD BOOKS FOR BEGINNERS

Thomas H. Crowley, *Understanding Computers*. McGraw-Hill. This is the most readable and straightforward introduction to the technicalities around.

Jeremy Bernstein, *The Analytical Engine*. Random House, 1964. History of computers, well told, and the way things looked in 1964, which wasn't really very different.

Gilbert Burck and the Editors of Fortune, *The Computer Age*. Harper & Row. Ignore the ridiculous full title, *The Computer Age and Its Potential for Management*; this book has nothing to do with management, but is a nice general orientation to the field.

BEWARE: Introductory texts tend to be about ten years outdated.

MIS Week, 7 East 12th Street, New York, NY 10003, (212) 742-6471, extension 4470. \$50 per year. News of databases and on-line services.

*Computerworld* (actually a weekly tabloid paper). Not free. More up-to-the-minute than most people have time to be. Computerworld, Circ. Dept., 797 Washington St., Newton, MA 02160.

Then there is Stan Veit's Computer Shopper, \$15 per year from Patch Publishing Company, 407 South Washington Avenue, Titusville, FL 32781. This is a monthly inch-thick tabloid with more square feet of bargain ads than BYTE, and quite a few useful articles.

*Computer*. (Formerly IEEE Computer Group News.) \$12/yr. Thoughtful, clearly written articles on high-level topics. Quite a bit on Artificial Intelligence (see flip side). IEEE Computer

Society, 16400 Ventura Blvd., Encino, CA 91316

*Computing Reviews*. Prints reviews, by individuals in the field, of most of the serious computer articles. Useful, but subject to individual biases and gaps. (See ACM, p. 35.)

*IBM Journal of Research and Development*. Showcase technical journal of miscellaneous content. \$7.50/year. IBM, Armonk, NY 10504.

*Journal of the ACM*. A highly technical, math-oriented journal. Heavy on graph theory and pattern recognition. (See ACM, p. 35.)

*Think*. This is the IBM house organ. Presumably free to IBM customers or prospects. IBM, Armonk, NY 10504.

There are also expensive (snob?) magazines, bought by executives. ■

## ABOUT FREE SUBSCRIPTIONS

Many of the magazines are free to "qualified" readers, usually those willing to state on a signed form that they influence the purchase of computers, computer services, punch cards, or the like. (They ask other questions on the form, but *whether you influence purchase* is usually what decides whether they send you the magazine.) It is also helpful to have a good-sounding title or company affiliation.

(Just to understand what this can mean, and square it with your conscience, see what the R.E.S.I.S.T.O.R.S. said about this, p. 110.)

### "COMPUTER BASICS"

The idea of "computer basics" is still, in a sense, a misleading fiction.

There are no basics, there is no center, and the mistake of the textbooks is to pretend there is one and narrow it down too far. (This book is meant to convey the real sweep and panorama.)

Hey now, here's a magazine called *Computopia*. Only \$15 a year. Unfortunately in Japanese.

Computer Age Co. Ltd.,  
Kasumigaseki Bldg.,  
Box 122, Chiyoda-Ku,  
Tokyo, Japan.

## What Happens If You Take Computer Courses?

There is a lot of talk about "best" ways of teaching about computers, but in most places the actual alternatives open to those who want to learn are fairly dismal.

*Universities*. Universities and colleges tend to teach computing with a mathematical emphasis at the start. Indeed, most seem to require that to get into the introductory computer course, you must have had higher math (at least calculus, sometimes matrix algebra as well). This is preposterous, like requiring an engineering degree to drive a car. (Gradeschool kids can learn to program with no prerequisites.)

It seems to be to cut down enrollment, since they're not set up to deal with all those people who want to learn about computers. (And why not?) Also it's a status thing; as if this restriction somehow should keep enrollment to students with "logical minds," whatever those are, or "mathematical sophistication," as if that were relevant.

*Computer schools*, community and commercial colleges, on the other hand, tend to prepare students only for the most humdrum business applications—keypunching (which is rapidly becoming obsolete), and programming in the COBOL language on IBM business systems. This gets you no closer to the more exciting applications of computers than you were originally.

Some experimental trends are more encouraging. Some colleges, for instance, offer "computer appreciation courses," with a wider introduction to what's available and more varied programming intended to serve as an introduction to this wider horizon.

*Highschool courses* seem to be cutting through the junk and offering students access to minicomputers with quickie languages, usually BASIC. Both Digital Equipment Corp. and Hewlett-Packard seem to be making inroads here.

"Computer science" is the popular academic name for what's taught to prospective computer people in colleges and universities. That's the good news. The bad news is that the term has become attached only to the most pompously academic and mathematical-type work, and so "computer science" as generally taught—for instance, based on ACM's old Curriculum '68—LEAVES OUT INTERACTIVE SYSTEMS, TEXT SYSTEMS AND GRAPHICS—THE PRINCIPAL COMPONENTS OF OUR WORLD OF TOMORROW.

The computer-teaching stuff on TV is also pretty bad. A series called "The New Literacy," in syndication on PBS and possibly elsewhere, presents the mainframe point of view.

Another series has a roomful of youngsters learning to program in BASIC from an instructor who reads loudly from a workbook. They are seen at TRS-80 computers, machines which are gone now and not lamented. But the show lingers on.

COMPUTER CAMPS have been around since the seventies. The best one I heard of started you on LISP (instead of BASIC). But unfortunately the camps are generally only for kids.

Then there are these courses called "The Impact Of Computers On Society." There's just one problem: THE IMPACT OF COMPUTERS ON SOCIETY HASN'T BEEN DECIDED YET. So far, every year's impact has been different.

### LEARNING HARDWARE:

Heathkit, Benton Harbor, MI, has for decades offered superb teaching kits showing various aspects of electronics, with instructional manuals that explain a great deal. They now have a PC-compatible computer kit. Do that in the evenings for a couple of years and you may be ready to repair computers professionally.

HOWEVER, note that that doesn't help you understand programming.



(INCIDENTALLY—)

People ask me often where they can learn about "science." As in all fields, magazines are usually the best sources of general orientation.

*Scientific American* is the favorite. Some stuff is hard to read but some isn't; the pictures and diagrams are terrific.

*Science* magazine is read by most actual scientists, and if you have a lively curiosity and can guess at the meanings of words, will tell you an incredible amount. (This is a main source for the science articles in the *New York Times*, which in turn...) Their articles on politics of science, and the future, are very interesting, important, and depressing. You have to join Am. Assn. for the Advancement of Science, Washington, D.C.

Daniel S. Greenberg's *Science and Government Report* (sorry—\$35 a year) is what really tells it. Greenberg is the man who knows, both what is shaping up in science and the insane governmental confusions and floundering responses and grandstanding and pork-barrel initiatives...

Greenberg is, incidentally, one of the finest writers of our time and a great humorist.

*Science and Government Report*, Kalorama Station (really?), Box 21123, Washington, D.C. 20009.

This is the wall that the handwriting is on.

**THERE IS NO GOOD PLACE TO LEARN PROGRAMMING** outside the universities. Best way may be on your own, but bad habits come easily. See books cited in "Programming" section, p. 40.

If you really want to understand the world of High Computerdom without learning to program, read *The Hacker's Dictionary: A Guide To The Computer Underworld*, by Guy Steele and True Names, by Vernor Vinge, Bluejay Books, 1984, with afterword by Marvin Minsky. These give you the best feel for the hacker world.

## BUMMERS

This is perhaps a minority view, but I think any introduction to computers which makes them seem intrinsically mathematical is misleading. Historically they began as mathematical, but now this is simply the wrong way to think about them. Same goes for emphasizing business uses as if that were all.

We will not name here any of the various disagreeable pamphlets and books which stress these aspects and don't make things very clear.

### Computer Toys

— A Warning

A number of inexpensive gadgets purport to teach you computer principles. Many people have been disappointed, or worse, made to feel stupid, when they learn nothing from these. Actually the best these things really can do is give you an idea of what can be done with combinations of switches. From that to learning what computer people really think about is a long, long way.

## MAGAZINES FOR SPECIFIC MACHINES

There are magazines for virtually every computer: PC and PC World, MacWorld and MacUser and MacTutor, AmigaWorld, and magazines for smaller machines like the Commodore 64, the Apple II and the laptops.

Then there are UNIX magazines, such as UNIX World.

## MAINFRAME VIEWPOINT

*Datamation*. \$55 a year, \$40 for students, or free. The main computer magazine, a breezy, clever monthly. Lots of ads, interesting articles the layman can read with not much effort. Twits IBM. Subscriptions are \$55 if you're not a computer person, free if you are. *Datamation*, 875 Third Avenue, New York, NY 10022.

## BINGO CARDS.

These are little postcards you find in all the magazines except the ACM and company ones. Fill in your name and an attractive title ("Systems Consultant" or "consultant" is good—after all, someday someone may ask your advice) and circle the numbers corresponding to the ads that entice you. You'll be flooded with interesting, expensively printed, colorful, educational material on different people's computers and accessories. And note that senders don't lose: any company *wants* its products known.

However, a post office box is good, as it helps to avoid calls at home from salesmen, wasting their time as much as yours. If you are in a rural-type area where you can assume a company name with no legal difficulties, so much the better.

## CONFERENCES.

Conferences in any field are exciting, at least till you reach a certain degree of boredom with the field. Computer conferences have their own heady atmosphere, compounded of a sense of elitism, of being in the witches' cauldron, and the sure sense of the impact everything you see will have as it all grows and grows. Plus you get to look at gadgets.

Usually to go for one day doesn't cost much, and at the bigger ones you get lots of free literature, have salesmen explain their things to you, see movies, hear fascinating (sometimes) speakers.

**THE JOINTS!** The principal computer conferences have always been the Spring Joint Computer Conference, held in an Eastern city in May, and the Fall Joint Computer Conference, held in a Western city in November (the infamous Spring Joint and Fall Joint, or SJCC and FJCC). In 1973, because of poor business the previous year, the two were collapsed into one National Computer Conference (NCC) in June (Universal Joint?). The Joints have always been sponsored by AFIPS (see nearby). The National Computer Conference will henceforth be annual, at least for a while.

The cost of attending is high—while it's just a couple of dollars to look at the exhibits, this rises to perhaps fifteen dollars to go to the day's technical sessions or fifty for the week (not counting lodging and eats)—but it's very much worth it. The lower age limit for attendees is something like twelve, unfortunately for those with interested children.

Other important conferences: the annual ACM conference in the summer; SIGGRAPH; and other conferences on special subjects, held all the time all over. Lists of conferences and their whereabouts are in most of the magazines; *Communications of the ACM* and *Computer Design* have the biggest lists.

*On most days of the year, there are half-a-dozen computer conferences around the country where papers are being presented—not to mention shows, expos, exhibitions and swap meets—for business computing, scientific computing, computer graphics and specific machines (such as MacWorld and PC Expos). The academic ones are listed in BYTE and the Communications of the ACM. Your local commercial shows will be advertised in your paper or at your computer store.*

**CONFERENCE PROCEEDINGS** (such as "Proc. ACM 65"; "Proc. SJCC 68"; "Proc. NCC 73").

As you may know, conferences largely consist of separate "sessions" in which different people talk on specific topics, usually reading out loud from their notes and showing slides.

Conference *proceedings* are books which result from conferences. Supposedly they contain what each guy said; in practice people say one thing and publish another, more formal than the actual presentation.

This leads to a curious phenomenon at the main computer conferences (SJCC, FJCC, ACM and now NCC). When you register they *give you a book* (you're actually paying perhaps \$15 for it), containing all the papers that are about to be given, nicely tricked out by their authors. **If you rush to a corner and look at the book it may change your notion of which sessions to go to.**

Anyway, the resulting volumes of conference proceedings are a treasure trove of interesting papers on an immense variety of computerish and not-so-computerish subjects. (Great for browsing. Expensive but wonderful. Horrible when you're moving, though, as they are big and heavy.)

**JOINT PROCEEDINGS.** Proceedings for the Spring Joint and Fall Joint, from the fifties to 1972, are available from AFIPS Press, as are proceedings of the NCC since 1973. (AFIPS Press, 210 Summit Avenue, Montvale, NJ 07645.) They cost \$20-26 each after the conference is over; less in microfilm. (At the Joint Conferences, AFIPS Press often gives discounts, at their booth, on back Joint proceedings.)

♦ If you want to spend money to learn about the field, Proceedings of the Joint Conferences are a fine buy.

*Back ACM Proceedings.* From the ACM.

*Other Proceedings.* Often sold at counters at conferences. Or available from various publishers. Join the ACM and you'll find out soon enough.

**T**RY TO GET TO THE NATIONAL JOINT. Just as every Muslim should go to Mecca, every computer fan should go to a National Joint (National Computer Conference, or NCC).

**NO QUALIFICATIONS ARE NEEDED.** Think of it as a circus for smart alecks, or, if you prefer, a Deep Educational Experience.

## The MAIN Computer Organizations

ACM, the Association for Computing Machinery. This is the main computer professional society; the title only has meaning historically, as many members are concerned not with machinery itself, but with software, languages, theories and so on.

If you have any plans to stick with the subject, membership in the Association for Computing Machinery is highly recommended. ACM calls itself "The Society of the Computing Community." Thus it properly embraces both professionals and fans.

Dues for official students are \$30 a year, \$65 for others, which includes a subscription to *Communications of the ACM*, the official mag. Their address for memberships and magazines is ACM, 11 West 42nd Street, New York, N.Y. 10036.

They have stacked the deck so that if you want to subscribe to any ACM magazines you'd better join anyway. Here are the yearly prices:

	Member	Non-Member	Student Member
<b>Communications of the ACM</b>	<b>free</b>	<b>\$90</b>	<b>free</b>
<b>Computing Surveys</b>	<b>\$11</b>	<b>\$75</b>	<b>\$6</b>
<b>Computing Reviews</b>	<b>\$30</b>	<b>\$95</b>	<b>\$25</b>
<b>Journal of the ACM</b>	<b>\$15</b>	<b>\$75</b>	<b>\$10</b>

The one drawback to joining the ACM is all the doggoned mailing lists it gets you on. It's unclear whether there's anything you can do to prevent this, but there oughta be.

*SIGs and SICs.* For ACM members with special interests (and we all have them), the ACM contains subdivisions—clubs within the club, of people who keep in touch to share their interests. These are called SICs (Special Interest Committees) and SIGs (Special Interest Groups). There are such clubs—SICs and SIGs—in numerous areas, including Programming Languages, Computer Usage in Education, etc. Encouraging these subinterests to stay within ACM saves a lot of trouble for everybody and keeps ACM the central society.

*The most important SIG is SIGGRAPH, the computer-graphics SIG, which gives great conferences of its own (see p. DM 80).*

AFIPS is the UN of computing. They sponsor the NCC. Just as individuals can't join the UN, they can't join AFIPS, which stands for American Federation of Information Processing Societies.

IFIP. This is the international computer society. Like AFIPS, its members are societies, so joining ACM makes you an IFIP participant. IFIP holds conferences around the world. Fun. Expense.



# THE POWER AND THE GLORY

Forget what you've ever heard or imagined about computers.

Just consider this:

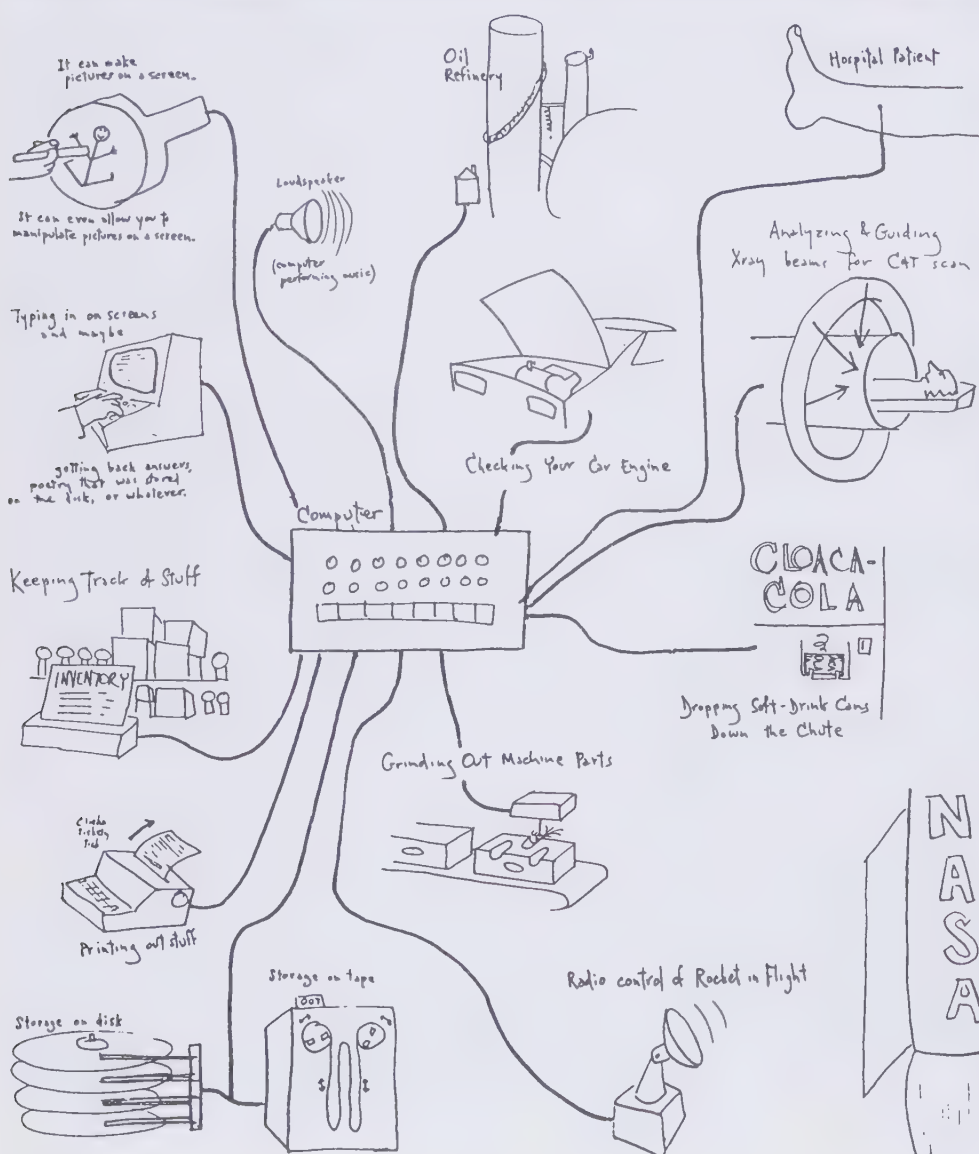
The computer is the most general machine man has ever developed. Indeed, it should be called the All-Purpose Machine, but isn't, for reasons of historical accident. Computers can control, and receive information from, virtually any

other machine. The computer is not like a bomb or a gun, which can only destroy, but more like a typewriter, wholly non-committal between good and bad in its nature. The scope of what computers can do is breathtaking. Illustrated are some examples (although having all this happen on one computer would be unusual). It can turn things on and off, ring bells,

put out fires, type out on printing machines.

Computers are incredibly dogged. Computers can do things repeatedly forever, or an exact, immense number of times (like 4,901,223), doing something over and over, depending on whether it's finished or not. A computer's activities can be combined in remarkable ways.

One activity, repeated over and over, can be part of another activity repeated over and over, which can be a part of still another activity, which can be repeated ad infinitum. THERE ARE DEFINITE LIMITATIONS on what computers can do, but they are not easy to describe briefly. Also, some of them are argued about among computer people.



## METHODS FOR DOING THINGS

There are really no clear bounds on "what computers can do."

The problem is always to think up *methods* for doing things by computer. (Also called *algorithms*.)

Basically, what can be done by computer is what can be done on a tabletop with slips of paper—comparing, copying, sorting, marking, doing arithmetic—and handing slips of paper out to users.

So the question should never be, "how would you do that by computer"—but "Can you think of a *method* for accomplishing that?" The "computer" is really irrelevant, for it has no nature and merely twiddles information on demand.

*There is no "THE COMPUTER."*

For years the public—and especially the more humanistic writers—have imagined that there is something they call the computer, which forced people to act a certain way. In fact this simply is not so, but many people benefitted from perpetuating this myth (see "Cybercrud," p. 27, and "Computer Center Politics," p. 161).

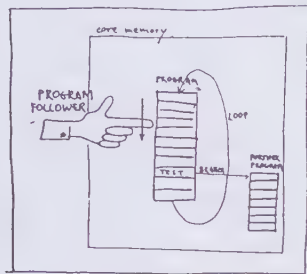
*There is no "COMPUTER WAY OF DOING THINGS."*

But people often think there's a specific computer way because they've seen a certain computer doing something a certain way, and figured that was how it had to be. 'Tain't so. Often it's just the way a particular system is set up; and if all computers do it, it's only because certain practices (conventions, tyrannies) have become traditional.

## THE DEEP DARK SECRET THE MAGIC OF THE COMPUTER PROGRAM

The basic, central magical interior device of the computer we shall call a *program follower*. A program follower is an electronic device (usually) which reads symbols specifying operations, carries out the step each specifies and goes on to the next.

The program follower reads down the list of instructions in the program, taking each instruction in turn and carrying it out before it goes on to the next.



The commands of the program cause the program follower to carry out the individual steps requested by each command of the program.

Now, there are program followers that just do that and nothing more; they have

to stop when they get to the end of the list of instructions.

A true computer, however, can do several things more.

### Principle 1: The Program Loop

It can jump back to an earlier point in the program and go on from there. Repeating the program in this fashion is called a *loop*.

### Principle 2: The Program Branch

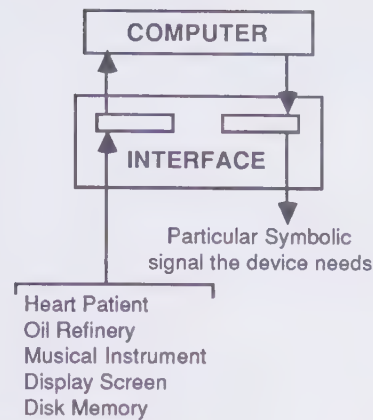
It can perform tests on symbols in the memory—for instance, to see if a loop has been done enough times, or if some other part of the job has been finished—and jump to some other program *depending* on these symbols. This is called a *branch*.

Finally, the computer can *change* the information stored in memory. For instance, it can place an *answer* in a specific part of memory.

### HOW CAN A COMPUTER CONTROL SO MANY DIFFERENT THINGS?

#### Principle 3: All Devices Look Alike

*Answer.* Different as they may seem, all devices are controlled in the *same* way. Every device has an *interface*, that is, its own special connection setup.



The computer, being a machine, doesn't know or care that device 17 (say) controls a hog feeder, or device 23 (say) receives information from smog detectors. But what you choose, in your program, to put into device 17, controls what the hogs eat, and what comes into device 23 will tell your program, you hope, about smog conditions. Choosing how to handle these things in your program is your business.

Here's the computer, then, in all its glory: a device with a symbolic program, stored in a memory, being stepped through by a program follower. 🐾

"What is an interface?" asked the baby machine.

"Whatever Turns You On," said its dad.

Whatever it may do in the real world, to the computer program it's just another device.

## The ALL-PURPOSE Machine

Computers are COMPLETELY GENERAL, with no fixed purpose or style of operation. In spite of this, the strange myth has evolved that computers are somehow "mathematical."

Actually von Neumann, who got the general idea about as soon as anybody (1940s), called the computer

### THE ALL-PURPOSE MACHINE.

(Indeed, the first backer of computers after World War II was a maker of multi-lightbulb signs. It is an interesting possibility that if he had not been killed in an airplane crash, computers would have been seen first as text-handling and picture-making machines, and only later developed for mathematics and business.)

We would call it the All-Purpose Machine here, except that for historical reasons it has been slapped with the other name.

But that doesn't mean it has a fixed way of operating. On the contrary.

### COMPUTERS HAVE NO NATURE AND NO CHARACTER,

save that which has been put into them by whoever is creating the program for a particular purpose. Computers are, unlike any other piece of equipment, perfectly BLANK. And that is how we have projected on it so many different faces.

## COMPUTERS DEFINED JUST LIKE CAMERAS AND CARS

Just the way everyone can understand cameras, viz.: "A camera is a device you point at something to willfully capture its appearance."

Just the way everyone can understand cars, viz.: "A car is a device people get inside which then goes somewhere else, under the willful control of the driver."

Well, how about "A computer is a device which manipulates information and external accessories, according to a plan willfully prepared by a planner."

## How Does the LOOP Work?

The computer does things over and over by changing a *stored count*, then testing the stored count against another number which is what the count should get to, and going to the beginning if the desired count has not been reached. This is called a *loop*. (If there's no way it can ever get out, that's an *endless loop*.)



How does a computer program print something out on a printing machine? It sends the code for each letter out to the printing machine.

How does a computer program respond to something a user types in? It compares the codes that come in from the letters he types with a series of codes in memory, and when it finds a match between letters, numbers, words or phrases, branches to the corresponding action.

How does a computer program measure something? It takes in numerical codes from a device which has already made the measurements and converted them to codes.

**E<sup>x</sup>act Science, Indeed**  
Laymen are under the false impression that computing is an exact science.  
On the contrary. It is a babel of different religions and points of view.

**Some computers also work with a mostly unchangeable memory. For instance, the processors in soft-drink machines, video games, refrigerators and VCRs. Some people might haggle and say those aren't real computers if the program can't be changed, but for the present purposes the distinction is minor.**



## WHAT, THEN, IS A (Digital) COMPUTER?

A device holding stored symbols in a changeable memory,

performing operations on some of those symbols in the memory,

able to change the sequence based on tests of symbols in the memory,

and able to change symbols in the memory.

(For example, do arithmetic and store the result in the memory.)

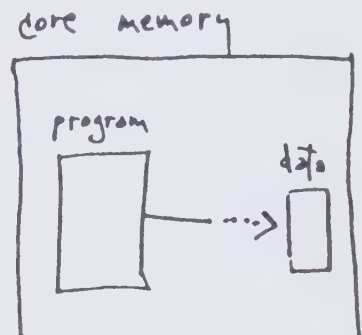
Rather than try to slip it to you or prove it in some fancy way, let's just state baldly: the power of such a machine to do almost anything surpasses all previous technical tricks in human history.

## WHAT DO PROGRAMS LOOK LIKE?

In what forms are these programs stored, you ask? Well, they are written by people in computer *languages*, which are then stored in some form in the computer's fast core memory, where the program follower can act on them.

We'll get to computer languages on p. 56.

Other things besides programs may be stored in the memory. Anything besides programs are usually called data.



The instructions of programs use the data in different ways. Some programs use a lot of data, some use a little, some don't use any. It is one of the fascinating and powerful things about the computer that both the instructions of a program, and the data they work on, are stored as patterns of bits in the same memory, where they can be modified as needed. Indeed, the program can modify its own patterns of bits, a very important feature.

Many ordinary people find computers intuitively obvious and understandable; only the complications elude them. Perhaps these intuitively helpful definitions may help your intuition as well.

1. Think of the computer as a WIND-UP CROSSWORD PUZZLE.

2. A COMPUTER IS A DEVICE FOR TWIDDLING INFORMATION. (So, what kinds of information are there? And what are the twiddling options? These matters are what the computer field consists of.)

3. A computer is a *completely general device, whose method of operation may be changed, for handling symbols in any specific way.*

## A HELPFUL COMPARISON

It helps sometimes to compare computers with typewriters. Both handle information according to somebody's own viewpoint.

### Nervous Question

"Can a Computer Write a Poem?"

"Can't Computers Only Behave Mechanistically?"

"Aren't Computers Completely Impersonal?"

### Helpful Parallel

"Can a Typewriter Write a Poem?"  
(Sure. Your Poem.)

"Can't Typewriters Only Behave Mechanistically?"  
(Yes, but carrying out your intent.)

"Aren't Typewriters Completely Impersonal?"  
(Well, it's not like handwriting, but it's still what you say.)

*This new field most people call "computers"—called in Europe by the far more appropriate term "informatics"—is the study of AUTOMATIC PLANS AND MACHINES THAT CAN BE RUN BY THEM.*

*And this field is really about a new kind of thinking. We often call it SYSTEMS THINKING, the designing and revision of intentional systems. In ordinary life we often take things as they come, do what's necessary on an ad-hoc basis; but when planning things to happen automatically, we must weave whole methods of doing things, ways to start and ways to finish and places to put things halfway through.*

*The systems that people create intentionally are rather different from the systems found in nature. (Although the Artificial Intelligence people would like to bring them closer together; see p. DM 120.)*

## LESSON 1: GETTING THINGS STRAIGHT

The greatest hurdle for the beginner (or "layman") is making an effort to grasp particulars of that which he hears about.

A. WHAT IS ITS NAME? Every system or proposal or project has a name of some sort. Make an effort to learn it, or you're stuck trying to refer to "that computerish thing".

(And don't be a snob about acronyms, those all-cap names and terms sprung from the foreheads of other words, like ILLIAC and PLATO and CAI. There's a need for them. Short words are too general to use for names, and long phrases are too unwieldy.)

B. IN WHAT PARTICULAR WAY DOES IT EMPLOY THE COMPUTER? For record-keeping? For looking stuff up quickly or fancily? For searching out combinations? For making up combinations and testing their properties? For enacting complex phenomena? As automatic typewriters? To play music, or just to store written notes?

It is hoped that you will become sensitive to these distinctions, and be able to understand and remember them after somebody explains them.

Otherwise you're stuck just referring to "that computer business," and you're in with the rest of the sheep.

## WHAT YOU'VE SEEN PROBABLY WASN'T "A COMPUTER"

Get out of your head the notion that some one system you've seen showed you what Computers Are Really Like. Computer systems can be as different externally as bats and whales. (Yet it's the same kind of heartbeat, but that's no help in dealing with them.)

## HORRIBLE MISUNDERSTANDINGS

Some people think of computers as things that somehow mysteriously digest and assimilate all knowledge. "Just feed it to the computer." is the motto. But what you feed into the computer just sits there unless there's a program.

"How would you do that by computer?" is a question people often ask. The question should be, "how would you do that at all?" If there is a method for doing something which can be broken down into simple steps, and requires no human judgment, then maybe we can take those steps and program them on a computer. But maybe we can also think of a simpler way to get them done.

Then there is the idea that a computer is something you *ask questions*. This assumes, I guess, the earlier premise, that the computer has already digested and assimilated a lot of stuff and can sling it back at you in new arrangements.

Actually what must happen, to get "questions" answered, is this: there must be some program that puts input material into a data structure. (See "Data Structure," p. 70.) Then you need programs will count and trace, or whatever, through the data structure in ways you desire. Then you need a way to start these tracing-and-searching programs going through the data structure in ways you want. So you need a program accepting input from a keyboard, or whatever, and starting the other programs in operation.

Then what is it computer people know, you may ask, that leads them to understand new systems quickly? Aha. Computer people simply adjust faster to whole new worlds.



## A Basic Reminder

we should all practice and have ready at the tip of our tongues:

WHY THE HELL NOT? YOU'RE THE ONES WITH THE COMPUTERS, NOT ME!

Let's froth up a little citizen indignation here.

## THE AUTOMOBILE ANALOGY (more)

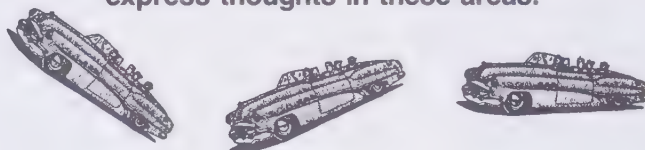
"The Interstate was bumper-to-bumper, but after we had lunch at the rest stop it cleared up till we got to the tollbooth. Harry got lost on the interchange, and we had to double back on the service road."

How incomprehensible to someone from 1905. Yet how simple-minded when you understand it. That's how it is with computers.

Computer talk sounds so strange and incomprehensible to you folks out there—yet to us in here it's often as simple as the lines above—if you know the fundamental concepts.

And nothing in the normal everyday world will have prepared you for them.

It's not jargon, but the simplest way to express thoughts in these areas.



A Program  
is —————> **A Design**  
For Events

A computer program is a structure of planned events—not just a series of things to take place, but an interweaving of many different things that either will or may take place, depending on the data or the time of day or the choice of a user sitting at a keyboard, or whatever.



# PROGRAMMING

To understand computers is to know about programming. (The hardware technicians may disagree, but this is not their book.) Programming is the **WEAVING OF PLANS OF EVENTS** (and where they are to take place)—the choreography of happenings.

The world is divided, as Art Luehrmann has pointed out, into people who have written a program and people who have not. Millions now know how to program in some form (though most are not suited to being professional programmers). But **everyone should write at least a few programs**—not to make a career of it, but just to know what it's about—

just as everyone should know how to do pencil-and-paper arithmetic, even if they don't add up numbers for a living.

If you've never written a program, it's like never having driven a car. You may get the general idea, but you may have little clear sense of the options, dangers, constraints, possibilities, difficulties, limitations and complications.

The principal activity of programming is **THINKING**. (Some TV writers seem to think it's pounding on keyboards.) Good programmers often go around for months scratching their heads before they start seriously writing code. (But if somebody launches right in to scribbling, that

doesn't mean they're bad; that may just be their way of thinking.) This stage is also called **DESIGN**, or sometimes **SYSTEMS ANALYSIS** (except that "systems analysis" in the IBM world means forcing your problems into IBM's solutions).

The next activity is **WRITING THE STEPS**, in whatever language you are using. This is called **CODING**. Then you put the plan for these steps into the computer, and try the program.

Incredible things are possible through programming—but it's very hard.

Managing the complexity of programming turns out to be the frontier of computer science.

Short programs are often much better than long ones for the same purpose; or vice versa. Good design of programs is complex, subtle, and infinitely controversial. It is clear, though, that when programmers are expected to produce a

certain amount of program code each day or week, either they are very badly managed or they are doing low-grade work indeed. (Many well-known software companies repeatedly miss deadlines; and software companies that make their deadlines sometimes produce very poor programs.)

Computer people often say that to understand computers you have to have a "logical mind."

There's no such thing. But saying such things intimidates many, especially those who have been told they do not **have** "logical minds."

What is meant, actually, is indeed important: in working with computers you must often work out the exact ramifications of specific combinations of things, without skipping steps.

But the other mode of thinking, the intuitive, has its place in the computer field

## WRITE A PROGRAM

To learn about computers is to learn about programming, and the way to learn about programming is to write programs. You start by learning a computer language. (As with human languages, it's easier for children than adults, but everybody can.)

Most people start with **BASIC** because it's there. But now on most personal computers it's possible to start with **Logo**, a much better language to start with.

You don't always have a choice of what setup and language to use. There are lots of other languages you can start with. Highly recommended: **APL**. For serious beginners: **LISP**, **Smalltalk**. A completely different starting language: **Lotus 1-2-3**.

## THE BEST WAY TO LEARN.

The best way to start programming is to have a terminal running an interactive language, and a friend sitting nearby who already knows the language and has something else to do but can be interrupted with questions.

And you just try stuff.

Till more and more you get the feel of it.

And find yourself writing programs that work.



THE BEST WAY TO LEARN.

"WHEN YOU  
BREAK THE  
PROBLEM  
INTO PARTS

you've determined its structure, and the work that will be done around it, forever."

—Mark Miller's Law of Irrevocable Subdivision

Good programming is not merely skill.

One of the worst programmers I knew had incredible speed and facility. The corporation he worked for thought he was their backbone—and so did he.

In minutes or hours he would construct a program to do whatever was requested. It would then take the technical writers months to explain it, and even then their writeups were only an approximation.

It is possible that this one guy messed up the corporation's software beyond any hope. The company went under.

Beginners' intuitive idea of "programming" is often a reasonable approximation to the notion of "data structure." Data structure is how information is set up. After it's set up, programs can twiddle it; but the twiddling options are based on how the information is set up to begin with.

too. Whichever your habitual style of mind, computers offer you food—and utensils—for thought.

## BIBLIOGRAPHY

Henry F. Ledgard, *Programming Proverbs*. Hayden, 1975 (different editions available for several programming languages—BASIC, Cobol, Fortran...).

George Ledan Jr. and Victor Ledan, *The Programmer's Book of Rules*. Lifetime Learning Publications, 1979. Good beginner's book on programming and programming style.

Communications of the ACM and ACM Computing Surveys are excellent browsing magazines on all aspects of programming.

Dr. Dobb's Journal of Software Tools (still called Dr. Dobb's by old timers) has good coverage of languages and techniques. So does Byte. A newer

magazine called Computer Language may bear watching.

**SERIOUS READING TOWARD A COMPUTER CAREER** (If you think you may be serious, join the ACM and subscribe to CACM and ACM Computing Surveys.)

Brian W. Kernighan and P.J. Plauger, *The Elements of Programming Style*. McGraw, 1978.

A professional's book of case studies.

Donald E. Knuth, *The Art of Computer Programming*. Addison-Wesley. Vol 1, second edition, 1974; Vol. 2, second edition, 1971; Vol 3, 1973.

Brian W. Kernighan and P.J. Plauger, *Software Tools*. Addison-Wesley, 1976. More about programming than about tools.

Howard Abelson and Gerald J. Sussman with Julie Sussman, *Structure and Interpretation of Computer Programs*. MIT Press, 1985. For when you make a serious career move into the field.

**ESTIMATING THE SIZE OF A PROGRAMMING PROJECT IS EXTREMELY DIFFICULT.** Experienced programmers often underestimate by half. Large corporate programming projects are often underestimated by a factor of ten or more. Beginners can easily underestimate the complexity of a project by a factor of a million.

Nelson's rule of thumb: Take your initial estimate, multiply by a thousand, and divide by the number of weeks you've been in the computer field. In other words, after about twenty years, your estimates tend to be approximately correct.

When programmers are expected to produce a certain amount of program code each day or week, either they are very badly managed or they are doing low-grade work indeed. (Many well-known software companies repeatedly miss deadlines; and software companies that make their deadlines sometimes produce very poor programs.) **THE BEST PROGRAMS ARE CREATED BY THREE PEOPLE OR LESS.** Big software teams guarantee disaster.

## PROGRAM VS SYSTEMS

A Vague Guideline to a Vague Distinction

A "program" runs on an ordinary computer, without necessarily interacting with the outside world; a "system" involves a whole setup, of which the computer and a program in it are just the central things.

**SYSTEM DESIGN** is how many problems are solved today. Much of program design is really the design of a larger system of which the program is a part—but, paradoxically, the design of the program often determines the rest of the system.

The activity of programming can help make you MORE calm and confident. FOR SOME PEOPLE IT IS EXCELLENT THERAPY, providing a series of small rewards and triumphs as the machine is tamed, step by step, to your intent.

Unfortunately programming can also make you very neurotic: imaginative people tend to attempt extremely difficult programming tasks that go on and on.

Still, the most important innovations come from programmers who "don't know it can't be done."



"PROGRAMMING PRACTICES" means the sorts of rules you keep in mind to keep your programs from getting tangled and messy.

Good programming yields good programs—programs which are clean, divided into sensible parts, and easy to modify. Good programs do not have fixes piled on fixes.

### SOME GOOD PROGRAMMING PRACTICES:

1. Don't be afraid to start over (Henry Ledgard). Repeated starts can mean a far cleaner final product.

2. Design long, program short. Don't be in a rush to write code. Think about the problem from every angle, figure out in detail how different strategies affect each other.

When you're sure the design is right, code it.

Impatience is the enemy. Design is productive work.

The time you spend designing (and starting over) will be more than made up for by the cleaner program that results.

Besides writing programs, one of the best ways to study programming is to read programs.

This is very demanding, because you have to suspend your view of how a given task should be done to see how it was done.

"...programs must be written for people to read, and only incidentally for machines to execute. [The problem really is] the techniques used to control the intellectual complexity of large software systems."

—Abelson and Sussman, *Structure and Interpretation of Computer Programs* (MIT/McGraw-Hill), preface.

Because everything you want to program seems so simple in principle, you think you could (in principle) program it all. Ah, youth.

"First think about what the ultimate would be. Then take a step away from that, and another, until you get to something you can build."

Matt Reedy



## DEBUGGING

Once you know a language you can cast spells in it; but that doesn't mean it's easy. A spell cast in a computer language will make the computer do what you want—

IF it's possible to do it with that computer;

IF it's possible to do it in that language;

IF you used the vocabulary and rules of the language correctly;

and IF you laid out in the spell a plan that would effectively do what you had in mind.

BUT if you make a mistake in casting your spell, that is a BUG. (As you see from the IFs above, many types of bug are possible.) Program bugs can cause unfortunate results.

### THIS BUG'S FOR YOU

*DEBUGGING is the next principal ac-*

*tivity of programming—how to correct the program when the machine, perfectly following your instructions, does not do what you intended. (See p. 42.) And you must change and change and change your program until everything comes out right. Making the loops go around in the right direction is only part of the problem.*

Debugging means changing and fixing your program till it works the way you want it to.

This is the part of programming people like the least.

You run your program and then try to find out what went wrong. It could be a mistake in the basic thinking ("logic error"), or a clerical error in the particular choice of commands to carry out a well-thought-out process ("coding error").

Some systems allow you to debug interactively, from a terminal. This helps a lot. You can run parts of your program, get it to stop at certain points to let you look around, and so on.

**S**omeone remarked that if buildings were built to the same standard as computer programs, the first woodpecker would bring down civilization.

According to the grapevine...a prestigious Southern university had a program where the number of months was carelessly set to 10 (as a dimension in an array). In November, nobody got their checks till this error was found.



For every bug that goes out, two more bugs go in.  
—folk saying

## QUIS CUSTODIET. HUH?

▼ Boston welfare recipients have been systematically short-changed for at least 14 years, according to *Computerworld* (10 Oct. '73, p. 2).

A systems analyst recently discovered that the welfare program was not calculating cost-of-living increases on a compound basis, as it should have been, but as a simple increase based each year on an obsolete original figure. However, it's too late to ask for refunds, and anyway not many welfare recipients take *Computerworld*.

According to a speaker at a computer conference a few years ago, a certain military aircraft, when crossing the equator, would suddenly flip over and fly upside-down. It seems that the navigation programmers had overlooked certain coordinate-boundary problems.

According to the *New Yorker* (14 Jan 85, p. 23), the Democrats generated post-election letters of thanks by computer in December '84 that went out over the signature of Geraldine Ferraro. Due to a bug in the interactive design—the inputters "neglecting to hit the 'mister' code"—they promoted the recipients to extravagant titles. The program brevetted contributors to such posts as lieutenant commander, judge, colonel, general.

"EVERY MAN A KING"  
HUEY LONG

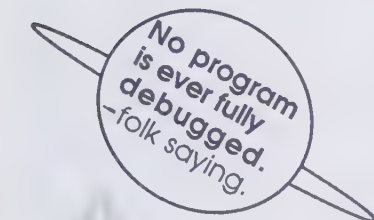
A program is like a nose; Sometimes it runs, sometimes it blows.

Attributed to  
Howard Rose



## BUG OR FEATURE?

There's an old joke in the computer world: "It's not a bug, it's a feature!" Meaning that a program's quirk, seen in a friendly light, becomes something you can pretend was supposed to be there.



Supposedly a big NASA rocket failed in takeoff once because of a misplaced dollar sign in a program.

Even if you don't write programs yourself, you may have to work with people who do. Getting what you want from them is a vital and delicate business called—

# PROGRAM NEGOTIATION

A very important kind of discussion takes place between people who want computer programs, but can't write them, and people who can write them, but don't want to. Or, that is, who don't want to get caught having to do a lot of unnecessary work if it could be done more simply.

Program negotiation, then, is where the "customer"—he actually may be the boss—says, "I want a program that will do so-and-so," and the programmer says, "I'd rather do it this way."

In a series of requests and counter-offers the customer explains what he wants and the programmer explains why he would rather do it a different way. *It is essential for both sides to make themselves completely clear.* Often the customer thinks he wants one thing but would be quite satisfied with another that is much easier to program. Often the programmer can make helpful suggestions of better ways to do it that will be easier for him.

Very bad things can happen if program negotiation is not done carefully and honestly enough. The programmer can misunderstand and create something that was not wanted. Or the customer can carelessly misstate himself and ask for the wrong thing. Or worst of all, the programmer can deliberately mishear and do something different, saying, "There, that's what you wanted," as he hands over something that isn't what was really asked for. And the poor customer may even believe it (see "Cybercrud," p. 27).

Program negotiation should be more widely acknowledged as a difficult and painful business. It is exhausting and fraught with stress; people (on both sides) get all kinds of psychosomatic symptoms (like abdominal pains, tics and chills). The fact that people's careers often depend on the outcome makes the atmosphere worse, rather than fostering the thorough and sympathetic cooperation which is essential.

If there is one thing that laymen in business should be taught about computing, this is it.

Another reason to know something about programming is to deal with computer people who want to sell you an outlook and the programs that go with it. It's important to ask the most intelligent,

## USEFUL, AND POSSIBLY EMBARRASSING QUESTIONS

If the Computer Priests start to pick on you, here are some helpful phrases that will give you strength.

I do not want to give the impression that the Guardians of the Machine are always bad guys. Nevertheless, sad to relate, they are not always good guys. (See "Computer Center Politics," p. 161.)

Like everyone out to bolster his position, including the plumber and the electrician, the computerman has learned how easy it is to intimidate the layman.

Now, these people are often right. But if you have reason to question the way things are done, whether you're a member of the same corporation, a consumer advocate or whatever, you are probably entitled to straight answers that will help settle the matter honestly, without put-downs. *Any honest man will agree.*

Now, these helpful questions, honestly answered, may elicit long mysterious answers. *Be patient and confident.* Write down what's said and sit down with glossaries and/or friends until you understand the answer. Then you can ask more questions.

I am not inviting the reader to make trouble flippantly. I am suggesting that many people have a right to know which has not been exercised, and there may be some discomfort at first.

## HOW DOES IT WORK?

(This question may have to be backed up as follows: "There are no computer systems whose workings cannot be clearly described to someone who understands the basics. I INSIST THAT YOU MAKE A SINCERE ATTEMPT.")

WHY DO YOU HAVE TO CLAIM IT HAS TO BE THIS WAY?

(SPEAK MORE SLOWLY, PLEASE.)

WHAT IS THE DATA STRUCTURE?

COULD YOU EXPLAIN THAT IN TERMS OF THE DATA STRUCTURE?

WHO DESIGNED THIS DATA STRUCTURE? And can I talk to him?

WHAT IS THE ALGORITHM?

WHO IS THE PROGRAMMER? And can I talk to him?

WHY DO WE HAVE TO USE A CAN-NED PROGRAM FOR THIS?

WHY IS THE INPUT LANGUAGE SO COMPLICATED?

WHY DO WE NEED CARDS? WHY CAN'T PEOPLE TYPE IN THEIR OWN INPUT?

WHY NOT HAVE A SIMPLE-MINDED FRONT END THAT LETS USERS CONTROL IT THEMSELVES?

WHY HAVE FORMS TO FILL OUT? WHY NOT HAVE A DIALOGUE FRONT-END ON A MINI?

WHY CAN'T IT BE ON-LINE? And on CRT terminals (see p. DM 11)?

WHY DOES IT HAVE TO BE THAT BRAND OF COMPUTER?

WHY NOT GET A SYSTEM WITH LESS OVERHEAD?

WHY SHOULD ALL COMPUTER OPERATIONS BE CENTRALIZED?

DON'T THEY GET IN EACH OTHER'S WAY?

WHY DOES IT ALL HAVE TO BE ON ONE COMPUTER?

WHY NOT PUT PART OF IT ON A DEDICATED MINI?

WHY CAN'T WE DO THIS PARTICULAR THING ALL ON A MINI?

WOULDN'T IT COST LESS IF WE GOT A MINICOMPUTER FOR THIS TASK?

WHY CAN'T THIS BE PROGRAMMED IN SOME LANGUAGE LIKE BASIC?

YOU KNOW AND I KNOW THAT COMPUTERS DON'T HAVE TO WORK THAT WAY. WHY DO YOU CHOOSE TO DO IT THAT WAY?

If these suggestions seem unnecessarily contentious, it is because some of these guys like to pick on people, and you may have to be ready. And you may need all the support you can get, if say, you take a stand like one of these:

"If the information is in there, I don't see why we can't get it out."

"You have no right to ask questions like this, and if the program requires it, change the program."

Remember, ILLEGITIMIS NON CARBORUNDUM (don't let the bastards grind you down).

According to Forbes magazine, the U.S. Patent Office, which ought to know something about technology, spent \$40 million on an inappropriate database system that takes longer to do a search than a person searching the old way. Why? "The instructions given to the database contractor weren't explicit enough." ("A Complete Fiasco," 17 June, 1985.)

## THE MEETING OF THE MINDS

### The Customer, Naive Advocate or Chump

I don't see why since it's a computer...

These are not details that concern me...

These are just technical issues...

I mean a computer can do all these things, can't it?

### The "Expert"

What you've gotta understand is that there are problems involved...

It can't be that way...

Leave it to me, it'll be just what you want...

**Comeuppance:** the customer will get what he deserves.

**Moral:** if you want something, you'd better damn well negotiate it at the detailed level.



It's awfully easy to fool people with simple words, let alone buffalo them with weird technical-sounding gab. The thing about tech talk is that it can really be applied to any area. The trick lies in the arrangement of boxcar adjective nouns, and in the vague use of windy terms that have connotations in some particular technical area—say, the space program.

Just consider. We might call a common or garden spade—

A PERSONALIZED EARTH-MOVING EQUIPMENT MODULE

A MINERALOGICAL MINI-TRANSPORT

A PERSONALIZED STRATEGIC TELLURIAN COMMAND AND CONTROL MODULE

AN AIR-TO-GROUND INTERFACE CONTOUR ADJUSTMENT PROBE

A LEVERAGED TACTILE-FEEDBACK GEOMASS DELIVERY SYSTEM

A MAN-MACHINE ENERGY-TO-STRUCTURE CONVERTER

A ONE-TO-ONE INDIVIDUALIZED GEOPHYSICAL RESTRUCTURIZER

A PORTABLE UNITIZED EARTH-WORK SYNTHESIS SYSTEM

AN ENTRENCHING TOOL

A ZERO-SUM DIRT LEVEL ADJUSTER

A FEEDBACK-ORIENTED CONTOUR MANAGEMENT PROBE AND DIGGING SYSTEM

A GRADIENT DISEQUILIBRATOR

A MASS DISTRIBUTION NEGENTROPRIZER

**451. A DIG-IT-ALL SYSTEM**

or,

AN EXTRA TERRESTRIAL TRANSPORT MECHANISM

Spades, not words, should be used for shovelling. But words should help us unearth the truth.

## GUIDELINES FOR WRITERS AND SPOKESMEN

The public is thoroughly confused about computers, and the press and publicists are scarcely free from blame. IT'S TIME FOR EXPLANATIONS. People want to know what computer systems *really* do—no more of this “latest space-age technology” garbage. Mr. Businessman, Mr. Writer, are you man enough to start telling it straight?

The computer priesthood, unfortunately, often wants to awe people with, or unduly stress, the notion of the computer being involved in a particular thing at all. It is time for everybody to stop being impressed by this and get on with things. Don't just copy-edit what they give you. Nose around and really find out, then write it loud and clear.

These simple rules are my suggestions for bringing on more intelligent descriptions that will help enlighten the public by osmosis.

1. FIND OUT AND DESCRIBE THE FUNDAMENTAL APPROACH AND PHILOSOPHY OF THE PROGRAM. This can invariably be stated in three clear English sentences or less, but not necessarily by the person who created it. THIS IS WHAT WRITERS ARE FOR: it is your duty to probe until the matter has become clear.

### EXAMPLES.

“This chess-playing program evaluates possible moves in terms of various criteria for partial success, and makes the move which has the highest merit according to these ratings.”

“This music-composing program operates on a semi-random basis, screening possible notes for various kinds of attractiveness...”

“This archaeological cataloguing system keeps track of a variety of objective features of each artifact, plus information on where it was, including linkages indicating what other artifacts were near it.”

What or whose computer is used to do a thing is of almost no concern (unless it is one of unusual design, of which there are comparatively few). Not the make of the

computer, but the GENERAL IDEA OF HOW THE PROGRAM OPERATES, is the most important thing.

Of course, if you are being paid by a hardware manufacturer, you'll have to name the equipment over and over; but recognize that your *real* duty is public understanding, and put the facts across. (If you think it can't be done, read the splendid Kodak ads in the *Scientific American*.)

2. Keep gee-whizzing restricted to the description of a system's psychological effect on real people. (What impresses you may turn out to be old hat.)

3. Look for angles special to what you're reporting. Pursuing details is likely to bring up better story pegs and more human interest. Instead of saying “computer scientists” have done something, you might find something more interesting for your lead; how about “The unlikely team of a biophysicist and a teenage art student...” or—finding what's special—“Never before has this been done on a computer so small, the size of a portable typewriter (and having only some 4000 words of memory)...”

4. Attempt to find out how *else* computers are used in the particular area, and mention these to help orient the reader.

This goes against the exclusivist tendencies we all have when we want to balldoo something. It is a matter of conscience, an important one.

### 5. Questions to ask:

What are the premises of your program?

What if people turn out to need something else?

What could go wrong?

And most important: What is *that*?

### IMPORTANT DISTINCTIONS

It is only by clarifying distinctions that people are ever going to get anything straight.

6. Do not say “the computer” when you mean “the system” or “the program.”

7. Don't say a “malfunctioning computer” (hardware error) if the computer functioned as it was directed on an incorrect *program* (software error). (And remember that the best programmers make mistakes, so that a catastrophic bug in a system is no sign that it was programmed by an incompetent, only that it isn't finished.)

8. (A particular point about graphics. See flip side.) Don't say “TV screen” if a computer screen is not TV, i.e., 525 horizontal lines that you can see on the screen if you look for them. (See pp. DM 84-86 versus p. DM 22.) HOW ABOUT: “visual display screen”?—you can add, “on which the computer can draw moving lines,” or whatever else the particular system does.

9. Don't assume that your audience is computer-illiterate.

10. Don't assume that it can't *all* be said simply. Only lazy or hard-pressed writers are unclear.

11. Do not use cutesy-talk, particularly that which suggests that computers have an intrinsic character. By “cutesy” I mean sentences like “Scientists have recently taught a computer to play chess,” Mis-Leads like “What does a computer sound like?” (when talking about music constructed by a particular program in a particular way), and awe-struck descriptions like, “At last the Space Age has come to the real estate business...”

12. Do *not* use the garbage term “computerized,” unless there is a clear statement of where the computer is in the system, what the computer is doing and how. A “computerized traffic system,” for instance, could be any damn thing, but a “system of traffic lights under computer control, using various timing techniques still under development,” says something.

13. Don't put in clichés as fact, for example by the use of such terms as “mathematical” or “computer scientist” unless they really apply. Do not imply any mathematical character unless you know the system possesses it: many programs con-



tain no operations that can fairly be called mathematical. Similarly, a "computer scientist" is someone widely or deeply versed in computers or software, not just a programmer. (Anyway, if something has been programmed by an entomologist, it is probably more interesting to refer to him as an entomologist than as a "computer scientist.")

14. Do not refer to apparent intelligence of the computer (unless that is an intended feature of the program). Credit rather the ingenuity of the system's creator. Do *not* say "the clever computer." If anybody is clever it is the programmer or program designer, and if you think so, say so. These guys don't get the recognition they deserve.

15. Never, never say "teach the computer" as an elliptical way of saying "write computer programs." Programming means creating exact and specific plans that can be automatically followed by the equipment. To say "teach" when you mean "program" is like "persuading" a car instead of driving it, or making a toilet "cry" instead of flushing it. (There are systems, described on the flip side, which simulate intelligent processes and may thus be said to "learn" or "be taught.") But neither programming nor simulated learning should be described in a slipshod fashion that suggests the computer is some sort of trainable baby, puppy or demon.)

16. Do not imply that something is "the last word," unless you have checked that it is.

17. *Note that terminology carries a hidden viewpoint. Those terms they're pushing at you—do you know what they mean? Is the terminology perhaps guiding you to think the way they want you to think? What would be a different way of phrasing it that would be less sympathetic?*

## BIBLIOGRAPHY

Ernest Gowers, *Plain Words*.

This wonderful little book showed English civil servants "bureaucratic writing" was totally unnecessary. Its precepts—mainly concerned with calling a spade a spade (see p. 44)—transpose exactly to the computer world.

## THE IDIOTIC WORD "MICROCOMPUTER"

—Orwell would've liked it.

Computer Lib has been called "the first microcomputer book." An honor, I suppose, but I must repudiate the word "microcomputer," an idiotic term that has increased the stupidity and confusion abroad in the computer field. So-called "microcomputers" are no different, were never any different from regular computers.

"Micro-" is properly a prefix for things that deal with smallness, not for things that embody it. A microSCOPE looks at small things, a microTOME slices thinly. But the personal computer is in every way the equivalent of machines like the PDP-8 or PDP-11, and does just what they did.

The silly word "microcomputer" uses "micro-" to mean made with integrated circuits. But why? We don't say "microtelephone" or "micro-tape recorder," despite the teen-siness of their circuitry. The computer does not change its functions because of these microcircuits, any more than do the telephone and tape recorder.

There is no difference between the "microcomputer" of the seventies and the "minicomputer" of the sixties. It's on a desk? There were minicomputers on desks. It's for one individual to use? That started in 1960 (see "The Classic LINC," p. 85).

### WHAT IS THIS SYSTEM ABOUT?

Handy questions to size up what a computer is supposed to be doing:

- What data does it contain?
- Where is the data stored?
- What other data will it link up to?
- What information do you suppose can reasonably be derived from that?
- What are the key input and output devices?
- In what forms does information go in and out?
- What do you suppose they might want to know?

"...As sophisticated electronic computer can store and recall some 100 billion 'bits' of information..." TIME, 14 Jan '74, 50.

Piffle. That's the overall size of the memory, which is utterly independent of the sophistication or general power of the computer itself.

Trillion-bit memories are available, and you could put one on a machine as small as a PDP-8.

My point is that the so-called "microcomputer" is NO DIFFERENT FROM WHAT CAME BEFORE, EITHER IN FUNCTION OR SIZE. No, "microcomputer" is a marketing term, cleverly concocted to suggest inferiority without actual difference. Like certain racial terms.

And when people started loving computers and realizing computers were the opposite of what they'd thought—saw computers as potential instruments of freedom and artistry, rather than oppression and regimentation—they too had to call them something different. So the public adopted the dumb marketing term.

And the term "microcomputer" has led to inane terms like "megamini" and "micro-mainframe"—spuriously technical-sounding terms. And silliest, since nearly all computers will be made with integrated-circuit processors in the near future, that makes all computers "microcomputers." Of what use is such a word?

(There are certain measurable sizes of computers: word length, data path, cycle time, addressability, number of program followers. Those make sense.)

There are no words as foolish as "microcomputer" in other fields, because in any other field a comparable word would be too silly.

Note that referring to a computer as if it were a living creature is *not* cybercrud; to say that a program "looks at" a device, "tries to" effect a procedure, and "goes to sleep," are all colorful brief ways of describing what really happens. (See "Guidelines for Writers and Spokesmen," p. 44.)

### FICTIONS ABOUT WHAT COMPUTERS DO

Many people suppose there is nothing computers cannot do; some people, indeed, think there is nothing computers do not *already* do.

A couple of years ago, a leading picture magazine carried a piece about Stanford's Artificial Intelligence Laboratory, claiming that one "Shakey the Robot" had been developed to near-human intelligence and capabilities. This was pure bosh, since repudiated in the computer magazines, but a lot of people Out There in Readerland believed it. (See "The God-Builders," flip side.)

Once I had a long discussion with a somewhat wild-eyed young woman who believed that the government was monitoring her brain with computers. I think I persuaded her that even if this were feasible it would cost the government tens of thousands of dollars to do it, and that probably no existing government agency was that interested in her thoughts.

I'm not sure she was persuaded.

In the computer field, the same things are often called by different names (for instance, the IBM 1800, a fairly ordinary minicomputer, is called by them the "IBM 1800 Data Acquisition and Control System"), different things are often called by the same names, and things can be inside-out and upside-down versions of each other in extraordinary variety. (Indeed, computer people may find this book inside-out, which is okay with me. Life is a Klein bottle.)

Sorting things out, then, means having a few basic concepts clear in your mind, and knowing when you see examples and variations of them.



# IT'S ALL SOFTWARE

"Software" means programs—the plans and directions and scenarios, sequences and cascades of premeditated possibilities, that sit in machines and run them.

Much of what is either familiar or awesome in the computer world is actually not so much the hardware as the programs that make the hardware run—programs which can be extremely difficult to create.

Boxes fool us.

Most of what laymen and journalists think is *The Computer* is really *The Program*—software.

Video games are software.

The "new computer they've built to do X," described on the nightly news, is probably software running in an ordinary computer.

New medical equipment is software. The amazing CAT scanner that shows a slice of a patient's head: Yes, it uses special equipment, but the main part is an elaborate program that directs the X-ray beam and figures out what's where in the depicted slice.

The checkups and screwups of the Internal Revenue Service; the magic of image synthesis (see pp. DM 100-119); the magic of computer audio (see p. DM 60); and your very own bookkeeping system, if you ever manage to get it set up—all software.

Your payroll check, the "Star Wars" military proposal, image synthesis, CAT scanners that slice patients in pictures, the simulation of nuclear winter that has so changed our sense of the future—all these are complex programs whose complexity far outshadows the complexity of the equipment they run on.

**THE MYSTICAL COMPLEXITIES OF SOFTWARE GO ON WITHOUT END.**

If you get into building software you may never come out.

There is no limit to the possible complication, no limit to the strange ideas, no limit to the fascination.

The lure of weaving strange ideas and happenings, fashioning them first within

your fevered brain, and then making them happen in the world, takes you to realms of abstract mysticism that are still only being discovered. Enter this land of spell-casting at your own risk, for doing magic is addictive.

## BIBLIOGRAPHY

Alan Kay's monumental software piece in the September 1984 *Scientific American*.

## "SOFTWARE PROGRAMS"

Software originally meant computer programs, but then the publishers and recording folks heard the word and decided it included their stuff as well—the contents of books and records and videotapes. Thus books and records and videotapes became called "software" as well—so software now merely means contents. And records, and movies, and magazines, and who knows what else.

So how now do we refer to computer programs, to distinguish them from the rest of "software"?

Well, what was originally called software are now widely called "software programs."

Don't let the videotape people hear this one, though, or they may decide it applies to them too. What term would we have to go to next—

**PROGRAMMABLY  
PROGRAMMED SOFTWARE  
PROGRAMS?**

*The French word for software, "logiciel," perfectly fuses two French obsessions: la logique, and le ciel.*

# GET ONE

This is advice to individuals or small companies who are buying one computer.

## WHY BUY?

There are various reasons to buy a computer. All are valid.

1. **Lust.** Somehow you just have to have it; you "just want one."
2. You want to learn about computers.
3. You want to learn to program.
4. **Specific need.** You have a particular, focused use; there are specific programs you want to run.

5. **General need.** You want to organize your life or your company.

These are different needs requiring different advice.

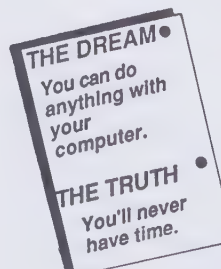
1. If your principal motivation is computer lust, you will satisfy it in your own way.

When you choose a computer, you are effectively choosing your next several years.

Choosing a computer system is choosing a Maypole—an all-involving whirlwind of concerns, cutting you off from various alternatives. Like a car, dog, boat or airplane, it tilts your whole life.

The computer you get becomes your world, the Maypole you dance around, your tool, your toy, your lover, your home, the center of your computer life.

**LOVE IT, HATE IT, BUT LEARN IT AND HACK IT AND MAKE IT YOUR OWN.**



**A SERIOUS USER IS SOMEONE WHO CARES ABOUT THE RESULT**

or

**INTENDS TO SAVE THE PRODUCTS OF HIS OR HER WORK OR PLAY.**

2. If you want to learn about computers, get a machine, and a few programs you think you want to live with. Settle down with them. And do some programming.

3. If you want to learn to program, see p. 40.

4. If you have a specific need, your problem is clear and you know what to do about it.

5. The case of general need is the worst. To organize your company is immensely bothersome and requires great diligence to keep it maintained; to organize your life as an individual is almost impossible, unless you have several secretaries or an extremely simple life.

### A COMPUTER IS A SOFTWARE ENVIRONMENT

In deciding what machine to get, the real question is the kind of software world you intend to live in. What basic computer hardware you get is in a way inconsequential, except for size or speed. That's because the different popular computer systems are different software environ-

ments. The world you live in is the operating system and the style of operation—icons and fonts on the Mac, function keys on the PC, and so on. (Indeed, if you get a UNIX system, the processor is mostly irrelevant, since the software is pretty much the same whatever it's running on.)

This is quite the opposite of the technoids' view, obsessed with chips and speeds, and hardly noticing the software. Yet it is the software you live in and face on the screen, hour after hour and month after month.

### RULES FOR BUYING A COMPUTER

1. It's an **emotional decision**, like buying a car or getting married. (Some people pretend to do these things rationally, but that's the masquerade of **their** emotions.) The one that tugs at your heartstrings may be the right one.

This is the overriding rule. However, there are a few other things to consider if your feelings have not already run away with you:

2. **GET A STANDARD MACHINE.** Do **not** get a nonstandard machine. (See

box, "The Standard Computers.") Otherwise you won't be able to get software or accessories for it. Would you buy a 1949 Studebaker? (At least you can get gas for a 1949 Studebaker.)

3. **GET THE MOST SERIOUS MACHINE YOU CAN AFFORD, AND THE MOST EXPANDABLE.** There is no proper size for a "home" or "personal" computer. As with high fidelity, it is a straight continuum from K-Mart up to the highest professional setup, which does all the same things but better.

4. (HOWEVER, if you can get **particularly** good support from a nearby dealer, consider his machine instead of one from across town.)

5. **CONSIDER A PORTABLE.** You will probably be moving your machines much more often than you expected to. (Note that by "portable" I mean **luggable**, not laptop.)

Above all,

6. **BE SURE TO GET AT LEAST TWO DISK DRIVES.** Anything less is a severe hardship.

### PERSONAL VS. BUSINESS COMPUTERS

There's no more difference between a personal and business computer than there is between a personal and business car.

### PERSONAL VS. BUSINESS USES

There is no real difference between what an individual needs and what a corporation needs—after all, a corporation is a legal person, isn't she?

Individuals need everything that companies need, and **more**. Personal software needs to do accounts payable and receivable, financial projection and taxes and database just like corporate software. But individuals' needs are even broader, extending to art, leisure, curiosity, self-improvement, mischief and play. So what corporations need is a **subset** of what individuals need.

And the **problems** are the same, for corporation or individual. They are the conceptual problems of mapping your needs to what's available; the difficulties of understanding how to use it; the difficulties of fitting it in to your personal or corporate life—intellectually, emo-

The saddest rule of all: the easiest stuff costs the most.

There is no upper limit on what you will want.

This includes speed, amount of fast memory (memory is like money—you can't have too much), amount of disk memory, graphics, audio output.

The personal computers in the year 2000 will make our megabyte machines seem puny. They may include full orchestras and real-time motion-picture generation.



You expect your life and your time to become organized. Instead, computers take your time and convert your life.

It's like marriage. When you dream of marriage, you tend to dwell on its imagined benefits; the fantasy does not include the effort, demands, complications and sacrifices that marriage actually entails.

Computers, unlike spouses, become ever cheaper and more reliable. But so far they're far more demanding and less gratifying, and a bigger adjustment.

*Think of buying a computer as like buying a car. A car just moves your body; your computer, though, is the chariot of your mind, carrying it through the whole universe. How much is your mind worth to you?*

*Spend at least as much on your computer as your car. Or spend proportionally less on your car in order to obtain real trance portation.*

*What is the distinction between practicality and fun? I guess that depends, in part, on what you do for a living (and whether you enjoy it).*

### UH OH, MAINTENANCE

A practical problem of immense importance is "maintenance," meaning repair and upkeep of computers and their accessories. Lots of guys in Boston and L.A. are having fun making computers, but here you are stuck in Squeedunk and it doesn't work anymore.

Trying to find people who will fix these things on a stable basis is a great problem.

You can sign a "maintenance contract" with the manufacturer, which is sort of like breakdown insurance: whatever happens he'll fix. Eventually. If you own equipment from different manufacturers, though, it's worse: each manufacturer will only contract to fix his own equipment. (And remember, interfaces have to be maintained too.)

This is the biggest point in favor of IBM. Their maintenance is superb. There's also something called third-party maintenance: companies who'll contract to keep all your hardware working. RCA and Raytheon are into that.



tionally, sociologically—getting it to Fit In and Catch On.

### WHICH WORLD?

*Only buy a standard serious computer. That rules out perhaps half the computers available.*

*Avoid the puppies (Commodore, Apple II) because they are dead-end machines. The action has moved on.*

*This includes also laptops: a laptop computer is like a dishpan on a camping trip—a partial facility that's invigorating for a while, but you want a shower when you get home. You need a full machine: a laptop without a mothership is a hardship.*

*That leaves (as of mid-1987) three standard serious computer families under*

*the Sun: the PC (now as low as \$500), the Macintosh (now as high as \$7000), and UNIX systems.*

*Lots of accessories and software are available for all of these.*

*The PC tends to be bought by conventional and straitlaced people; the Macintosh tends to be bought by bright eccentrics. (However, bright eccentrics are building the future.) And UNIX systems are for hackers, serious computer students, and companies that want to develop their own software.*

*At the high end (above \$10,000), the workstation machines from Sun Microsystems are the current standard, and above that, the Iris graphic workstations of the flight-simulator class (see p. DM 113). But I assume you don't want to spend that much.*

### THE PC: A Low-Cost Standard

By a "PC" I mean any computer that mimics the IBM PC well enough to run Bruce Artwick's "Flight Simulator," accepts standard PC accessory boards, and has a keyboard rather like the standard PC keyboard.

Such machines now cost as little as \$500, and can be dandied up with extraordinary graphics and audio boards, as well as a host of other accessories and mass memories.

The PC family is currently THE family of workhorse computers for all kinds of add-ons. The PC is just a straightforward casing around the Intel 8088, with an open architecture—a huge policy change for IBM—allowing everybody to make attachments.

BASICALLY the IBM PC can be thought of as the Apple II three years later.

(The Apple II came out in '78, the PC in '81.) But the PC makes the Apple II look like a rollerskate next to a bicycle, what with more room on the disks, a fuller screen, a bigger keyboard on a long cord, and up to 640k of memory. But the flavor of complication is distinctively IBM's, a distinct change of atmospherics from the Apple.

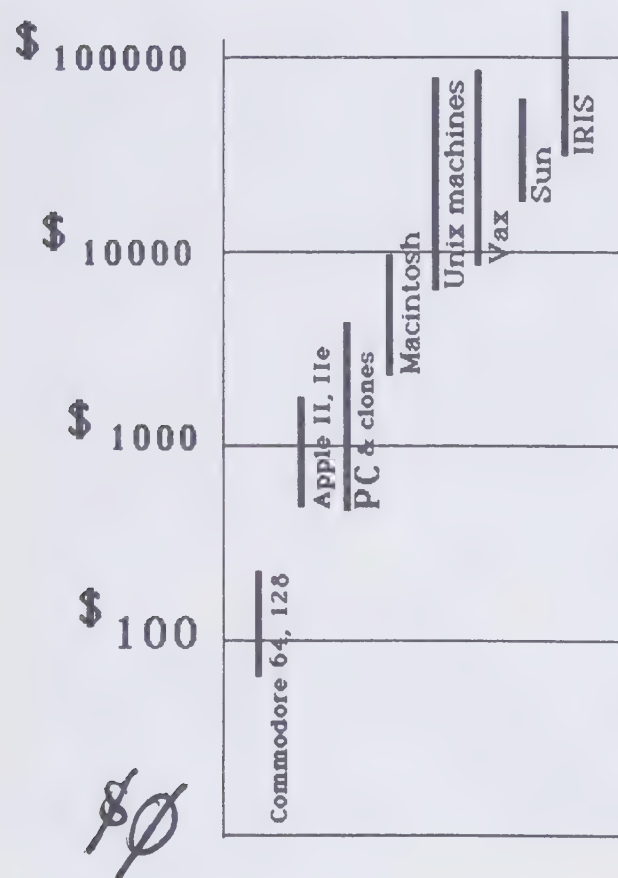
As a cheap, reliable, worldwide standard, the PC is remarkable. For inconsistent, ill-thought-out complications it is infuriating. But you can get lots of stuff for it, do lots of programming on it, and get lost in it for years.

While the PC is standard, upgrade paths for it are not. (See next page.)

Two particularly nice PC compatibles of special interest:

The Sanyo color portable (model MBC 775) has a built-in color monitor, takes

## STANDARD PERSONAL COMPUTERS, 1987



**THERE'S NOW AN OFFICIAL BLUE-BOOK! Don't expect much trade-in value, though, because the prices are going down so fast.**

### USED COMPUTERS

While in principle there would seem to be every advantage in buying used computers, there are certain drawbacks. Service is the main one: The manufacturer is not very helpful about fixing discontinued machines, and you may have to know how to do it yourself. Even with machines still available, you may have trouble getting onto a service contract from the manufacturer, since it "may have been mistreated." (American Used Computer in Boston will usually guarantee that its merchandise *will* be accepted back into manufacturer's contract service.) A final drawback is price: A popular machine may cost as much used as new, since they're saving you the waiting period.

It's kind of unfortunate: Otherwise usable machines get wasted. (But here's waste for you: Certain well-known laboratories, owned by a profit-making monopoly, *smash* their used computers if nobody wants them within the lab. They claim they can't resell them because they would be "competing" with the manufacturers. I wish the conservationists would get on that one.)

### "IT'S NEVER THE RIGHT TIME TO BUY."

Yup, you can always do better in six months. You just have to trade that against the lost time in your life.

*The Macintosh has been called "the best of all available worlds."*

full accessory boards and has a full standard keyboard.

The Zenith 181 portable runs on batteries and has a lovely big blue screen—but when you get it home you can hook it up to a monitor. (Warning: not all software can be moved to its small disks.)

## BIBLIOGRAPHY

Peter Norton, *Inside the IBM PC: Access to Advanced Features and Programming*. Simon & Schuster, 1985.

His Norton Utilities are also a must for when you have MS-DOS problems.

Books on MS-DOS are varied and necessary.

The PS/2, from IBM, is a half-open system which IBM intends to control and keep expensive. If you commit to it, you are climbing into the soup pot.

## If You Love It, Get It:

### THE MACINTOSH:

#### Your Yuppie-Craftsy Computer

It is easy to start using the Macintosh: pull down menus like windowshades, point with the mouse and click. (Getting good with particular programs can still take a long time, though.)

PEOPLE SEE THE MACINTOSH IN DIFFERENT WAYS. To corporate America it's for "desktop publishing" (see p. DM 28).

Some think it's a graphics machine, and indeed it comes with funky ways of visualizing the disk and programs, and neat fonts you can print out.

At many colleges it's a standard new way of doing papers.

To High Hackerdom it's a serious clean machine with a lot of goodies built in.

To many it's just an amiable companion.

The standard interface, more or less enforced by Apple, is only part of the story. Built into it are programs for a lot of fast graphics (the excellent "Quickdraw" routines by Bill Atkinson), and the fabled Toolkit of functions available to programmers.

Predefined data formats exist for text, bit-map graphics ("MacPaint format"), compound graphics ("MacDraw format"), and page layout.

In principle the presence of all these functions should make it simple to program. In fact, programming that uses all this stuff—"the application level"—is a bear. But it means that well-made applications can do a great deal under fairly simple control.

The Macintosh has the most overtly emotional following of any computer, typ-

ically young, well-to-do and individualistic. (There was once a Macintosh Convention and Ski Weekend.) Anyway, if you find yourself irresistibly in love with the Mac, don't fight it; get one.

The original Macintosh has a small screen, but you can have big additional screens attached, or get a Mac II, which offers big screens and color.

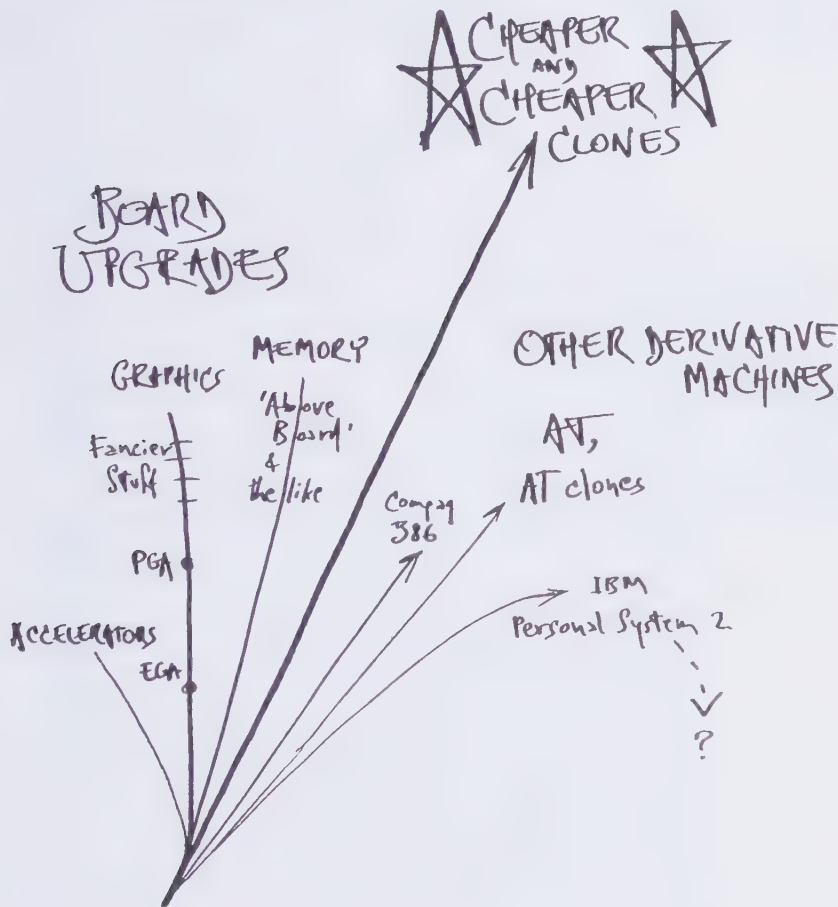
Portable Macintoshes—which are actually repackagings of the same machines—are available from various firms; the leader is Colby Computer in Mountainview, CA.

## BIBLIOGRAPHY

Inside Macintosh, Apple Computer, 3 vol. Addison-Wesley, 1985.

Stephen Chernicoff, *Macintosh Revealed, Vol I: Unlocking the Toolbox*, and *Macintosh Revealed, Vol. II: Programming with the Toolbox*. Hayden, 1985.

# UPGRADES OF THE PC



The Macintosh is most often compared with the PC. For a different perspective, however, see "Man, Bytes, Dog," in *The New Yorker*, 2 July '84, for a thoughtful, probing comparison between the Macintosh and the cairn terrier, a small user-friendly dog. The author points out that the cairn terrier doesn't flicker when you look at it and wags its tail.



ROCKFORD BUSINESS CARDS

The hero of "The Rockford Files" would print up new business cards, in minutes, as he needed them. The Macintosh is great for that: The standard screen is exactly the right ratio. Just typeset on the screen, print out, and reduce by 76% and then 67% on card stock. I know a guy who frequently does this on the way to the airport. (The hard part is trimming it neatly to 2" x 3 1/2" with the copy correctly centered.



## WHY NOT UNIX?

A number of excellent UNIX machines are now available under or near \$10,000; and so you may very well ask: Why not a UNIX machine as a personal computer? (Note that to run full UNIX, there must be a kernel mode on the chip, so the PC and original Mac won't do.)

Now, UNIX machines are very much alike: they all run the UNIX operating system (see p. 104), which is the hub of all the programs you run. UNIX is an operating system for tying small programs together cleanly, efficiently and rationally. UNIX interpenetrates its programs to an extent that, say, CP/M and MS-DOS do not. UNIX programs may constantly use pipes and forking, for instance, which can't occur under other operating systems.

UNIX is written in C, so UNIX systems all run at least the C language, one of the supreme favorites among hackers. With C as the language and UNIX as the environment, the hardware identity of the machine virtually disappears.

What distinguishes UNIX systems is what other software they come with.

The only problem is: personal-type software is not yet widespread in the UNIX world. Software prices are higher and programs are meant for corporations. There are integrated office packages becoming available under UNIX; but expect them to be priced at corporate levels. A great many scientific and business applications have been put on UNIX systems, and many more will be; but the games and \$120 business packages aren't there yet.

So UNIX is not ready to come home with you and lie by the fireplace the way

your Mac and PC will. But if you want to be a hacker, get UNIX if you can.

### Products Of Interest:

Symmetric Computer Systems, San Jose, CA, sells a portable VAX with UNIX, C, APL, Logo, Prolog and a thousand programs (whatever they may be) for \$5000. You'll need a terminal.

The Mac II can be gotten with UNIX. Unfortunately, in UNIX mode it doesn't run other Macintosh programs; it can only be one or the other at a time.

### PERIPHERALS FOR YOUR MINI

Some kinds of peripheral devices, or computer accessories, are always necessary. Only through peripherals can you look at or hear results of what the computer does, store quantities of information, print stuff out and whatnot.

Trying to print lists of available stuff here is hopeless. There are thousands of peripherals from hundreds of manufacturers.

### PERIPHERALIA SAMPLER

Much of the thrill of computerdom is the extraordinary range of possible actions that are possible from your system. These require accessories, or peripherals.

All the incredible toys—ahem, peripherals—developed by the computer field in the last thirty years can now be yours. For a price.

Your computer can become all sorts of things. An astonishing variety of stuff is already available for your Mac or PC, and more comes on the market every week.

GRAPHICS: TV cameras or conversion boards to bring in pictures. Paint sys-

Every separate part you want for your computer system exists somewhere, seemingly. But you probably can't afford it, let alone the time and technical effort needed to tie it all together.

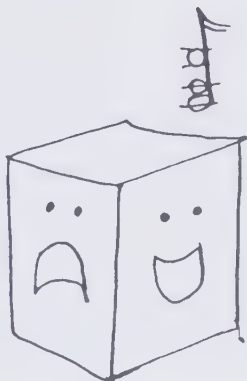
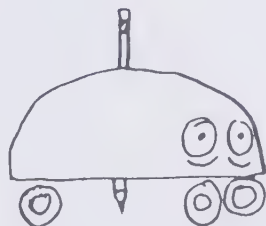


### BRaille

No joke here. People are still making Braille copies of things by hand. But the way to do it is by computer: the machine can punch out new copies of whatever's stored in it, repeatedly.

Much equipment is now available in this area; organizations for the blind keep up with these things. Note that they may have use for volunteer input-typists who have their own computers.

Everybody talks about voice recognition, but real voice recognition is still a long way away; so is Optical Character Recognition for handwriting, although OCR has at last arrived for cleanly typed or printed text.



### YOUR TURTLE AND MUSIC BOX

Surely nobody can resist the peripherals offered by General Turtle, Inc., 545 Technology Square, Cambridge, Massachusetts 02139.

The Turtle is a sort of casserole on wheels that takes a pencil down the middle. Attached to your computer, it can be programmed to ramble around drawing pictures, or just do wheelies on the parquet. \$800.

Then the Music Box is \$600. It sings in four voices, enough for a lot of Vivaldi, does five octaves and looks to the computer like a Teletype. They will play you samples on the phone (617/661-3773).

For either of these you need a Controller (\$1300).

The company is defunct. You can't get the physical turtle any more; but its spirit lives on wherever Logo is used. For the little drawing pip you send around the screen is called a turtle, in homage of the old rolling pot. The original turtle was great, though; kids loved to beep its horn (by program).

Musical equipment may be had in profusion. Music can be done all by software now, or through the MIDI interface to synthesizers (see p. DM 62.)

WHOEVER HAS THE MOST TOYS WHEN HE DIES WINS. AMEN

tems to create pictures. Soon: image-synthesis movie-making systems. (See Dream Machines side in general.)

**AUDIO:** music synthesizers, speech generators, speech recognizers (not yet very capable).

**POINT OF SALE:** cash register drawers; bar-code wands and bar-code printers. (Also a coin-slot device to make your computer a video game.)

**TELEPHONING ACCESSORIES.** A number of packages are now available to make the PC operate as an answering machine, **STORING THE INCOMING MESSAGES ON DISK.** These include SAM, "the Smart Answering Machine" from Dialectron of Mountain View, CA; the "phone slave" package (from Active

Voice in Seattle); and "Watson" (PC board and software from Natural Microsystems Corporation, Natick, MA), which can act as answering machine, out-caller and reply solicitor, and dictation machine.

You could in principle hook up hi-fi equipment, though to the best of my knowledge the makers of programmable hi-fi stuff won't sell you the components yet.

You like those cheery message boards with moving signs? Put one in—or on—your car! (**SIGN LINE**, from Universal, Cedar Rapids, IA.) Tell the other drivers **exactly** what you think.

Various sorts of arms and mechanical outputs can be gotten; a magazine called Robotics Age reviews these, as does Byte. (See "Robotics," p. DM 140.)

## THE VITAL IMPORTANCE OF STANDARDIZATION

This is the place to tell you that you should get only standard stuff unless you know **exactly** what you're doing. Standard things break far less frequently, and can be **fixed**; they can be **plugged into** each other, or, in the case of a standard computer (such as the PC and the Mac), you can get more—and cheaper—programs to run on them. (Standard **software**, awful as it usually is, can accept data from, and move data into, more places—more **other** software—than the other programs individually.)

Standards are **made in the marketplace**. Nobody controls them. The fancy professional Standards Committees, and the National Bureau of Stan-

dards (NBS), and the International Standards Organization (ISO), basically just make suggestions.

What gets standard depends on a lot of factors, such as Getting It On the Market First, Company Clout, and (last) quality. (Time was when IBM could dictate standards, but now its products—especially in the personal computer market—compete with the others just like products from other big companies, such as Apple and Hitachi.

Stay away from off-standard hardware. Just stay away. Even among the standards there are so many darn choices.

## The Worm Turns

The most capacious storage available is presently in optical disks that write permanently but have so much space you hardly mind. These are called **WORM drives** (Write Once, Read Many). Kodak is offering trillion-bit models.

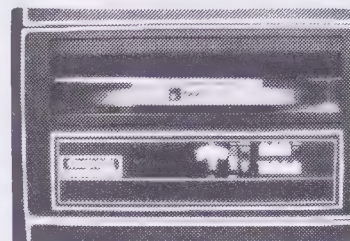
The only problem: the space gets used up faster than you expect by conventional software. The solution: new forms of indexing (see "Project Xanadu," p. DM 141).

You'll never know what you'll see next. In 1969 one firm announced a "high-density read-only memory device" which anyone could see was a plain 45 RPM phonograph—but with digital electronics. And it made sense. But it doesn't seem to have caught on.

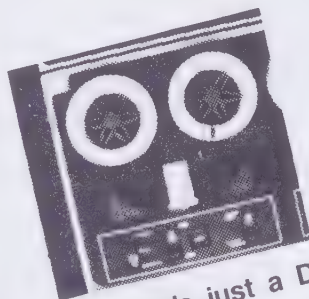
Now there's a big push behind another converted audio disk, called "CD-ROM"—that is, audio Compact Disk, to be sold holding digital information (ROM, Read Only Memory).

Hasn't caught on yet, and read-write disks at comparably low cost are around the corner. Stay tuned.

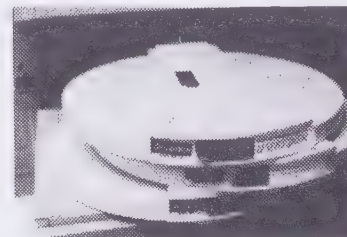
**BIG SCREENS** are available to extraordinary resolutions now for the PC, Mac and Sun. (I've heard of a 2 million pixel screen for the PC.) There is probably an upper limit but we're not there yet; see p. DM 22.



- Disk drive for the 11. Most such devices go at 30 spins a second, or 1800 rpm. The heads that read and write information are on moving arms that have to be positioned on the different tracks. (Some disks have a head for every track, which costs more.) If you have disk drives (\$5500 each) you need a controller (\$5500). Sigh.



Scared? It's just a DEC-tape drive, upside down.



- Disk cartridges for this model disk drive. The brown-coated disk itself is hidden in the plastic case. Nevertheless, they sometimes get scratched or break. A disk costs \$75 and holds up to 2,400,000 characters of information (1.2 million PDP-11 words, which are 16 bits each).



## Beigers Can't Be Choosers

Those of us who hate the color beige have a tough time. Most personal computers and printers are in one beige or another.

The time of kandy-kolor paisley tinsel-flake plaid padded computers and accessories with racing stripes, likenesses of fashionable rock stars and comic strip characters and movie monsters, smiling faces and hearts and flowers has not arrived. But soon, soon.

**FLASH!** For (only!) \$200, Aesthetics, of Palo Alto CA, will paint your Macintosh any color you choose!

You can get ribbons in colors for various dot-matrix printers—and yes, the printout irons onto a T-shirt—from Diversions, Inc., Belmont, CA. Similarly from Frontrunner Computer Industries, Reno, Nevada, which offers not only T-shirt ribbons in five colors, but film ribbons in silver, purple and other nifty colors.

### RULES ABOUT SOFTWARE

(for individuals and companies)

1. Don't try to program it yourself unless you really know what you're doing.
2. Don't assume that any two programs will tie together reasonably.
3. Don't assume that a program which supposedly does what you want (functionality) does it the way you want (virtuality; see p. DM 68).
4. Be suspicious of all that is said in the software field.

There is little agreement on the boundaries of programs: for instance, where "word processing" leaves off and other things begin. And it's like that in every area. One program may have far different stuff in it, stretching in different directions, from another program of the same general intent.

## What is "Software" Really?

**PROGRAMS ALREADY WRITTEN,  
PROGRAMS YOU'RE STUCK WITH,  
PROGRAMS YOU COULD BUY BUT CAN'T AFFORD,  
PROGRAMS YOU COULD BUILD BUT DON'T HAVE  
TIME TO.**

**THE CHANGEABLE PART;  
WHAT YOU HAVE TO PAY EXTRA FOR;  
THE EXPENSIVE PART THAT'S HARD TO CHANGE (as  
distinct from hardware—a paradox).**

## PRINTERS!

You can get printers costing from a hundred dollars to a hundred thousand. Dot-matrix printers are incredibly reliable, laser printers are very pretty, and some of them come in colors. The software problems can be fierce.

### ROLL THE DRUMS...

IBM has announced a laser printer, the model 3800, which prints at 13,360 lines per minute. Like most of their printing machines, it's good. It is basically an electrostatic drum copier, like the original Xerox 914, on which the patents have run out. (Even Toshibafax now makes one.)

Anyway, this spectacular beast writes with a scanning laser on the electrostatic drum. But IBM has cleverly found the way around the problem of spoilage of the drum surface: Instead of just a polished metal surface, it's a renewable surface, which is itself changed automatically when required for new images.

Moreover, you can have up to 18 type fonts, defined in 18x24 dot matrices. (THESE ARE KEPT ON A FLOPPY DISK, AND UP TO FOUR MAY BE CURRENT AT ANY ONE TIME.) Fonts are user-designable.

A flash-projector can put business forms on the drum.

Now for the bad news: base price is \$310,000, plus extensive extra charges. Also it doesn't make carbon copies, and needs a new box of paper spliced on the end every 20 minutes.

(Canon, of Japan, has put out a more modest laser printer.)

Canon's more modest laser printers have taken over most of the market, putting laser printers throughout corporate and upscale America, for as low as \$2000.

**Suggestion.** Before buying a printer, ask the clerk to show you how to change the ribbon. It might change your mind.

### SOFTWARE

Computer programs, or "software," used to come free with the computer. But IBM turned around and "unbundled," meaning you had to buy it separately, and there has been some following of this example. However, for users who are buying a computer with some canned program for a particular purpose, prices are obviously for the whole package; it's

people who use the same computer for a lot of different things that have to pay for individual programs.

There are many small software companies. For the cost of a letterhead anyone can start one; the question is whether he has anything special to sell. Some people whomp up programs on their own which turn out to be quite useful.

Obviously, to create big systems for intricate management purposes requires a great deal more effort. Traditionally these are done by vast programmer teams working in COBOL or the like, constantly fighting with monitor programs and chewing up millions of dollars.

*We all want the same thing (though in different styles) and it's probably possible; but for the near future, resign yourself to*  
**THE HEARTBREAK OF  
SOFTWARE: THE VISION AND THE  
MESS**

Everybody knows what a computer is for.

It's for GETTING ORGANIZED.

It's for KEEPING TRACK OF THINGS.

It's for KEEPING TRACK OF FACTS.

It's for KEEPING TRACK OF DETAILS.

It's for KEEPING TRACK OF STUFF  
YOU'RE IN THE MIDDLE OF.

It's for TYING THINGS TOGETHER.

It's to PROVIDE YOU WITH AN  
OVERVIEW.

It's for SIMPLIFYING THE WAY YOU  
DO THINGS.

Most people would agree on these objectives, whether they are running a company or contemplating a computer for their personal use; note that corporate and personal uses are really just the same.

But what is the reality? A hideous tangle of complexity, incompatibility, stuff you have to keep track of about the computer; endless details to learn and try to fit together.

You think it'll all be sensibly organized, and indeed it *could* be. But instead what you'll find is many pieces that don't fit together; inane categories and groupings, stupid layouts of data, stupid layouts on screen, confusing arrangements of choices, and all of these tied up in the minds of programmers who don't think like you.

Today's market software is in general tough to install, hard to use, dumb, incompatible, artificially narrow, twisted in concept, stupid in appearance, silly in restriction, opaque in explanation and mystifying as to the mental orientation of its perpetrators. All this will change; whether by slow steps, as is now happening, or huge leaps, as may happen—we will see.

## OPERATING SYSTEMS

are software you don't have much choice about. (See p. 99.)

### Stuff You Can Get:

#### THE MAIN THREE PROGRAMS

Everyone in computerdom thinks they

know what the basic programs are: word processing, spreadsheet, database. They're your Three Friends; your Scarecrow, Tin Woodsman and Cowardly Lion. (I think that ten years from now even these supposedly basic programs will have undergone some serious changes, and gotten rather destabilized. They're only considered basic right now because our horizons are still narrow.)

#### 1. "WORD PROCESSING"

"Word processing" refers, by convention, to certain simpleminded text manipulation and printout programs. (Programs that also help you organize are currently called Outline Processors, and programs that print text in more than one column are called Desktop Publishing. See text systems, p. DM 27.)

## SOFTWARE BABBLE-ON:

### PROBABLY THE FIRST PERSONAL SOFTWARE EVER OFFERED

One Benjamin Pitman offers a magnificent program in FORTRAN to generate textual garbage. It's so good it can be used to expand proposals by hundreds of pages. He calls it Simplified Integrated Modular Prose (SIMP) and it sells for \$10.

*Some novelty programs offered by Ben. Pitman, Esq. (see text)*

**Additional Programs**

**Calendar**  
This FORTRAN program can produce a calendar for any year from 1964 through 2100. The calendars are printed at the bottom of a Playboy bunny who is perched atop a bar stool. \$20.

**Birth Announcement**  
This PL/I program produces a picture of a baby in the fetal position with the details of a birth announcement in the interior. The details may occupy up to 160 characters. See reduced copy attached. \$15.

**Simplified Integrated Modular Prose (SIMP)**  
This FORTRAN program produces pages after pages of double tall. The output is a permanent data set of phrases followed by one or more user-written title cards. Each title card is placed at the top of a one page listing. The program uses a letter from the title and a random number generator to select phrases from the title and combines them into sentences and paragraphs to produce the 'report'. The program may easily be expanded by increasing the size of the arrays and the data in the phrase deck. See attached example. \$10.

**Santa and Reindeer**  
The deck when listed produces a Santa in his sleigh being pulled over several tree tops and houses by a reindeer. Above are the words: MERRY CHRISTMAS AND HAPPY NEW YEAR. \$5.

**Snoopy**  
The deck when listed produces a Snoopy. \$5.

**More recent address:**  
c/o Computech Systems Inc.,  
1819 Peachtree Rd.,  
Atlanta GA 30309.

The personal and desktop computer market have brought a wild and woolly new world—an explosion, an unleashing of energy and talent and zaniness as chaotic, egotistical and creative as rock music.

What's wanted one month is dead the next. And everyone keeps trying to guess the New Direction.

There are packages for everybody and everything; for gas station and graveyard management; even "Herdstar" for managing your fleet of cows. And an extraordinary variety of games (discussed on the other side of the book—see p. DM 19).

This loony vision is totally American (See "American Dreamer," p. DM 3)—with the American lunacy that gave us Fantasia and Daffy Duck and Marvel Comics and Dungeons and Dragons. (But, as with rock music, as soon as the English figure it out, we're in trouble. Douglas Adams' adventure game, Hitchhiker's Guide to the Galaxy, may be the first example.)

**STATUS REPORT NUMBER 5**

THE PROJECT CONTINUES TO REQUIRE CONSIDERABLE ANALYSIS AND TRADE-OFF STUDIES TO ARRIVE AT THE FINAL SYSTEM ARCHITECTURE. THEREFORE, IT IS INTUITIVELY OBVIOUS THAT A CONSTANT FLOW OF EFFECTIVE INFORMATION, MAXIMIZING THE PROBABILITY OF SUCCESS AND MINIMIZING THE COST REQUIRED FOR THE ANTICIPATED THIRD GENERATION EQUIPMENT.

AS A RESULTANT IMPLICATION, PARALLEL ORGANIZATIONAL CAPABILITY BASED MANAGEMENT FLEXIBILITY FOR THE STRUCTURAL DESIGN, BASED ON SYSTEM-ORIENTED CONCEPTS. ACCORDINGLY, INITIATION OF CRITICAL SYSTEM DEVELOPMENT QUALIFICATION LIMIT. IN THIS REGARD, THE PRELIMINARY DEVELOPMENT IS FURTHER COMPOUNDED, WHEN TAKING INTO ACCOUNT THE POSSIBLE IMPACT OF STANDARDIZATION.

ON THE OTHER HAND, THE FULLY INTEGRATED TEST PROGRAM ADDS EXPLICIT PERFORMANCE LIMITS TO THE SUBSYSTEM COMPATIBILITY TESTING.

THUS, IT IS ONLY LATELY BEING THAT ANY ASSOCIATED SUPPORTING ELEMENT EFFECTS THE SIGNIFICANT IMPROVEMENT TO SYNCHRONIZED DIGITAL PROGRAMMING.

BASED ON INTEGRAL SUBSYSTEM CONSIDERATIONS, THE INCORPORATION OF ADDITIONAL SYSTEM CONSTRAINTS AND OVERLAPPING PERFORMANCE CONSTRAINTS TO THE EVOLUTION OF SPECIFICATIONS OVER A GIVEN TIME PERIOD.

IN RESPECT TO SPECIFIC GOALS, A PRIMARY INTERRELATIONSHIP BETWEEN SYSTEM AND/OR SUBSYSTEM TECHNOLOGIES IS REQUIRED TO ENSURE CONTINUOUS INTERFACIAL WITH THE MINIMUM THROUGHPUT TIME CONCEPT. TO ACHIEVE THIS, ANY TRANSITIONAL INPUT OR OUTPUT STATE PRESENTS EXTREMELY INTERESTING CHALLENGES TO ANY DISCRETE CONFIGURATION WIDE.

SIMILARLY, THE CHARACTERIZATION OF SPECIFIC CRITERIA MUST UTILIZE BOTH FUNCTIONALLY INTERWOVEN WITH THE SIMPLIFIED HARDWARE. IN PARTICULAR, DIRECT ACCESS MODULAR I/O DEVICES WILL GENERATE THE MULTI-RESPONSE PARAMETERS FOR ANY DESIGNATED AUTOMATIC PERIPHERAL DEVICE.

HOWEVER, A LARGE PORTION OF THE INTERFACE COORDINATION COMMUNICATION RECOGNIZES THE IMPORTANCE OF OTHER SYSTEMS AND THE NECESSITY FOR TIME-PHASED INTERRUPT HANDLING.

**SIMP EXAMPLE**



## 2. SPREADSHEET PROGRAMS

One of the few truly new ideas in the computer field in the last ten years is the spreadsheet program. Everybody should have one; I would argue that it is more important to teach children spreadsheet than to teach them long division.

The computer spreadsheet was invented by Dan Bricklin and Bob Frankston; their original creation was called VisiCalc, and it came out in 1979. The impact on American business was immense; hundreds of thousands of Apple computers were sold just so the purchasers could use VisiCalc. Now there are dozens of spreadsheet programs available for every computer.

A computer spreadsheet is a grid of numbers that update automatically when-

ever a change is made. You create a map of what numbers are to be affected by what others, and the resulting numerical operations are done automatically. You do this by preloading specific cells in the grid with orders about what to do with numbers that are put there. The set of formulas you put into a sheet is called a **spreadsheet model**—a new kind of computer program. Your grid of numbers is always kept up to date by the formulas you've set up.

The spreadsheet can be very large; you can use it for records, possibilities, plans. In the commonest form, the spreadsheet adds numbers and takes percentages; this is used for all kinds of planning by corporate types. But there is no limit to the numerical complexities it can handle.

The computer spreadsheet is a new

way of programming. You do not write lists of commands; instead, you arrange relationships.

And your spreadsheet setup always **works as you're building it**—unlike conventional programming, which does not work until all the pieces are in place. The overall spreadsheet program is a frame in which the formulas you put in are operative immediately. **Spreadsheets can be programmed incrementally.** They hold together as you change them, unlike programs written in conventional languages. They work when they start out; they work after each change. (And though you may make errors, they do not interfere with the continual running of the system.) This means you can build programs you need for many different purposes piece by piece, and have the constituent opera-

tions work all along the way.

VisiCalc came out for the Apple II, and made the Apple take off. A program called Lotus 1-2-3 did roughly the same thing for the PC; its main new features were pie charts and bar graphs that could be automatically done from the numbers in the spreadsheet; and macros, automatic operations you could add yourself.

At \$400, Lotus 1-2-3 has been a fabulous success, and has become virtually standard in American business; it allows the user to search for specific items and to write programs that work with the spreadsheet. (The trademark comes from the fact that founder Mitch Kapor used to teach Transcendental Meditation.)

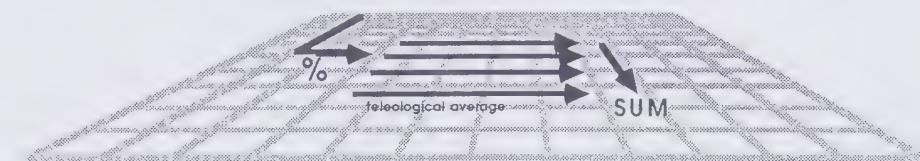
There have been numerous spreadsheet programs, many with their own twists; for instance, Microsoft's Multiplan

“The best uses for personal computers haven't been invented yet. To disagree with that is simply a failure of imagination.”

—Mitch Kapor

### SECRET FORMULAS:

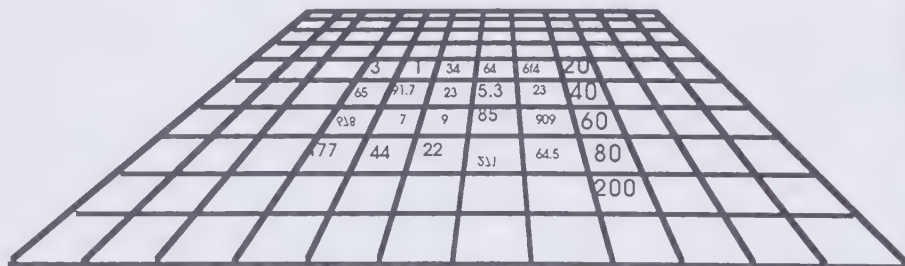
no way to see the structure as a whole



of the

### NUMBERS UNIDENTIFIED:

the visible, but mysterious, spreadsheet



Choosing software is like moving to another country—what you want to know beforehand can't be found out until you get there.

### PERSONAL PRODUCTIVITY SOFTWARE FOR THE POWER USER FNORD

The latest stupid catchphrase for software that has no particular purpose but might help, if you figure out how, is “personal productivity software.” This term is big and vague and can be applied to anything.

Another nice stupid term is Power User. Who is the Power User? Why, anybody who can figure out how to use that complicated stuff. You're beginning to hear the software people lament about the lack of Power Users to appreciate their grand complications.

• EXPECT TO  
• SPEND A  
• LOT OF TIME  
• AND MONEY  
• ON SOFT-  
• WARE.

The Good News: for many things you want to do, you can buy programs.

The Bad News: you may not like them or be able to make them fit with anything else.

allows linkages between different spreadsheet islands, and allows cells to be named, which simplifies putting in the formulas. Others are programmed for high speed, large size or the like.

But it is amazing to see how completely all the other brands have stuck to the VisiCalc design. The VisiCalc design was excellent and gemlike; but there are many possible directions that the spreadsheet program could change, and by and large it has not.

### 3. DATABASE PROGRAMS: Electronic File Cards

"Database" is what used to be called Information Retrieval (see p. DM 127). In personal software, this now refers to various programs that help you store, sort and find items of information. You use virtual file cards, stored electronically, and riffle through them with commands given on the screen.

Your database system holds your countable and categorizable data items.

The items can be anything: names and addresses, hotel reservations, your underground comic collection. And you ask the system (by some method) how many whatchamacallits with so-and-so traits it's holding information about.

You can get counts of items, pieces of text, addresses; request all the items with a given description, or with certain traits but not others.

It's like the card game Go Fish, where you also make requests and get items back:

"Give me all your jacks."

"I have two."

"Give me all your tens."

"Go fish."

So you might inquire of your personal address database, "Who are my friends in Idaho?"—helpful if you're driving through those parts.

And it would either give you the names and addresses, or print out something like

GO FISH.

(The exact language in which you state these inquiries is tricky. Like computer

languages in general, it will tend to be intricate, exact and frustrating, but the particular style will depend on what database you are using.)

A database is **work**. YOU CAN'T MAKE USEFUL INQUIRIES UNLESS YOU KEEP IT UP TO DATE.

There are great differences among the many, many database programs available—it seems that the most programs for the PC are databases—and no suggestions will be offered here. The products span a great range, and you should in principle take months of study to make the choice.

### OTHER KINDS OF PROGRAMS

Besides the Standard Three Programs, you can get canned software in spectacular variety—for business, engineering, art; involving simulation, 3D models (see CAD/CAM, p. DM 83), audio, pictures and moviemaking (see much of DM side). As well as artificial intelligence, whatever that may be this week.

Presently, like other programs, these do not tie together either.

### TRYING TO TIE THINGS TOGETHER

On the Mac, the unified interface and data structure are one way of tying things together—as is the Clipboard, a mechanism for moving data between programs.

But better ways have to be found.

### INTEGRATED SOFTWARE

"Integrated software" usually means a package which combines today's big three — WP/spreadsheet/database — in one unified chunk, maybe with other functions thrown in.

Instead of using separate programs, you stay inside the one program and do everything there, without going out to the operating system.

There are integrated software packages for the most popular machines—"Magic Desk" for the Commodore 64, "Jane" for Apple II and Apple II plus, Appleworks for Apple IIe and IIc. For the

PC: Symphony, Framework, Ability, Enable, Context. For the Mac: Jazz, Ensemble.

At prices of up to \$600, the integrated software package may seem like a lot of money till you realize what it does. Think of it as paying for the loose programs plus some extra to avoid using the separate applications separately, and to **avoid having to connect them through the operating system**. (The manufacturer of the software is also guaranteeing that the parts will tie together—no small promise, given the incompatibilities of the software world.)

That's the good news. The bad news is the extraordinary complication of some of the integrated packages. Worst on the PC is Symphony; easiest is Ability. (Framework is clean but very hierarchical, so there's a lot it won't do.)

Furthermore, integrated software doesn't **begin** to incorporate the many varieties of other programs you may want.

### THE GOOD NEWS

Xensoft, of Berkeley CA, makes a program, Xenocopy, that converts among 300 file formats.

### THE BAD NEWS

It's needed.

"Turn \$145 Into a Million! Discover R.E. Pro-1 for only \$145. You save by ordering at this special introductory price...and get all the easy-to-use tools required to become a Real Estate tycoon."

(Ad in March 85 Macworld for what appears to be a spreadsheet model.)

### ON-THE-SIDE PROGRAMS

Both the PC and the Macintosh have programs that can be used (among other things) to carry data to and from different programs.

On the PC these are called "memory-resident programs." On the Mac they are called Desk Accessories. For both machines they are hard to set up.

### OVERVIEW SYSTEMS

For both the PC and the Mac, there are Overprograms that let you run other programs underneath, and (to varying degrees) move data between them. On the Mac there is only one, Switcher. For the PC there are IBM's Topview, Quarterdeck's DESQ, Concurrent PC-DOS from Digital Research, and Windows from Microsoft. Unfortunately all of **these** tend to be expensive and hard to set up. In principle these could help you bring together a greater variety of programs than the integrated or on-the-side methods. In practice it's **all** a mess.

"The best of the stuff is a brilliant solution to the wrong problem."

—Roger Gregory

## Wonderful Worlds of Incompatibility: The Pits.

Almost every software package is a pit: you put your data in, you can't get it back out. Your commitment to the program is semi-irrevocable, because of its special features

Sometimes there are compatible data formats that can go between programs. But not always. This is a venerably oppressive scam in computerdom: make your stuff compatible with what came before, but leave no compatibility path away from it. So the customer is stuck

**The term "database" is also used for somebody else's collection of items stored in this fashion, which you may search and explore for a fee. There are now thousands of "database publishers," whose treasure chests can open up to you. (See "Issues," p. 165 and p. DM 127.)**



**"With our system you won't have to program. You'll just string commands together." This is like that other great lie: "Our new church is non-denominational."**

There is no longer any reason to know about hardware, unless you are a specialist. To get computers to do what you want, you program. Which requires an immersion in one or more COMPUTER LANGUAGES. And most of the new ideas, too, are in computer languages.

**Why can't there be just one computer language? Good question. Like, why can't there be just one religion?**

**Computer languages are a religious issue. A person usually adopts a computer language on the basis of a one-on-one personal conversion. People are fanatically loyal to their favorite language, and often cannot rationally discuss others.**

## Department of Notational Engineering

is what it used to read on the door of Drew McDermott at Yale.

This is only partially a joke. The creation of computer languages is essentially the creation of useful notations by which humans may successfully inform automatic machinery of their desires. Inventing notation, and operations corresponding to these notations, is a high and mighty new art.

• Most of the great computer languages represent the forceful vision, personal obsessions and unifying ideas of one individual (two for C), carried through with relentless consistency. Ken Iverson, Alan Kay, Charles Moore, John McCarthy all designed elegant and powerful languages.

A comma ends an argument in the TRAC language? Ah, that all arguments could be ended so easily.

—My grandfather.

# MAGIC LANGUAGES

A computer language is a system for casting spells. This is not a metaphor but an exactly true statement. Each language has a vocabulary of *commands*, that is, different orders you can give that are fundamental to the language, and a *syntax*, that is, rules about how to give the commands right, and how you may fit them together and entwine them.

Learning to work with one language doesn't mean you've learned another. You learn them one at a time, but after some experience it gets easier.

There are computer languages for testing rocket ships and controlling oil refineries and making pictures. There are computer languages for sociological statistics and designing automobiles. And there are computer languages which will do any of these things, and more, but with more difficulty because they have no purpose built in. (But each of these general-purpose languages tends to have its own outlook.) Most programmers have a favorite language or two, and this is not a rational matter. There are many different computer languages—in fact thousands—but what they all have in common is *acting on series of instructions*. Beyond that, every language is different. So for each language, the questions are

WHAT ARE THE INSTRUCTIONS?

and

HOW DO THEY FIT TOGETHER?

Most computer languages involve somehow typing in the commands of your spell to a computer set up for that language. (The computer is set up by putting in a bigger program, called the *processor* for that language.)

Fortran Machine



TRAC Language Machine



(Same computer with different language processors loaded in its core memory)

Then, after various steps, you get to try your program.

Once you know a language you can cast spells in it; but that doesn't mean it's easy. A spell cast in a computer language will make the computer do what you want—

IF it's possible to do it with that computer;

IF it's possible to do it in that language;

IF you used the vocabulary and rules of the language correctly;

and IF you laid out in the spell a plan that would effectively do what you had in mind.

BUT if you make a mistake in casting your spell, that is a BUG. (As you see from the IFs above, many types of bug are possible.) Program bugs can cause unfortunate results. (Supposedly a big NASA rocket failed in takeoff once because of a misplaced dollar sign in a program.) Getting the bugs out of a program is called debugging. It's very hard. (see p. 42.)

## COMPUTER LANGUAGES

are what make computers go 'round.

If your computer only did one thing, then to start it you'd only need one button to press.

If your computer only did two dozen things, without variations, then you could let each operation be started by pressing one of the keys of the terminal, and that would be that.

But that's not what it's about.

We have *lots* of different things that we want computers to do, and we want one command to work on different varieties of data, or on the results of a previous command, or even to chew on another command itself; and so a computer language is a **contrived method of giving commands to a computer that allows the commands to be entwined in a complex fashion.**

This means having rules the computer can carry out and the person can remember.

This means having basic operations that can be built into bigger operations

(routines, subroutines, subprograms, programs).

Thus a computer language is really a method by which a user can tie these programs together. Computer languages are built according to *contrived* sets of rules for tying programs together. Such rules are limited only by the imagination of their contrivers. Each computer language has its *own* contrived system of rules, and it may be completely different from the contrived rules tying together any *other* computer language.

Computer languages tend to look like nothing else you've ever seen. Thus computer programs, which of course have to be written in these computer languages, look pretty weird. Some programs look like old train schedules (in multiple columns). Some look a little like printed poetry. In any case, a COMPUTER PROGRAM NO MORE LOOKS LIKE ITS RESULT THAN THE WORD "COW" LOOKS LIKE A COW.

One of the central concepts of this book is that of a "program follower," a dynamic entity which somehow follows a program. Well, EVERY LANGUAGE HAS A PROGRAM FOLLOWER FOLLOWING ITS OWN PARTICULAR RULES. These rules are contrived for convenience, suitability to a purpose, and "aesthetics" of a sort—often some form of stark compression. (The program followers *wired* into computers are somewhat more akin to one another; see "Rock Bottom," p. 77). About all we can say languages have in common is: EVERY COMPUTER LANGUAGE ALLOWS LOOPS, TESTS AND BRANCHES, AND COMMUNICATION WITH EXTERNAL DEVICES, as mentioned on p. 37. Beyond that the differences are incredible.

A computer language is a system for tying together the fundamental operations of computers for larger tasks. Each computer language fits together according to its own principles, based in part on the personality and preoccupations of the person or people who designed it.

Modern computer languages generally can handle all the main kinds of programming: text handling, number crunching, storing files on disk memory and getting them back, and controlling whatever external devices you may have. Even making pictures in some way or other.

# A Simple ▼ Sample Program

## THE FORTUNE-COOKIE PRISONER:

### AN EXAMPLE OF PROGRAM LOOPS

*In the original Computer Lib, I had sections teaching three languages: BASIC, APL and TRAC Language. We've dropped that in this edition.*

*However, to give you a feel for the most fundamental program construct—the program loop—here's the same example in each of the three languages.*

*A program loop is a series of events that repeats over and over. And here are examples to show you how different this can look in three different languages.*

#### IN TRAC Language

```
PROGRAM)
#(PS, HELP; I AM TRAPPED IN A LOOP)#(CL, PROGRAM)
```

If we type in to our TRAC processor

```
#(CL, PROGRAM)
```

it should type

```
HELP; I AM TRAPPED IN A PROGRAM LOOP
HELP; I AM TRAPPED IN A PROGRAM LOOP
HELP; I AM TRAPPED IN A PROGRAM LOOP
```

indefinitely.

Why is this? Let's go through the steps.

We have a string named PROGRAM which consists of

```
#(PS, HELP; I AM TRAPPED IN A PROGRAM
LOOP)#(CL, PROGRAM)
```

The TRAC processor scans across it to the first right parenthesis.

```
#(PS, HELP; I AM TRAPPED IN A PROGRAM LOOP)#(CL, PROGRAM)
```

and now executes this.

It happens that PS is the PRINT STRING instruction. PRINT STRING prints out its second argument, and forgets the rest. But the only argument after PS is

```
HELP; I AM TRAPPED IN A PROGRAM LOOP
```

so it prints *that*. If it had said

```
HELP, I AM TRAPPED IN A PROGRAM LOOP
```

the PRINT STRING command would only have printed HELP since a comma ends an argument in TRAC language.

Now, the PRINT STRING command leaves no result, so it is vaporized; all we have left in the work area is

```
#(CL, PROGRAM)
```

which is now scanned. But that's another CALL, and

when it is executed by fetching the object called PROGRAM, its replacement in the work area is

```
#(PS, HELP; I AM TRAPPED IN A PROGRAM
LOOP)#(CL, PROGRAM)
```

and guess what. We done it again.

#### IN BASIC

```
43 PRINT "HELP, I AM CAUGHT IN A LOOP"
67 GOTO 43
68 END
```

The program will start at the first instruction, which happens in this case to be instruction number 43. That one prints a message. The next command, by line number, is 67. This tells the program follower to go back to 43, which it does.

```
43 PRINT "HELP, I AM CAUGHT IN A LOOP"
67 GOTO 43
68 END
```

```
43 PRINT "HELP, I AM CAUGHT IN A LOOP"
67 GOTO 43
68 END
```

The result is that your terminal will print

```
HELP, I AM CAUGHT IN A LOOP
HELP, I AM CAUGHT IN A LOOP
HELP, I AM CAUGHT IN A LOOP
```

interminably, or until you do something drastic. It never gets to the END statement. (Two strategies for doing something drastic are usually to hold down the CONTROL button and type C, or hold down both CONTROL and SHIFT buttons, if you have them, and type P. One of these usually works.)

#### IN APL

And here is our old friend, the fortune-cookie prisoner.

```
INF
[1] ← 'HELP, I AM CAUGHT IN A LOOP'
[2] → 1
```

On line 1 the program prints out whatever's in quotes. And line 2 causes it to go back and do line 1 again. Forever.

*All of the loops go on forever, which soon becomes tiring. And so one of the big problems of programming is to create tests and conditions for getting out of a loop once it has started operation. In other words,*

### LEAK BEFORE YOU LOOP



*Books on most of these languages are available at places like Dalton's and at university bookstores. I suggest starting with magazines, though, like Byte, Dr. Dobbs, or Computer Language, before getting seriously involved with a particular language. Unless it's your assigned job, of course.*

**Last I heard, Ken Iverson had moved to I. P. Sharp, the worldwide APL time-sharing system operating out of Canada.**

*FORTH speeds are not comparable to any other processor, but some think this chip is in the Mini-Cray class. (See p. 90) Note, however, that Moore—an astronomer and physicist—is opposed to floating-point numbers on doctrinal grounds, so that makes the comparison more difficult.*

*Most of these languages are available, in some form or other, for the personal computer—that is, PC or Macintosh.*

This is serious business. Languages like BASIC must be considered by people who want simple systems to do understandable things in direct ways that are meaningful to them, and that don't disrupt their companies or their lives.

You can't add  
to a jewel.  
Adding more  
mud to a  
bowl of mud  
gives you a  
bigger bowl of  
mud.

Everyone should have some brush with computer programming, just to see what it is and isn't. *What it is*: casting mystical spells in arcane terminology, whose exact details have exact ramifications. *What it isn't*: talking or typing to the computer in some way that requires intelligence by the machine. *What it is*: an intricate technical art. *What it isn't*: science.

Every programmer who's designed a language, and created a processor for it, had certain typical uses in mind. If you want to create your own language, you figure out what sorts of operations you would like to have be basic in it, and how you would like it all to fit together so as to allow the variations you have in mind. Then you program your processor (which is usually very hard).

*The languages listed here are some of the most important, roughly as well known in the computer community as States of the Union—and with the same regional loyalties.*

*Some stones, because of their elegance and cleanliness, are gems. These are gems.*

*The C language was developed at Bell Labs by Brian Kernighan and Dennis Ritchie; it is the sophisticates' favorite for fast, practical programs (unlike LISP, which is the sophisticates' favorite for sprawling, complex, ever-changing and experimental programming.)*

*C has a declarative-looking structure like the FORTRAN family, but allows much tighter control to programmers who want to use it. It allows tight programming at the bit level, like machine language, yet has all the clean control features wanted by structured-programming freaks. In clean and simple ways it allows the programmer to create and step among arbitrary structures and arrays of any kind. The UNIX operating system is programmed in C, as are an ever-increasing*

(A superb elegantized version of C was created at Datapoint by Gene Hughes under the name of DASL. It was used for internal development but never got out the door, despite its cleaner syntax and greater power.)

Some people call it a “scientific” language. Some people call it a “mathematical” language. Some people are most struck by its use for interactive systems, so to them it’s an interactive language. But most of us just think of it as THE LANGUAGE WITH ALL THE FUNNY SYMBOLS, and here they are:

Enthusiasts see it as a language of inconceivable power with extraordinary uses. Cynics remark that it has all kinds of extraordinary powers for inconceivable uses—that is, a weird elegance, much of which has no use at all, and some of which gets in the way.

This is probably wrong. APL is a terrific and beautiful triumph of the mind, and a very useful programming language. It is not for everybody, but neither is chess. It is for bright children, mathematicians, and companies who want to build interactive systems but feel they should stick with IBM.

APL is one of IBM's better products, probably because it is principally the creation of one man, Kenneth Iverson. (Actually Iverson designed the language at Harvard and programmed it on his own initiative after moving to IBM; it was added to the product line by popular demand.) APL is a language of arrays, with a fascinating notation.

*It is superb for mathematical explorations and demonstrations, and every kid should have it.*

*FORTH is like APL in the brilliant interweaving of its constructs. (Cynics say they are alike in that neither language can be read after it's written.) Originally created by Charles Moore to control radio telescopes, FORTH has proven its power, efficiency and compactness in every field from hospital management to motion-control special effects. (One of FORTH's special advantages is that it runs on almost every computer now available, and is the*











## LISP FAMILY

(PARENTHESES (LIKE YOU WOULD NOT BELIEVE))

In the MOST GENERAL COMPUTER LANGUAGE

LISP

(or preferably SCHEME)

LISP is probably the favorite language of the artificial-intelligence freaks (see Dream Machines p. 120). A fondness for LISP, incidentally, is not considered to reflect on your masculinity.

LISP is a "cult" language, and its adherents are sometimes called LISPians. They see computer activities in a somewhat different light, as composed of ever-changing chains of things called "cars" and "cudders," which will not be explained here.

LISP was developed by John McCarthy at MIT, based largely on the Lambda notation of Alonzo Church. It allows the changing of operations and data in deeply intermingled forms. While it runs on elegant principles, most people object to its innumerable parentheses.

Joseph Weizenbaum, also of MIT, has created a language called SLIP, somewhat resembling LISP, which runs in FORTRAN. That means you can run LISP-like programs without having access to a LISP processor, which is helpful.

Used to be that in order to learn LISP you had to be a member, or hanger-on, of an artificial intelligence department, which meant you had to be at a large university or research organization. (Indeed, traditionally the way you got into artificial intelligence was by hanging around and learning LISP. If you got good enough you became accepted and got a job.)

It was restricted this way for several reasons: it only ran on big machines, it was impossible to learn except through practice and the advice of local hackers.

Gradually, the dialects of LISP became very different at different institutions, and so "learning LISP" meant different things depending on where you were.

The original John McCarthy LISP split into two families: MACLISP at MIT, or MIT LISP, and the West Coast dialect. MIT LISP became SCHEME and LMI LISP (sold by LISP Machines, Inc. with their computers), and the West Coast dialect became InterLISP, a huge and busy system supported by Xerox.

Common LISP endeavors to bring East and West Coast dialects together. Note that "Standard" LISP is not standard.

While there are still several other supported dialects around, now there is a standard with (of all things) Defense Department blessing. This is Common LISP, and it's already available for the IBM PC.

Note that the SCHEME dialect (the cleanest variety of LISP) is the basis of Abelson and Sussman's excellent book (see nearby).

Everyone agrees that SCHEME (which looks just like LISP) is a much better language, but Common LISP has a probability of wider use—and can be learned through the superb San Marco LISP Explorer (see nearby and next page).

## THE BEST BEGINNER'S LANGUAGE: LOGO

Logo is generally looked at in two sharply different ways. As publicized to primary and secondary educators, Logo is a much better language for kids (or anybody) to learn than BASIC. (This is true.) To those who know its secret identity, however, Logo is really LISP without the parentheses.

There is no question that Logo is a better language for beginners than BASIC: it has no line numbers, no Gotos, and named subroutines that call each other in a well-behaved fashion. Also it has nice drawing techniques on the screen, called "turtle graphics."

But Logo has a more subversive purpose. It is a move by the LISPians to give the full, immense power of LISP to kids. This is terrific.

Seymour Papert, one of the developers of Logo and its chief publicist, argues passionately for the language in his book Mindstorms (BASIC Books).

Logo encourages you to think recursively, a style considered very important in the artificial-intelligence community. However, some people get the impression from Papert that thinking recursively is the right way to think—rather than one of the many different mathematical styles that kids ought to have access to. Of course kids should learn Logo. It can be argued that they ought to learn APL as a counterbalance.

Note that a case can now be made for LISP itself as a first language, now that it's becoming widely available.

## WHERE ARE THEY NOW?

TRAC\* Language (see footnote next page.)

One computer language described in the first Computer Lib—Calvin Mooers' TRAC\* Language—is no longer around. It's also a member of the LISP family.

TRAC Language had an interesting structure: it consisted of strings of characters which ask to be replaced by other strings of characters. This happened repeatedly, in a palpitating and unending cycle, getting various jobs done on the way. Actually, it was in some senses isomorphic to LISP, with all the same mind-boggling parentheses, but unfortunately much slower because of all the string-scanning. Nevertheless, it could be run in very small machines and had certain other good qualities.

No version of TRAC Language is available to the personal-computing market. However, a rather similar language is

available under the name SAM76. (Supposedly SAM stands for "Strachey and McIlroy," not "same as Mooers'.")

However, Calvin Mooers has burst into the late eighties with a similar but improved language and a bunch of with-it new trademarks. The new language is VXM, with tree-structured data, in which you can program SoftRobot agents and "super shells that span and unify multiple systems." The company is Command Technologies, Incorporated, in Boston. VXM does interactive stuff and can also negotiate with the operating system, offering the hope of maybe CLEANING UP THE FILE PROBLEM. \$200 for the PC, \$1000 under XENIX.

## BUREAUCRATIC LANGUAGES

### YECCCH, IT'S COBOL

Research and hobby types hate COBOL or ignore it, but it's the main business programming language. Your

**Laurent Siklossy, Let's Talk LISP. Prentice-Hall. Exceptionally readable and nice; distinguishes between aspects which vary among different dialects.**

**Patrick Henry Winston and Berthold Klaus Paul Horn, LISP. Addison-Wesley. Introduction to Common LISP.**

**Abelson and Sussman, Structure and Interpretation of Computer Programs, MIT/McGraw-Hill. A serious book for someone who wants to become a fine programmer. Uses the Scheme dialect.**

**Golden Common LISP with the San Marco LISP Explorer. A magnificent product at a reprehensible price (\$500). Allows you to learn the language through exploration, modify programs while they are halted, and lots more. Excellent for the serious beginner.**

An MIT Logo researcher named Radia Perlman built a setup for programming without typing. Here's the writeup I ran before it was done:

A special "preliterate" terminal to allow non-readers (possibly including chimps and gorillas) to program in Logo. Plastic credit cards will have symbols for the various picture and music-box functions. To write a program, or create a movie on the scope, the user will insert function cards in slots. Color coding will be used for program transfer: A red card means "jump to the red subroutine." Since this is MIT, the full recursive power of the system will of course be available. (My hope is that chimpanzees and other little slotniks can be taught recursive program definition. Then will the public wake up to computers being easy?)



income tax, your checking account, your automobile license—all are presumably handled by programs in the COBOL language.

COBOL, or COMmon Business Oriented Language, was more or less demanded by the Department of Defense, and brought into being by a committee called CODASYL, which is apparently still going. COBOL uses mostly decimal numbers, is designed basically for batch processing (described elsewhere), and uses verbose and plonking command formats.

Just because it's standard for business programming doesn't mean it's the best or most efficient language for business programming; I've talked to people who advocate business programming in FORTRAN, BASIC, TRAC and even APL. But then you get into those endless arguments...and it turns out that a large proportion of business programmers only know COBOL, which pragmatically settles the argument.

There are people who say they've discovered hidden beauties in COBOL; for instance, that it's a splendid language for complex pointer manipulation (see Data Structure, p. 70)

That's what makes horse racing.

**\* TRAC**  
*Language was a registered trade mark of Rockford Research, Cambridge, MA. VXM and "SoftRobots" are trademarks of Command Technologies.*

## THE FIRST LADY OF COMPUTERDOM: ADA

(But would she have approved?)

The language ADA was named after the first computer programmer—Ada Augusta Lady Lovelace, illegitimate daughter of Lord Byron. Brilliant and beautiful, she lived with Charles Babbage during his long years of travail in attempting to build the Analytical Engine, which would have been the first computer; and she wrote the first program for it.

Unfortunately, the language named in her honor is probably not worthy of her. It is huge, complex and sprawling, an omnibus language like PL/I. While the Defense Department wanted a general-purpose language for all purposes, this one has a number of glitches in the control structures that some people regard as very dangerous.

C.A.R. Hoare has gone so far as to suggest that ADA's design is so bad that its uncontrollable programs could start a nuclear war.

## UNUSUAL AND MYSTERIOUS

### THE PATTERNALIA OF

### SNOBOL

SNOBOL is the favorite computing language of a lot of my friends. It is a list-processing language, meaning it's good for amorphous data. (It derives from several previous list-processing languages, especially IPL-V and COMIT.)

SNOBOL is a big language, and only runs on big computers. The main concept of it is the "pattern match," whereby a string of symbols is examined to see if it has certain characteristics, including any particular contents, relations between contents, or other variations the programmer can specify; and the string substitution, where some specified string of symbols is replaced by another that the programmer contrives.

## LOGIC PROGRAMMING: PROLOG

All previous programming has required you to specify the sequence of operations. A new, interesting and powerful language, Prolog, permits you to specify the operation of a computer in terms of the objectives to be sought, in a specific logical notation.

Enthusiasts of the language claim that this makes it unnecessary to consider the sequence in which the computer carries

out its operations, making it a complete breakthrough. Actually, the Prolog programmer must be just as exact about the operational sequence implicit in his program as before, but in new ways: unfortunately some ways of stating a program in Prolog work much better than other ways which are logically equivalent, and there's no way yet for the program to make the right decisions on its own.

A special advantage of Prolog, though, is the fact that the **reader** of the program does not have to know the correct formulations to make the program work right, and can **read** the program strictly for its logical content. (Assuming it has already been proven to work.)

## CONCURRENT PROLOG

is not the same as Prolog; its name is misleading. Mark Miller says (1986), "Flat Concurrent Prolog will take over the world. It's everything I used to think Objects and Actors were." (see below.)

## OBJECT AND ACTOR PROGRAMMING

Object and Actor programming is a completely new way of thinking and programming.

The new kind of language is what they call "object-oriented," which means that your pieces of program and data behave more like physical objects than like pieces of information. An object—that is, a data or program thingy—responds to messages which are sent to it by other objects, and each object has a vocabulary of messages it is prepared to send and receive.

Programs and data structures are both packaged as "objects," named chunks that enumerate **all the things they can do**—the "methods," or programs they can use, and the outside **messages** that will trigger them.

## SIMULA

Moreover, all the above languages have the property that objects can spin off many copies of themselves, which act independently (called **instances**).

This sorcerer's-apprentice structure comes mainly from a Norwegian language called Simula.

These features effectively change the character of programming completely. For instance, to simulate a number of objects interacting, the program can spin off a copy of itself for every object, and each copy (mimicking the real-world object),

can then respond to its continually-changing environment as required. In other words, this type of language means that programs behave much more like the things being simulated than they ever did before.

Simula was created by Kristen Nygaard of Norway, a gracious and thoughtful old-world socialist.

## WHERE ARE THEY NOW?

Nygaard is creating a new language, called Beta, the son of Simula; and he tells me it will have extensive hypertext features. (See "Hypertext," p. DM 29.)

The best-known object language is legendary and widely misunderstood:

## THE LEGEND OF SMALLTALK

Alan Kay, acidulous celebrity-designer, claims that he did not actually **invent** a new kind of programming, but just noticed it around him in Simula and Sketchpad (see p. DM 96).

Anyhow, Alan (as everyone calls him) went on to develop new languages based on the object model. At Xerox PARC (see p. 128) he created a language called Smalltalk, a deceptively innocent name, for his Dynabook project (see p. DM 51). This language is, in the words of one friend of mine who has used it, "a miracle of power." The reasons why have to do with **the ease of modifying programs**, of rebuilding, of starting it over, of hacking till you get it right.

Though versions of Smalltalk were working in the early 1970s, there has been no way for anyone outside Xerox PARC, save for a few of the anointed, to get hold of it. Now, finally, it is possible to buy special Smalltalk machines from Xerox (\$60,000) and Tektronix (\$15,000). There are versions out for the PC and the Macintosh.

Why something so badly needed has been so tragically delayed, and why what began for the masses became a quintessentially elitist undertaking, will be discussed for the next hundred years.

For some reason the developers chose the hot-air balloon as the symbol of Smalltalk. Smalltalk is much more than hot air. What its impact will be on an ever-seething computer marketplace is absolutely impossible to predict; but it almost surely will not have the immense impact it would have had ten years ago.

## NEON

Neon (from Kriya Systems, Sterling, Virginia) is a clean, elegant object language for the Macintosh.

It is like Smalltalk, but built out of FORTH, and highly regarded by those who have used it.

## HIGH SEAS

Objective C is a language that allows you to program in C within an object-oriented paradigm. So is C++.

## OBJECT LOGO

allows you to program in Logo in terms of objects and messages and inheritance. This may be the hottest beginner's language in the computer field—but nobody is sure yet.

## MESA

is a large and clumsy object language used at Xerox. ("It's what you get when a lot of clever people hack on something obsolete." —anonymous PARCkie.)

## ACTORS

is a language you can't get, created by Carl Hewitt at MIT.

In Smalltalk, the different objects *seem* to be independent, but actually maintain a conventional control structure: one object is always in control, and the program is always at one particular step or other.

With Actors, however, Hewitt built a language *that really was what Smalltalk seemed to be*: myriads of independent entities floating around, doing their thing, reacting to each other.

## INTERESTING STUFF

### WHO SHAVES OCCAM?

Occam was created in England, largely from the language designs of C.A.R. Hoare.

The intent of Occam, named after the medieval philosopher who advocated simple hypotheses, is to permit you to program many things to happen at the same time on a bunch of separate processors. This is specially advantageous in programs like Pac-Man, where many things happen simultaneously; but it's also the same problem as the so-called Star Wars defense, that great Pac-Man in the sky (see p. 171).

(The processors are the Transputer chip (see p. 99) from Inmos, which you'll

supposedly be able to box up by the hundreds into parallel supercomputers. So you need a language to express the complex synchronizations and interrelationships required for such programs.)

(Note that Transputer boards, but with a single Transputer chip, are available at this writing for the PC and Apple II.)

Thus Occam has operations for saying things may happen independently (PAR) or happen in sequence (SEQ). In addition, a command called ALT—a sort of parallel IF—tests the flow—or torrent—of events.

While some of my friends don't like Occam's lack of data typing, pointers and recursion, this unique language may open the door to a whirlwind of low-cost parallel computers—with possibly the ability to do image synthesis in real time (see image synthesis, Dream Machines side, pp. 100-119).

## VISUAL PROGRAMMING LANGUAGES

Any method for giving exact instructions to a computer is a computer language.

Strings of characters are not the only way, and there have been numerous attempts to do it visually instead.

Bert Sutherland built a compiler in the late sixties that would compile from flowcharts. (A more recent form of structured flowchart, called Nassi-Schneiderman diagrams, would be wonderfully suitable for direct compilation, but I don't think it's been done.)

In recent years experiments have included work by the Logo group at MIT in using little symbols for the steps (following the lead of Radia Perlman's slot machine; see p. 61) and THINGLAB at Xerox PARC. Aaron Marcus, the designer, has also talked of visual computing languages, and Scott Kim, the Magic Inverse Calligrapher, did a visual language for his Ph.D. thesis.

And everyone is looking forward to Jaron Lanier's programming language Mandala (featured on the cover of the September 1984 Scientific American). A warm-hearted young musician-turned-

programmer, Jaron's neat new secret stuff has everybody grasping. (It can now be revealed that he's marketing an interactive glove for personal computers.)

## REIGNING CATS & DOGS

There are some miscellaneous languages and language types you ought to know about.

### LOTUS 1-2-3

is a very widespread interactive system—a spreadsheet with crude graphics—that has a language of its own ("Lotus Macros").

Writing Lotus Macros is a lot of people's first experience of programming. Fine if you can see beyond it.

### SABRE

is the language your travel agent probably uses; but you can't get it. Nor do you really want to.

### JCL

Some Call It Despicable,  
Some Call It Home

"After you study it for six months, it makes perfect sense."

—An IBM enthusiast.

JCL is a language with which you submit programs to an IBM 360 or 370 computer. "Submit" is right. Its complications, which many call unnecessary, symbolize the career of submission to IBM upon which the 360 programmer embarks.

The 360s and 370s have been replaced, both by bigger machines (which is the biggest varies from year to year) and smaller office models (from IBM and Japan). Various other software arrangements have evolved to cushion users from its awfulness, but JCL rolls right along underneath.

## SPECIFICATION LANGUAGES

are ways of specifying the design of other computer languages. That way you can haggle about the designs beforehand.

Backus-Naur Form (BNF) is the best known.

## APPLICATION GENERATORS

are a new animal—hybrid semi-languages at various levels—that permit serious programming with some attention to interactive design (see Dream Machines in general). Many of these allow the programmer to "paint screens" and interface with databases. They can be very simple, or full-fledged programming systems like Clarion for the PC.

## IMPLEMENTATION LANGUAGES

sometimes called

### P-CODES

An implementation language is a common language that is identically programmed for different machines—a sort of transportable machine language. An important example is the UCSD p-system, made to underlie implementations of Pascal, from the University of California at San Diego and whence the name p-code.

Such codes can be very condensed, but may lose on speed. (Thus Aztec C has its own p-code, which you can bypass for the parts that have to run very fast.)

## THEN, THERE'S ALWAYS MACHINE LANGUAGE

If you feel like making programs run fast, and not take up very much core memory, you go to machine language, the computer's very own wired-up deep-down system of commands. It takes longer, usually, but many people consider it very satisfying.

Then, of course, if you have a particular style and approach and set of interests, you will probably start building up a collection of individual programs for your own purposes.

Then you'll work out simplified ways of calling these into operation and tying their results and data together.

Which means you'll have a language of your own.

### Q. What's lower-level than Machine Language?

A. INTEGRATED-CIRCUIT COMPILERS!

Yes, friends, you can compile chips now, but it's expensive. See p. 99.

**"An 'object' is like a package that describes a specific kind of data and the set of all procedures that may work on that data. Thus, an object is a higher-level grouping of information; a type of package specifically designed for modularity and flexibility."**

**—Thomas Lubinski and Ingeborg Hutzel, "An Object-Oriented Graphical Kernel System," Computer Graphics World, July 1984.**

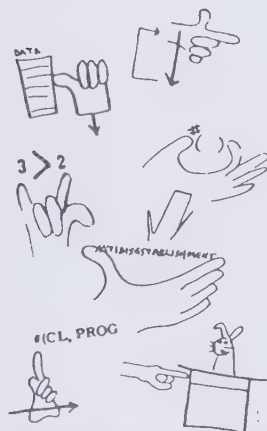


# PROGRAM FOLLOWERS BENEATH THE PROGRAM FOLLOWERS

One of the central concepts of this book is that of a "program follower," a dynamic entity which somehow follows a program. Well, EVERY LANGUAGE HAS A PROGRAM FOLLOWER FOLLOWING ITS OWN PARTICULAR RULES. These rules are contrived for convenience, suitability to a purpose, and "aesthetics" of a sort—often some form of stark compression.

About all we can say languages have in common is: EVERY COMPUTER LANGUAGE ALLOWS LOOPS, TESTS AND BRANCHES, AND COMMUNICATION WITH EXTERNAL DEVICES. Beyond that the differences are incredible.

The rules of different program followers can be very, very different.



## HOW DO COMPUTER LANGUAGES WORK?

Basically there are two different methods.

A *compiling language*, such as FORTRAN or COBOL, has a compiler program, which sits in the computer, and receives the input program, or "source program." It analyzes the source program and substitutes for it an object program, in machine language, which is a translation of the source program, and can actually be run on the computer. Many instructions in machine language are often needed to compile a single instruction of the source program. (A source program of 100 lines can easily come out a thousand lines long in its output version.) In fact, compilers tend to be very slow programs; but that depends on the amount of "optimizing" they do, that is, how efficient they try to make the object program.

An *interpretive language* works differently. There sits in core a processor for the language called an interpreter; this goes through the program one step at a time, actually carrying out each operation in the list and going on to the next.

Interpreters are perhaps the easier method of the two to grasp, since they seem to correspond a little better to the way many people think of computers. That doesn't mean they're better. For programs that have to be run over and over,

compiling is usually more economical in the long run; but for programs that have to be repeatedly changed, interpreters are often simpler to work with.

*APL and LISP are usually interpretive. FORTRAN and Pascal compile. BASIC comes in both flavors.*

## A BLACK ART

Making language processors, especially compilers, is widely regarded as a black art.

## BIBLIOGRAPHY.

David Gries, *Compiler Construction for Digital Computers*. Not for beginners, but a beautiful book. Good on abstract theory of languages, too.

## AN INTERPRETER

carries out each instruction as it's encountered.



## A COMPILER

chews the instructions of the language into another form to be processed later.

An Interpreter *carries out*, A Compiler *sets up*.



these commands are stored in the computer's memory, and the computer's program follower gets to them, they cause it to respond directly by electronic reflex. This is called *machine language*, the very language of the machine itself.

Every programmable device has a "machine language," or rock bottom code system that activates the thing directly; its program follower responds electrically to these codes, and enacts them one instruction at a time.

True computers are programmable devices that can modify their own instructions, change their sequence of operations and do other versatile stuff.

In most available computers the machine languages are binary, meaning composed of only two alternative symbols. (See p. 65.) Binary because it's a sensible way of organizing the machine's structure; it permits programs to be reduced to a single common form of information, and permits programs to be stored in binary memory. Each individual instruction or command ordinarily occupies one memory slot, though some computers have commands of varying length.

Different computers have different machine languages, but the instructions of all computers are basically similar. Big computers have more commands, with more variations, and carry them out faster; but those variations are just extra ways of saving steps, not qualitatively different features.

These deep-down operations ARE ALL THE THINGS THE COMPUTER EVER DOES. However, in their combinations these instructions can be woven into chains and diadems of complex actions.

ALL COMPUTER PROGRAMS ARE EVENTUALLY WRITTEN OR ENACTED IN THE MACHINE'S PARTICULAR BINARY LANGUAGE.

Now, it is entirely possible to write your programs at this level, considering and arranging rock-bottom commands. This is called machine-language programming (and assembly programming; see examples a little later on). Indeed, working at this level is very highly respected in some quarters. Others avoid it.

*Many people seem to think they would understand computers if only they could imagine how they worked deep down inside.*

*This is a laudable wish, but in general the lowest-level mechanics really do not help you understand computers, any more than understanding the details of engines and carburetors helps you understand cars and traffic.*

*However, this section is here to satisfy any urge you may have to understand the lowest level. What really counts, though, is higher languages and getting experience in them. So skip this section if you feel like it and come back some other time—if you feel like it.*

## ROCK BOTTOM:

## THE WORLD BENEATH THE HIGHER LANGUAGES

Every computer is wired to accept a specific system of commands. When

This is a very serious matter of taste and what you're working on.

Higher-level languages, seen on earlier pages, have more convenient forms for people, but must be translated, either ahead of time or on a running basis, to the bottom-most codes that make things happen in the machine. All of them are built out of machine language. Writing the language processors, programs that enact or translate these higher-level languages, is considered a black art. (See p. 64.)

## BINARY PATTERNS

are what the computer operates on deep down. "Binary" just means that only two symbols are used (just as "decimal" means that ten symbols are used). Patterns of binary symbols happen to be electrically convenient, so that's how computers are built, but that would change if some more convenient set of symbols came along.

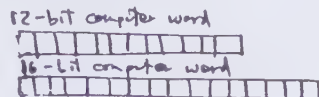
Binary patterns are very systematic and easy to deal with. Consider the number of binary symbols you can have in just four spaces. LET'S USE THE LETTERS X AND O, AND PUT THEM IN ALPHABETICAL ORDER. SO YOU'LL SEE THAT WE'RE TALKING ABOUT PATTERNS, RATHER THAN NUMBERS.

o	o	o	o
o	o	o	X
o	o	X	o
o	o	X	X
o	X	o	o
o	X	o	X
o	X	X	o
o	X	X	X
X	o	o	o
X	o	o	X
X	o	X	o
X	o	X	X
X	X	o	o
X	X	o	X
X	X	X	o
X	X	X	X

You can see that the pattern repeats in certain interesting ways. Each column repeats itself as you read down; adding a new position to the left doubles the number of possible patterns you can have in the row.

These are the infamous "bits" you have heard of. As you can see, there is nothing hard or complicated about them. The number of bits in a thing are the number of spaces which can be either X or O.

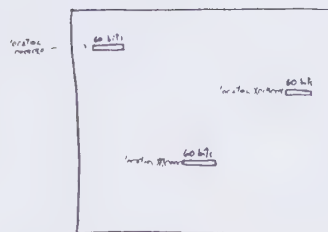
Now, the most basic fact about any computer is its *word length*, that is, the number of spaces in a standard memory slot of that computer.



A "12-bit computer" (like the PDP-8) has memory words that are all twelve bits long. A "16-bit computer" (like the PDP-11) has memory words that are all 16 bits long.

Actually computers with small word lengths like these are called minicomputers. Big computers have much bigger word lengths. The IBM 360 has a 32-bit word length. The Control Data 6600 has a 60-bit word.

Now, it is an interesting fact that not only are computer memories divided up into slots, or locations, of equal length,



but each of those locations has an address, that is, a number by which the contents of the location can be found. And these numbers are binary.

Many forms of information are kept in binary patterns which are not numbers. For instance, letters of the alphabet are usually sorted as 8-bit patterns.

X X O X O O O X

THE LETTER "Q"  
(IN ASCII CODE)

The big point is,  
AT THE BOTTOM PROGRAMS ARE  
BINARY AND DATA IS BINARY,  
since it's all stored in binary memory.

But since that suits few people's individual purposes, we build up HIGHER LANGUAGES AND DATA STRUCTURES. So that different users deal with different mechanics corresponding better and more conveniently to the structures that interest them.

However, we will have to stop using these X's and O's. It's not really *done*, so we will switch to the more usual way of writing binary patterns with 1s and zeroes. (Apologies to readers who hate numbers; but remember that these patterns, while we may write them out as 1s and zeroes, may *represent* wholly non-numerical kinds of information.) That means the letter Q is

1 1 0 1 0 0 0 1

but it's still the letter Q.

Of course, bits may also represent numerical information. And so we pass on to...

## INSTRUCTION-SET

The system of command patterns designed and wired into a particular computer, each with its exact results. (The instructions in the set are the vocabulary of a machine language.)

The big point is,

AT THE BOTTOM PROGRAMS ARE BINARY AND DATA IS BINARY,

since it's all stored in binary memory.

But since that suits few people's individual purposes, we build up HIGHER LANGUAGES AND DATA STRUCTURES. So that different users deal with different mechanics corresponding better and more conveniently to the structures that interest them.

When people tell you that "all computers really operate on 1s and zeroes," the right answer is "SO WHAT?" Bits are no more the reality when you're talking about data than atoms are the reality when you're talking about furniture.

SUPPORT COMPUTER SCIENCE  
Every Little Bit Counts

—postage-meter cancellation at Yale Dept. of Computer Science

## A Bit Is Not A PIECE

People who want to feel With It occasionally use the term "bit" for any old chunk of information, like a name or address. This is Wrong. A Bit is the smallest piece of binary information, an item that can be one of two things, like heads or tails, X or O, one or zero; and all other information can be packed into a countable number of bits. (How many may depend on the data structure chosen.)

As a handy rule of thumb: every letter of the alphabet or punctuation mark is eight bits; for heavy storage of everyday decimal numbers, every numerical digit can be further packed down (to four bits in BCD code).





## BINARY NUMBERS.

These are the same old binary patterns, but when we decide to treat them as numbers, they are binary numbers.

Let's count. Note that these are the same combinations of bits as before, merely put in the more usual notation.

decimal number	binary number
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111
08	1000
09	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

As you observe, the higher numbers need more and more bits to hold them.

This brings up some interesting facts.

CERTAIN NUMBERS ARE SPECIAL because they are the number of things that can be specified by a certain number of bits.

Special number

2	one bit
4	two bits
8	three bits
16	four bits
32	five bits
64	six bits
128	seven bits
256	eight bits
512	nine bits
1024	ten bits



("ONE K" is 1024; memories and everything else come in Ks, or multiples of 1024.)

Actually the term "k," standing for "kilo-," should mean one thousand, and

the term BK, or Binary K, is used by fussy people to stand for the very important nearby number 1024. But computer people generally use expressions ending in K for the following special numbers:

### THAT'S HOW MANY COMBINATIONS FIT IN

#### NUMBER

2048 ("2K")	eleven bits
4096 ("4K")	twelve bits
8192 ("8K")	thirteen bits
16,384 ("16K")	fourteen bits
32,768 ("32K")	fifteen bits

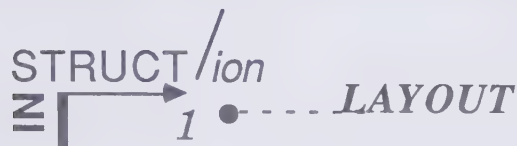
Above this number they increase very fast, and we generally have to look them up, but the idea is this: *the number of bits used to select something limits the number of things you can select among.* For instance, if you have a computer memory with 32K different locations, you need fifteen bits exactly to specify a location in memory.

Here are some ramifications:

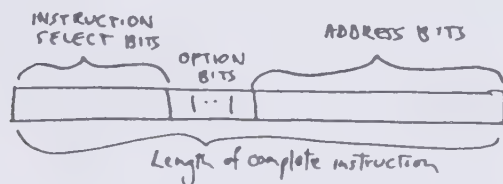
▪ The word length of a computer determines how large a number it can hold. A computer with a twelve-bit word can only hold a number up to 4095 in one memory location (since we use 000 000 000 000, the first combination, to stand for zero); if we want to use longer numbers we have to set aside two or more word locations per number. (A 16-bit computer can hold a number up to 65,535 in one memory location.)

▪ In designing data structures, if you use binary codes (rather than, say, alphabetical characters), you have to allow enough bits for all the alternatives that might turn up.

▪ In the design of the wired-in instructions for a computer, therefore, the number of bits set aside to specify an address in core determines whether that instruction can select from the whole memory, or just a part of it.



An occult aspect of computer design is the matter of how to pack into the so-many bits of an instruction word all the options the programmer should have.



The number of bits in the address determines the number of places in the memory that the programmer can choose among. 15 bits in the address means a choice of 32,768 memory locations. 7 bits means a choice of only 128. (See "Binary Patterns," p. 65.)

Deciding what the instruction layouts are to be hinges on the architectural design of the computer (see p. 76) and the instruction-set. It all gets worked out together.

It's ultimately a matter of design elegance, but the consequences are very concrete. An elegant instruction-set is easy to use and therefore saves a lot of time and money. (Anyone interested in studying the matter might want to compare the PDP-11, a 16-bit computer with a brilliantly designed instruction-set, with some other 16-bit computer.)

## A Ghost Of A Chance

*A film called "The Deerhunter" told us that the Vietnam war was really about Russian roulette.*

*In it, a hero goes berserk, and commits himself to a life of Russian roulette duels, where he faces an adversary and they pass the gun back and forth till one dies.*

*He does this every week in the fleshpots of Saigon, and (according to the story) survives for one year, or fifty-two games.*

*This is a good example of powers of two, or doubling size.*

*The odds of surviving one game are*

*1/2*

*the odds of surviving two games are*

*1/2x2 or 1/4*

*the odds of surviving three games are*

*1/2x2x2 or 1/8*

*the odds of surviving 52 games are*

*1/2<sup>52</sup>, or about 1/4,000,000,000,000,000*

*—a number even smaller than the relative brain size of the screenwriter.*

## FUNDAMENTAL OPERATIONS OF COMPUTERS—A GREAT MYSTERY IS ABOUT TO UNFOLD

### YOUR BASIC COMMANDS, NOW

(Computers exist which do little more than these, and yet they can in principle do anything fancier computers can do.)

**TO BE SHOWN:** The following are the rock-bottom basic operations of computers, available as specific instructions in all computers (with some variation).

The first seven listed below will be used in the extended example in the next spread.

**LOAD** a binary pattern from core memory to a main register.

**STORE** a binary pattern in core memory from a main register.

**SEND OUT** ("OUTPUT") a binary pattern from an external device.

**BRING IN** ("INPUT") a binary pattern from an external device.

**ADD TWO** binary patterns together. (This causes them to be treated as numbers, whether they were to begin with or not.)

**JUMP**—Go to another part of the program and forget you were here.

**TEST TWO** binary patterns against each other, and branch or not in the program depending on the result.

Somehow

LOADING, STORING,  
MODIFYING AND TESTING  
BINARY PATTERNS

DOESN'T SEEM  
TERRIBLY FRAUGHT  
WITH POSSIBILITIES;

but the endless variations and ramifications make chess look like tic-tac-toe.

And part of the power, of course, is in the great speed, the teeny fraction of a second each step takes; five hundred operations yet take only about a thousandth of a second. So no matter how intricate the enactment to which these tiny steps are built, it still happens awfully fast.

### A WIND-UP CROSSWORD PUZZLE

We look at last at what really happens inside a given computer. It must be a specific computer because there is no single

inner language for all computers. For simplicity's sake (like most introductory texts) we hereby present a fictitious machine.

### THE FIDO

(Faithful Instrument, Domesticated and Obliging).

The FIDO is a twelve-bit machine. The main register (it has only one) is twelve bits long, and every memory slot is twelve bits long.

Every instruction is twelve bits long; every data word is twelve bits long, though of course much longer pieces of data can be put together by taking more than one twelve-bit word. ■

### BITS AND WHAT THEY CAN SELECT.

*In designing computers and programs, you sometimes have to decide how big to make certain data sections in terms of how many bits they can hold, and how many things that many bits can select from.*

**NOT TO BE SHOWN:** Here are the rest of the utterly fundamental commands of computers. (These are not used in the forthcoming example.)

**TEST ONE SPECIFIC** binary pattern, and branch in the program depending on the result.

**SET AN ACCESSORY IN OPERATION/TURN IT OFF.**

**REVERSE** (or "COMPLEMENT") a binary pattern—changing all the X's to O's and vice versa.

**SLIDE** (or "SHIFT") a binary pattern sidelong through a register.

**FLIPPER** (or "LOGICAL") operations between two binary patterns, especially—

**OR** (or "INCLUSIVE OR" or "IOR")—result is an X where either original pattern was an X.

**AND** (or "MASK")—result is an X only where both original patterns had an X.

### FANCY OPERATIONS

The following operations are desirable but not strictly necessary, and many computers, especially minicomputers, don't have them all.

**SUBTRACT.** (Can also be done if necessary with combination of adds and flips.)

**MULTIPLY.** (Can also be done if necessary with combination of adds, shifts and tests.)

**DIVIDE.** (Can also be done if necessary with combination of subtracts, shifts and tests.)

**MORE FLIPPER** ("LOGICAL") operations:

**XOR** (or "EXCLUSIVE OR")—result is an X only where one pattern had an X, but not both.

**NAND**—reversed AND.

**NOR**—reversed OR.

**SUBROUTINE JUMP**—"Go to another part of the program but remember this place because you'll be coming back on your own."

**RETURN FROM SUBROUTINE**—"Go back to wherever it was in the program that you last came from."

**PUSH** (on Stack machines only, see p. 78)—take a binary pattern and put it on top of the Stack.

**POP** (on Stack machines only, see p. 78)—take whatever binary pattern is now on the top of the Stack.

**ADD ONE** (or "INCREMENT")—(Useful when you're counting the number of times something has been done.)

**SUBTRACT ONE** (or "DECREMENT," not "excrement")—(Also useful when you're counting the number of times something has been done.)

**ASTRONOMICAL/INFINITESIMAL ARITHMETIC** (or "FLOATING POINT" arithmetic)—operates on a certain number of Significant Digits and keeps separate track of the decimal point—actually a Binary Point, since it's rarely if ever done decimally.

◆ Very important in the physical sciences.

*Almost any operations can be "built in." The sky is of course the limit, since any electronic operation can be added to a computer's instruction-set if desired—say, "turn on the electric blender" or "multiply quaternions"—but the former is more easily done as an output instruction, and the latter as part of a program.*

*The VAX, your Most Baroque Computer, has a huge number of complex instructions built in. (See p. 90.)*



(using the instruction codes on p. 69.)

There is a certain folk hero whom the people all call Bucky. It is said that he wears three wristwatches: one for where he is now, one for where he will be next, and one that tells what time it is at his home.

It's a pain trying to keep track of binary numbers for where things are stored.

Most people don't like this stuff.

**-Programmed  
by Mike O'Brien.**

EVERYTHING  
IS DEEPLY  
INTERTWINGLED.

## BASIC INSTRUCTIONS OF THE FIDO COMPUTER.

For a revelation of its Secret Identity, See Nearby

(Binary pattern selecting operation) OPCODE  
(5 bits)

(Binary pattern selecting where to perform operation)  
leaves 7 bits for ADDRESS

XXXXXX00000000  
(don't matter)

00X000000000  
(address goes here)

0XX000000000  
(address goes here)

XX0000000000  
(address goes here)

XX00X0000000  
(address here)

X0X000000000  
(address goes here)

0000X0000000  
(address goes here)

### OPERATION CALLED FOR

#### CLEAR AC

This instruction causes the AC to be filled with zeroes.

#### ADD (from memory to AC)

This adds the contents of the specified memory location to the contents of the AC. Result remains in the AC. Whatever was in the memory before is still there. This instruction is also used to bring a new pattern to the AC, copying it from the specified memory location; but you have to CLEAR the AC first, so you're adding it to zero.

#### STORE

This instruction copies the contents of the AC to the specified memory location. Whatever *was* in the memory location is destroyed.

Whatever was in the AC is still there too.

#### INPUT\*

This instruction copies the contents of a specified device register to the AC.

#### OUTPUT\*

This instruction copies the contents of the AC to a specified device register.

#### JUMP

This instruction makes the program follower take its next instruction at the specified address and go on from there.

#### TEST, SKIP IF EQUAL

This is a common test instruction, permitting the program to branch depending on various conditions. The contents of the AC are compared with the specified core memory location. If they are not the same, the program continues and takes the next instruction in the normal fashion. IF the two patterns *are* the same, the program follower SKIPS the next instruction and goes on to the one after.

Whatever the next instruction is, then, determines the course of events if the two patterns turn out to be the same.

For instance, that middle instruction can be a JUMP instruction, taking the program to a whole nother part of core memory and a new series of events.

\* Note: these instructions have been changed slightly to protect the innocent (you).

If you want information on the machine language and assembly language of any given machine, write the manufacturer for the *programming manual*. There may also be a *pocket card*.

## GUESS WHAT ?? ?

The FIDO is nothing but a stripped-down version of that beloved family pooch of computerdom,

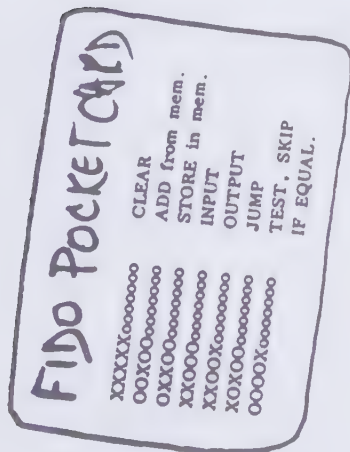
The PDP-8. (Described on p. 85.)

If you buy a PDP-8 from Digital Equipment Corporation, you get all this and more. (Except for the external devices.) And the PDP-8, of course, allows much bigger memories than 128 slots, but that's too complicated for here.) Arf.



"Assembly language programming is good for the soul."  
—Folk saying

"Assembly language programming is a terrible mistake and a waste of human resources, except under special circumstances."  
—Modern view





Data structure may consist of any conceivable symbolic representations, knitted into an overall information setup.

The idea of data structure does not apply to computers alone, but also to the information in our minds. For instance—

“Logical deduction” really consists of techniques for finding out what’s already in a data structure.

“Logical inconsistency” means a data structure contradicts itself. Rarely does it happen that a computer helps you discover something new about a subject that you didn’t suspect or see coming without the computer; after all, you have to set up a study in such a way as to make room to find things out, and you can only make room to find some things out.

## A CLASSIC *Misunderstanding*

“Computers put everything into pigeonholes.”

Wrong.

People put things into pigeonholes. And designers of computer programs can set up lousy pigeonholes. If you let ‘em. More sophisticated programming can often avoid pigeonholes entirely.

People’s naive idea of “programming” is often a reasonable approximation to the notion of “data structure.” Data structure is *how information is set*

*up*. After it’s set up, programs can twiddle it; but the twiddling options are based on how the information is set up to begin with.

# DATA STRUCTURE: INFORMATION SETUPS

One of the most common and destructive myths about computers is the idea that they “only deal with numbers.” This is TOTALLY FALSE. Not only is it a ghastly misunderstanding, but it is often an intentional misrepresentation, and as such, not only is it a misrepresentation but it is a damned lie, and anyone who tells it is using “mathematics” as a wet noodle to beat the reader with.

Computers deal with symbols and patterns.

Computers deal with *symbols of any kind*—letters, musical notes, Chinese ideograms, arrows, ice cream flavors, and of course numbers. (Numbers also come in various flavors, simple and baroque. See chocolate box, p. 71.)

*Data structure* means any symbols and patterns set up for use in a computer. It means what things are being taken into account by a computer program, and how

these things are set up—what symbols and arrangements are used to represent them.

The problem, obviously, is Representing The Information You Want Just The Way You Want It, in all its true complexities.

(This is often forbiddingly stated as “making a mathematical model”—but that’s usually in the rhetorical, far-fetched and astral sense in which all relations are “mathematical” and letters of the alphabet are considered to be a special distorted kind of number.)

Now it happens that there are many kinds of data structure, and they are interchangeable in intricate ways.

The same data, with all its relationships and intricacies, can be set up in a vast variety of arrangements and styles which are inside-out and upside-down versions of each other. The same thing

When I lived in New York, I had a driver’s license with the staggering serial number

NO 5443 12903 3-4121-37

Now it may very well be, as in some serial numbers, that information is *hidden* in the number that Insiders can dope out, like my criminal record or automobile accidents, if any. (N is my initial, and two of the digits show my date of birth, a handy check against alteration by thirsty minors. But the rest of it is ridiculous.) The fact that that leaves 15 more decimal digits means (if no other codes are hidden) that New York State has provision in their license numbering for up to 999,999,999,999,999 inhabitants. It is doubtful that there will ever be that many New Yorkers, or indeed that many human beings while the species endures.

In other words, either New York State is planning on having many, many more occupants, or an awfully inefficient code has been adopted, meaning a lot of memory space is wasted holding those silly big numbers for millions of drivers. However, that doesn’t represent a lot of money. 10 million decimal spaces these days fits on a couple of disk drives. But it’s an awful pain in the neck when you want to cash a check.

(say, the serial number, 24965, of an automobile) may be represented in one data structure by a set of symbols (such as the decimal digits 2, 4, 9, 6, 5 in that order), and in another data structure by the position of something else (such as the 24965th name in a list of automobile owners registered with the manufacturer). Furthermore, many different forms of data may be combined or twisted together in the same overall setup.

The data structure chosen goes a long way in imposing techniques and styles of operation on the program.

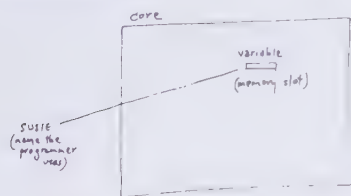
On the other hand, the computer language you use has a considerable effect upon the data structures you may choose. Languages tend to impose styles of handling information. The decision to program a given problem in a specific language, such as BASIC or COBOL or APL, either locks you into specific types of data structure, or exerts considerable pressure to do it a certain way. In most cases you can't set it up just any way you want, but have to adjust to the language you are using—although today's languages tend to allow more and more types of data.

Plainly, then, it is these overall structures that we really care about; but to understand overall structures, we need an idea of all the different forms of data that may be put in them.

## VARIABLES AND ARRAYS

The earliest data structures in computers, and still the predominating ones, are *variables* and *arrays*.

A *variable* is a space or location in core memory. (For convenience, most programming languages allow the programmer to call a variable by a name, so that he doesn't have to keep track of its numerical address.)

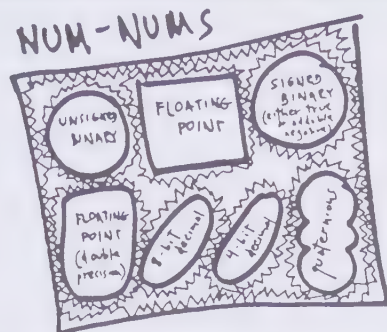


An *array* (also called a *table*) is a section of core memory which the programmer cordons off for the program to put and manipulate data in. If SPENCER is the name of the array, then SPENCER(1) is the first memory slot in it,

## IF YOU WANT NUMBERS, WE GOT 'EM

For historical reasons computers have been used mostly with numbers up to now; but that is going to be thoroughly turned around. Within a few years there may be more text—written prose and poetry—stored on computers than numbers.

During the recent massive lawsuit by Control Data against IBM, it was revealed that IBM had an awesome number of letters and communications stored on magnetic memory.



The basic kinds of number operations wired into *all* computers are few: just add (and sometimes subtract) binary numbers. However, up above the mini-computer range, a computer may have multiply, divide, and more. Fancier computers offer more types and operations on them.

**PLAIN BINARY**—Very important for counting. Represents numbers as patterns of 1s and 0s (or Xs and Os, if you prefer). How to handle negative numbers? Two ways:

**TRUE NEGATIVE**—binary number with a sign bit at the beginning, followed by the number.



Trouble is, the arithmetic is harder to wire for this kind, because there are two zeroes (plus and minus) between 1 and -1.

**ADDABLE NEGATIVE**—this system does a sort of flip and begins a negative number with all ones. It means that the machine doesn't have to have subtraction circuitry: you just add the flipped negative version of a number, and that actually subtracts it. This has now caught on generally. (It's usually called "two's complement negative," which has some obscure mathematical meaning.)

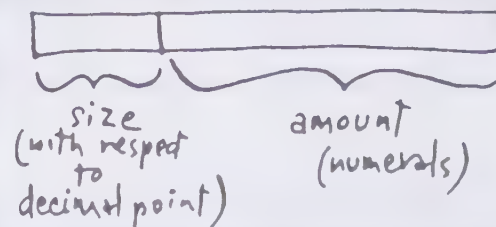
**BCD (Binary-Coded Decimal)**—the accountant's numbering system. Used by COBOL. It's plain old

decimal, with every numeral stored in four bits; the machine or language has to add them one numeral at a time, instead of crunching together full binary words.

**IMAGINARY numbers**—(two-part numbers following certain rules).

**QUATERNIONS**—(like Imaginary numbers but worse).

**FLOATING POINT**—the scientist's number technique for anything that may not come out even. Expresses any quantity as an amount and a size.



The "amount" part contains the actual binary numerals, the "size" is the number of places in front of or after the decimal point that the number starts. Very important for astronomical and infinitesimal matters, since a floating-point number can be bigger, say, than

9,876,543,210,000  
or smaller than  
.00000001234567

For some people even this isn't precise enough, so they program up "infinite precision arithmetic," which carries out arithmetic to as many places as they want. It takes much longer, though.

**BIGNUMS** is slang for any form of number that needs to get very large, beyond the limits (or accuracy) of floating point. A nice example is the *HUMBER*, or *HUMongous nUMBER*, devised by the Xanadu group. The first byte of the *humber* either contains the number (if it's 127 or less), or states how many bytes of binary the number contains. This is good for positive integers up to

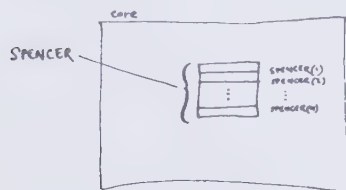
$$2^{(127 \times 8)} - 1$$

which is very large. It also has the special advantage that "zero" is only one byte, and indeed can be all zeroes.

**TUMBLERS** are large-scale forking bignums for an ever-growing scheme of numbering things, like the Dewey Decimal system, where every serial number is permanent but you can keep putting new ones in between. Tumblers were designed by the Xanadu group to work on an ever-growing repository network with more and more storage computers. There's even a *tumbler arithmetic*. (See my book *Literary Machines*, cited on p. DM 147.)



SPENCER(2) is the second, and so on up to however big it is.



The contents of a numerical *field*, or piece of data coming in, can simply be stuffed by the programmer into a variable.

The contents of a *record*, or unified set of fields, can get put into an array. The program can then pick into it for separate variables, if desired, or just leave them there to be worked on.

Then you twiddle your variables with your program as desired.

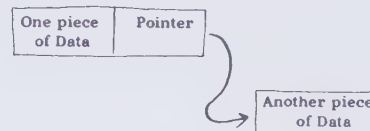
When you've done one record, you repeat. That's how lots of business programs go. Some other routine kinds, too.

## FANCY STRUCTURES

Many forms of advanced programming are based on the idea that things don't have to be sorted next to each other, or in any particular order.

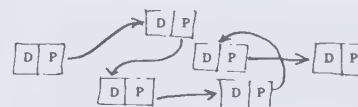
If things aren't next to each other, we need another way the program can tell how they belong together.

A *pointer*, then—sometimes called a *link*—is a piece of data that tells where another piece of data is, in some form of memory. Pointers often connect pieces of data.



A pointer can be an address in core memory; it can be an address on disk (*diskpointer*); it can point to a whole string of data, such as a name, when there is no way of knowing in advance how long the string may be (*stringpointer*).

A series of pieces of data which point to each other in a continuing sequence is called a *threaded list*.

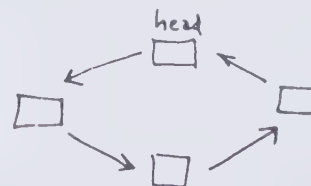


For this reason the handling of data held together by pointers—even though it may make all sorts of different patterns—is called *list processing*. (The term “list processing” might seem to go against common sense, as it might suggest something like, say, a laundry list, which is structured in a very simple blocklike form. But that's what we call it.)

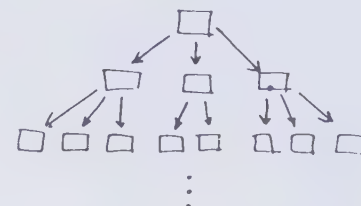
Prominent list-processing languages include SNOBOL and LISP (pp. 61-62).

Here are some interesting structures that programmers create by list processing:

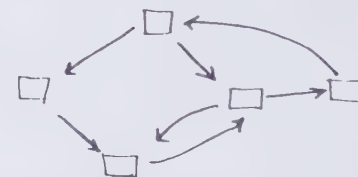
**RINGS (or cycles).** These are arrangements of pointers that go around in a circle to their first item again.



**TREES.** These are structures that fan out. (There are no rings in a tree structure, technically speaking.)



**GRAPH STRUCTURES** (sometimes called *plexes*). Here the word “graph” is not used in the ordinary way, to mean a diagrammatic sort of picture, but to mean any structure of connected points. Rings and trees are special cases of graph structures. Graph structures can go any which way.

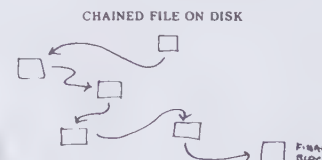


## FAST-CHANGING DATA

One of the uses of such structures is in strange types of programs where the interconnections of information are changing quickly and unpredictably. Such operations happen fast in core memory. In this kind of programming the pointers are changed back and forth in core memory, every which way, all the time. Presumably according to the programmer's fiendish master plan—if he's gotten the bugs out. (See Debugging, p. 42.)

## FANCY FILES

But these structures are not restricted to data in core memory. Complex and changeable files can be kept on disk in various ways by the same kind of threading (called “chaining” on mass storage).



Another way of handling changeable files is through a so-called *directory*

# The PUNCH CARD Mentality

Punch cards are not intrinsically evil. They have served many useful purposes. But the *punch-card mentality* is still around. This will be seen in the programmer who habitually sets things up so we have to use punch cards (when other media, or interactive terminals, would be better); who insists on the user or victim putting down numbers (when with a little more effort the program could handle text, which is easier for the human, or even look up the information in data it has already); who insists that people's last names be cut down to eleven letters because he doesn't feel like leaving a longer field or handling exceptions in his program; who insists on the outsider cutting his information into snarly little codes, when such digestion, if needed at all, could be better done by the program; and so on.

**The punch card mentality is responsible for many of the woes that have been blamed on “computers.”**



*An Idea Whose Time Has Passed* ■

## CODED DOWN DATA:

*Codes* are patterns or symbols which are assigned meanings. Sometimes we make up special codes to cut down the amount of information that has to be stored. On your driver's license, for instance, they may reduce your hair color to one decimal digit (four bits of information), since there are less than nine possibilities for quick identification of hair-color anyway.

Obviously, codes can be any darn thing: any set of symbols that is less than what you started with. But by compressing information they lose information, so that subtleties disappear (consider the use of letters A to F to grade students). When you divide a continuum into categories, not just the fewness of the categories, but the places you draw the line—called “breaks” or “cutting-points”—present problems. Such chopping frequently blurs out important distinctions. Coding is always arbitrary, frequently destructive and stupid.

Lots of ways now exist to handle written information by computer. These often present better ways to operate than by using codes of this type. But many computer programmers prefer to make you use codes.

(NOTE: there are two other senses of “code” used hereabouts: 1) the binary patterns made to stand for any information, especially on input and output; 2) what computer programs consist of, that is, lines of commands.)

block, which keeps track of where all the other blocks are stored.



But these techniques, you see, may be used in both fast and slow operations, and for any purpose, so trying to categorize them tends not to be helpful. (Note also that these techniques work whether you're dealing with bits, or characters, or any other form of data.)

A CONCRETE EXAMPLE. Suppose we want to represent the genealogy of the monarchs of England, so far as is known, in a computer data structure. NOTE THAT A DATA STRUCTURE IS DIFFERENT FROM A PROGRAM: if several programmers agree beforehand on a data structure, then they can go separate ways and each can write a program to do something different with it—if they have really agreed on a complete and exact layout, which they may only think they've done.

First we consider the subject matter. Genealogy is conceptually simple to us, but as data is not as trivial as it might seem at first. Every person has two parents and a specific date of birth. Each pair of parents can have more than one child, and individual parents can at different times share parenthood with different other individuals.

Presumably we would like a data structure that allows a program to find out who was a given person's parent, who were a given person's children, what brothers and sisters each person had, and similar matters (so far as is known by historians—another difficulty).

Note that just because it is simple to put this information in a wall chart, that does not mean it is simple to figure out an adequate data structure.

Note too, that any aspect of the data which is left out cannot then be handled by the program. What's not there is not there.

[The "right" way, of course, is to do it in LISP or Logo, making] up a data structure whose individual sections would show parentage, dates, brothers and sisters and so on.

The braver approach is to try to set it up for something like FORTRAN or BASIC, languages which treat core memory more like a numerically addressed array or block, as does rock-bottom machine language.

Let us assume that we have decided to use an array-type data structure, for instance, to go with a program in the BASIC language on a 16-bit minicomputer. We do not have much room in core memory, so for each person in our data structure we are going to have to store a separate record on a disk memory, and call it into core memory as required.

After much head-scratching, we might come up with something like the following. It is not a very good data structure. It is not a very good data structure on purpose.

It uses a block of 28 words, or 448 bits, per individual, not counting the length of their name, which is an additional 8 bits per character or space. However, this in itself is neither good nor bad. It's more than you might expect, but less than you might need.

(Incidentally, out of concern for storage space, some data fields are packed more than one to a 16-bit computer word. This is scornfully called bit-fiddling by computerfolk who work on big machines and don't have to worry about such matters.)

individual's own (name)	1.	monarch no. (if any) sex - (1 bit)	
	2.	serial no.	
	3.	stringpointer → (to name area)	
	4.	(two 16-bit words long)	
mother	5.	serial no.	
	6.	serial no.	
	7.	serial no.	
	8.	(up to five)	
father	9.		
	10.		
	11.		
	12.	serial no.	
brothers (up to five)	13.		
	14.		
	15.		
	16.		
sisters (up to five)	17.	start (11 bits)	no. months
	18.	start (11 bits)	no. months
	19.	serial no.	
	20.		
female children up to five	21.		
	22.		
	23.		
	24.		
male children up to five	25.		
	26.		
	27.		
	28.		

As explained already, that was the basic block. We still have to keep the names somewhere, in a string area. Whether to keep this in core all the time, or on disk, is a decision we needn't go into here.

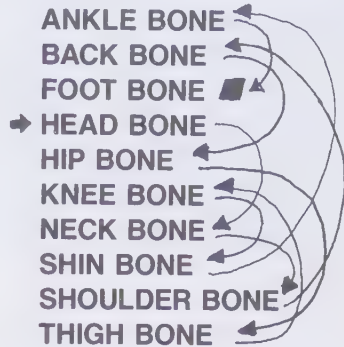


Here are some assumptions I have embodied in this data structure. That is, I had them in mind. (The parts you didn't have in mind are what get you later.)

- Parents and children of monarchs are included, as well as monarchs.
- All monarchs have a separate monarch number.
- No monarch reigned more than twice. (?)
- No monarch or parent of a monarch had more than five children of one sex. (Note the danger of these assumptions.)
- We are not interested in grandchildren of monarchs unless they are also monarchs, or siblings, or parents of monarchs.
- The information about the different people can be input in any order, as the years of reign can be stepped through by a program to find the order of reign.

If this seems like too much bother, that is in a way the point. Data structures must be thought out. Since computers have no intrinsic way of operating or of handling data (though particular languages will restrict you in particular ways), you will have to work all this out, and a carelessly chosen data structure

Remember the song that had a pointer data structure? (in alphabetical order)



## ACCOUNT NUMBERS

In principle we no longer need account numbers.

Now that text processing facilities are available in most (if not all) major computer languages, the only excuse for not using these features is the programmer's notion of his own convenience—not that of the outside customer or victim. Example. Someone I know got brand new American Express and Carte Blanche credit cards. He made no note of their numbers. Then he lost them both. Duly he reported the losses. Neither service could look him up, they said, without the numbers. Not having used them, he had no bills to check. Even though he was the only person at that address with anything like that name. And why not, pray tell? Either because they were fibbing, or because they had not seen fit to create a simple straightforward program for the purpose. (See Basic Rejoinder, p. 39)

I have heard of similar cases involving major life insurance companies. Don't lose the numbers. Let's all dance to it:

WHEN ANYTHING IS ISSUED TO YOU, WRITE THE NUMBER DOWN.



## SOMETIMES IT JUST SITS THERE SOMETIMES IT COMES AND GOES.

Data usually has to be marshalled into rows, or even regiments and battalions, before it can go into a computer.

We all have the same experience, whether writing something down or trying to keep computer disks under control: *THE AMOUNT OF DATA IS IRREGULAR WITH RESPECT TO THE SPACES WE ALLOW FOR IT. From this come many woes. Or rather, from having expected data to line up regularly—a tradition of old-style programming—come many woes.*

Marshall McLuhan pointed out that the desire to line things up in even rows—which he thought was a Protestant trait—created military marching formations and grid cities. Also, he might have added, bad data structures.

"ASCII and ye shall receive"  
● The Industry

An obscure and intricate study of the interchangeability of data structures—how they fundamentally interconvert—has been the longtime research of one Anatol Holt, who calls his work Mem-Theory. Mem is from memory, and also, conveniently, a Hebrew letter.

This is an extremely ambitious study, as it in principle embraces not just much or all of computer science, but perhaps mathematics itself. Math freaks attention: Holt has said he intended to derive all of symbolic logic and mathematics from relations and pointer structures. Let's hear it for turning Russell on his head.

### WHERE ARE THEY NOW?

*Holt is hard to get ahold of, and his interesting game of Mem, which dealt with relational transposition, is no longer available. I just ran into him, though, after twenty years, and he says it may be reissued.*

## Data Media

A *data medium* ("medium" is the singular of "media") is anything that holds the marks of data outside the core memory of a computer. Thus punched cards and punched paper tape may be used as *input media*, used for putting information into a computer. (Each medium needs a corresponding *input* or *output device*, to whisk across the surface and translate its marks or holes into the corresponding electronic pulses.)

There are three types of data media: *input*, *output* and *storage* media. An input medium carries the data in. An output medium receives the results of a program; for instance, a sheet of paper coming out of a printing device.

Storage media are output media that may be used as input media later on.

*The units and arrangements of data used for input, output and storage are in principle not necessarily the true ones of the data structure used by the program. The blocks and records of storage, for instance, may have irregular data with pointers sitting in them. (Unfortunately there is some carryover, in that programmers are tempted to use data structures which are easy to store and run in and out, rather than handling the true complexities of the subject. This is always a temptation.)*

will leave something out, or fail to distinguish among important differences, or otherwise have its revenge.

(For instance, if you haven't noticed yet: we left out *legitimacy*. For many purposes we want to know which kings were bastards.)

(Self-test: is five bits long enough to express the greatest number of months any English monarch reigned?—see "Binary Patterns," p. 65. Or do we have to fix this data structure on that score also?)

To give you a sense of the sort of program this data structure allows:

A program to ascertain how many kings were the sons of kings would look at each entry that had a monarch number, test whether the monarch was male, and if male, would look at the male parent's serial number. Then it would look up that parent's entry, and see whether it in turn had a monarch number, and if so, add one to the count it was making. Then it would go back to the entry it had been looking at, and step on to the one after that.

This is actually a pretty lousy data structure. The clumsiness of this approach to such data—and you are welcome to think of a better one—shows some of the difficulties of handling complex data about the real world. Things like lengths of names and numbers of relatives produce great irregularities, but make these kinds of data no less worthy of our attention.

We could add lots of things to our data structure (and so make it more unwieldy). For instance, we might want to mark each serial number specially if it referred to someone who was the offspring of a monarch. We could simply set a particular bit to 1 in the serial number for them (called a *flag* or *tag*). We could also flag dates and genealogies that are regarded as uncertain. *There is no limit to the exactness and complexity with which information may be represented.* But doing it right can, as always, be troublesome.

A lot of computer people want to *avoid* dealing with complex data; perhaps you can begin to see why. But we must deal with the true complexities of information; therefore languages and systems that allow complex information structures must become better-known and easier to use.

## THE FRONTIER: COMPLEX FILE STRUCTURE

The arrangements of whole files—groups of records or other info chunks—are up to the programmer. The structure of files is called, not surprisingly, *file structure*, and it is up to the programmer to decide how his files should be arranged.

Habits die hard. The notion of sequence—even false, imposed sequence—is deep in the racial unconscious of computer people. An interesting concrete term shows this nicely. Because computer people often think any file should have a basic sequence, they use the term *inverted file* for a file that has been changed from its basic sequence to another sequence. But increasingly, all the sequences are false and artificial. Where now are inverted files? All files are inverted if they're anything.

Fortunately, the final frontier of data structure is now increasingly recognized as the control of complex storage of files on disk memory. The latest fancy term for this is *data base system*, meaning planned-out overall storage that you can send your programs to like messengers.

The fact that IBM now has moved into this area (with its intricate "access methods" and all their initials) means complex storage control has finally arrived, although the pioneering work was done by Bachman at GE some years ago. Till the last few years, external storage, with pointers and everything, has not been conveniently under the programmer's control except in crude ways. Finally we are seeing systems beginning to get around that automatically handle complex file structures in versatile ways that programmers can use more easily.

## MAGNETIC STORAGE

The principle of fields applies in data media, especially magnetic tape and disk. We may extend the notion of a field to explain records and files.

A *field*, generally speaking, is a section of positions on some medium reserved for one particular piece of information, or the data in it.

A *record* is a bunch of fields stored on some medium which have some organized use. (For instance, the accounting information held by an electric utility company about a particular customer is likely to be stored as a record with at least

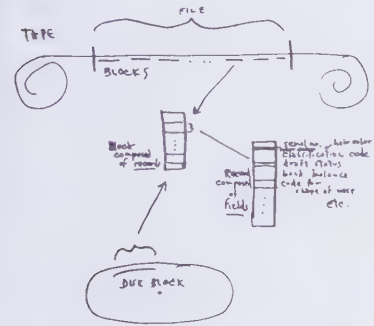


these fields: account number; last name; initials; address; amount currently owed.)

A *file* is a whole big complete bunch of information that is stored someplace. In many applications a file is composed of numerous similar, consecutive records. For instance, an electric company may well store the records for all of its customers on a magnetic tape, ordered by account number (account 000001 first).

Storing sequences of similar records in long files is typical of business programs, though perhaps this should begin to change. It's especially suited to batch processing, that is, handling many records in the same way at the same time.

Now, the divisions of field, record and file are conceptual: they are what the programmer thinks about, based on the information needs of a specific computer program.



**BLOCKS, TRACKS AND SECTORS** are the divisions made by storage methods inside the hardware; they have nothing to do with the structure of your data.

*Files* are the things you save that have to be named and kept track of, and ain't it a pain in the neck? Different kinds of files are allowed (or required) under different operating systems.

### TRADITIONAL CONVEYER-BELT PROGRAMS: BATCH

Many traditional business programs are of this type, reading in one data record at a time, doing something to it (such as noting that an individual has paid the exact amount of his gas) and writing out a new record for that customer on the current month's tape.

### THE PROBLEM

Standardized fields, blocks and records are often necessary or convenient.

But, on the other hand, the kinds of computer programs people find oppressive often have their roots in this kind of data storage and its associated styles of programming, especially the use of fixed-field records as the be-all and end-all. The more interesting uses of the computer (interactive, obliging, artistic, etc.) use a greater variety of data structures.

### INPUT AND OUTPUT CODES

Data has to get inside the machine somehow, and results have to get back out. Two main types of codes—that is, standardized patterns—exist, although what forms of data programs work on inside varies considerably. (The input data can be completely transformed before internal work starts.)

1. ASCII (pronounced “Askey”), American Standard Code for Information Exchange. This allows all the kinds of numbers and alphabets you could possibly want (for instance, Swahili) for getting information in and out of computers. ASCII is used to and from most Teletype terminals and keyscopes.

However, ASCII is also used for internal storage of alphabetical data in many non-IBM systems, and it is also the running form of a number of programming

languages. IBM's deliberate undermining of the ASCII code is a source of widespread anger. (See IBM, p. 140.)

*Just to show us how crazily things can evolve, the ASCII transmission code has gone in at least two directions that would have been impossible to predict. One is NAPLPS, a coding system that extends ASCII to graphics (originally developed in Canada under the name Telidon), which may be seen delivering slow interactive graphics—usually maps—in various shopping malls.*

*The other is MIDI, a system for tying together music synthesizers and related equipment (see p. DM 62).*

*ASCII has become the standard code for sending and storing alphabetical material (except at IBM installations).*

*ASCII was made big enough for upper and lower case, but it's bursting at the seams as they try to cram more and more of the world's spoken and written languages and symbols into its framework.*

2. EBCDIC (pronounced “Ebsadick”), Extended Binary Coded Decimal. This was the code IBM brought out with the 360, passing ASCII by. (IBM seems to think of compatibility as a privilege that must be earned, i.e., paid for.) EBCDIC also allows numbers, the En-

glish alphabet, and various punctuation marks. This is used to and from most IBM terminals (“2741 type”).

And Also:

HOLLERITH, meaning the column patterns that go in on punched cards. (They can also come out that way, if you want them to.)

CARD-IMAGE BINARY. If for some reason you want exact binary patterns from your program, they can be punched out as rows or columns on punch cards.

STERLING. Just to show you how comical things can get, the original PL/I specifications (see p. 60) allowed numbers to be input and output in terms of Pounds, Shillings and Pence (12 pence to the shilling, 20 shillings to the pound). No provision was made for Guineas (the 21-shilling unit), or farthings, unfortunately.

### YOU AND YOUR FILES

*Whether you are a corporation or an individual, planning, setting up and maintaining computer storage is a mess.*

*First there is the question of making programs work together. This is usually done by arranging for the files to be compatible.*

**“There is a growing feeling that data processing people would benefit if they were to accept a radically new point of view, one that would liberate the application programmer's thinking from the centralism of core storage and allow him the freedom to act as a navigator within a database. ... This reorientation will cause as much anguish among programmers as the heliocentric theory did among ancient astronomers and theologians.”**

**—Charles W. Bachman, “The Programmer as Navigator.” CACM Nov 1973.**

**Bachman was the prime mover in the development of large linked disk data systems at General Electric; he is the Pioneer. This is about big n-dimensional stuff.**

**“ASCII not what your computer will do for you.”**

**—IBM**

**Note:** By decent standards of English, the word *data* should be plural, *datum* singular. But the matter is too far gone: *data* is now utterly singular, like “corn” and “information,” a granular collective which may be scooped, poured or counted.

**But I draw the line at media. Media are many, “media” is plural!**

### 16-Bit Code For Text

*Some have proposed using 16-bit codes for text. That would certainly take care of the question of where to put underlines, boldface, etc. There are 64K different possibilities. Supposedly they are doing this in Japan. However, the situation over there is very different, since they use the Kana alphabet, the English alphabet and the Chinese alphabet.*



Expect computer programs to be incompatible. Sometimes, however, it is possible to get them to work together, using the output of one as input to another. (Under the UNIX operating system this method of connection is especially convenient, but UNIX is not yet common in the personal computer world.)

In the personal software world, the output files from one program can *sometimes* be used as the input files to another.

For instance, getting files from one word processor to go into another word processor is not necessarily easy.

For instance, almost all spreadsheet programs will accept files created by Lotus 1-2-3, but not vice versa.

Database programs are worse. A lot worse.

But the grimmest aspect of dealing with computers is **file management**—naming and keeping track of all the pieces of your work, which may reside on separate disks or hard-to-find places called **directories** or **catalogs**.

Not only do you have to name them. You have to **back them up**, making sure that all the stuff you're saving that has been recently changed is safe in its present form.

Then there is the problem of **master file update**, or making sure that small changes that have arrived from different places are correctly assimilated to some main file.

Then there is the issue of **historical archiving**, setting things up so you can later check the way things *were*. This is far easier with paper than with today's computer systems.

All these are administrative issues—but only because a **technical** solution to unburden us from these problems has not reached the hands of the people.

A sweeping solution is the Xanadu system—see p. DM 141—but that is not yet commercially available.

#### BIBLIOGRAPHY

Malcolm C. Harrison, *Data-Structures and Programming*. Scott, Foresman, 1973.

This book can be recommended to ambitious beginners. It has useful summaries of different languages, as well as fundamental treatment of data structures as they intertwine with specific languages.

data, damyata, dhayadvam - T.S Eliot, The Wasteland

What the computer *really* is.

## COMPUTER ARCHITECTURE The Nuts and Bolts

Computers are basically alike. Ignore their appearances; a roomful of roaring cabinets may have a great deal in common with a small blinking box; indeed, they may have the same architecture, or structure, and therefore be the same computer.

A computer, then, internally just consists of certain places to work on information (main registers), certain places to keep it the rest of the time (memories), certain pathways and interconnections between them, an instruction-set having certain powers whose instructions can be operated on out of memory, and a program follower that carries out the instructions of that instruction-set.

The structure of computers, in their glorious similarities and fascinating differences, is called *computer architecture*.

*The fact that computer innards are on chips these days confuses this a little. The world of computer architecture has changed immensely since the original Computer Lib, but only through re-packaging: most computers are now built*

*with standard commercial micro-processors, program followers on a chip. (Only a few computer builders design their own architectures from other than existing chips, and that for special purposes—such as the Cray, the LISP machine, and a few others.)*

*But microprocessors are simply pre-fabricated computer sections, actually built out of the same structures as before. So let's consider in this section what the structures of computers are, then later (p. 96) what packaged parts they come in now.*

#### REGISTERS AND MEMORIES

Computers are made, basically, of two things, *registers* and *memories*. A register is where something happens to information; a memory is where nothing happens to information. Let's go over that slowly.

A register is a place where something happens to information: the information can be flipped around, tested, changed by

*The term core memory has been used since earliest times to mean the main fast memory inside a computer, as distinct from the outside accessory memories, like disks and tape.*

*In recent years the term has gone out of fashion, since some people consider a "core" to be a specific obsolete hardware way of holding information. Instead, they use terms like "main memory" or "RAM."*

*This is silly, and is a manifestation of the engineer's wooden ear for language. Different storage technologies come and go, but as long as there is fast memory in the middle—core also means "middle"—then as far as I'm concerned the name for it continues to be CORE MEMORY.*

*(Carry around a copy of this item if you want to start arguments with loutish engineers.)*

A NOTE ON "CORE MEMORY"

arithmetic, or whatever. (We noted earlier that registers are what connect a computer to its accessories. They are also principal parts of the computer itself.)

A memory is a place where nothing happens to information. A program puts the information there, and there it stays till some program pulls it out again or replaces it.

A main or *general register* (often called the *accumulator*, for no good reason) is where the program brings things to be worked on, tested, compared, added to and so on. There can be several of them in a computer.

Other registers perform other functions in the computer; a given computer's design, or *architecture*, is largely the arrangement of registers and the operations that take place between them.

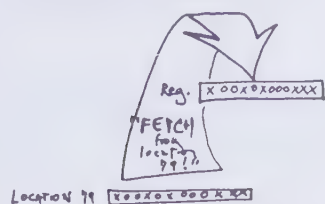
The reason we don't just have all registers—and no memories at all—is that registers traditionally cost more than memories.

Memories come in all sizes and speeds. So lots of computers have big slow memories, such as disk memories, along with their small fast memories.

A memory consists of numerous holding places or *storage locations*, each holding one standard piece of information for the computer, a *word* having a specific number of bits (see pp. 65-66). We must stress: a "COMPUTER WORD" HAS NOTHING TO DO WITH ENGLISH WORDS OR ALPHABETICAL CHARACTERS. The term refers to a specific machine's standard memory slot, having a fixed number of bit positions.

One important reason for this standardization is that each holding place, or memory location, can be given a number or address. If every slot in the memory has an address, information can be stored in specific places:

"FETCH from location 79!"



A core memory has a definite rhythm or *cycle*, into which it divides the passing time. The memory cycle of a core memory is so important that its duration is often called the cycle time of the computer. A request to the core memory made at the beginning of the cycle is honored at the end of the cycle. Core cycles are very fast, being these days about one microsecond, or millionth of a second.

A core memory can only perform one act (store or fetch) during one memory cycle.

Core cycles during which nothing is requested of the memory simply go by.

One last point about core memories. The number which specifies an address to the memory is a binary pattern—just like all the other information. (Or more exactly, whatever binary pattern is supplied to the memory as the address to store or from which to fetch, that pattern will be *treated* as the address to store or from which to fetch, that pattern will be *treated* as a binary number whether it was supposed to be or not. It could be the alphabetic word GRINCH which got there by mistake (see Debugging, p. 42), but the memory will treat it as an address number and go to the address specified by that pattern.

## THE ROCK BOTTOM PROGRAM FOLLOWER

How, you ask desperately, does this innermost program follower work? The one that is built into the computer?

Aha.

Basically it consists of two specific registers, the Program Counter (usually abbreviated PC) and the Instruction Register (usually abbreviated IR), and other electronic stuff, loosely termed "decoding logic."

(Since we are already visualizing the program follower as a little *hand*, let's think of the index finger as the program counter and imagine that the thumb can flip an instruction into a little cup, the Instruction Register or IR. What the heck.)

WHEN a program is set into operation, the binary pattern specifying its first

address in memory is put into the program counter.

Then the instruction at that address is fetched to the program follower (that is, put in the instruction register), decoded and carried out.

THEN THE PROGRAM COUNTER AUTOMATICALLY HAS ONE ADDED TO IT, SO IT POINTS TO THE NEXT INSTRUCTION.

The instruction pulled from memory is held in the command or instruction register and there decoded by the system's electronics.

It is of no concern to the programmer *how* this is done electronically. (And indeed electronics is generally of little concern to computer people, unless they are trying to design or optimize computers or other devices themselves. Indeed, the

electronic techniques are constantly changing.)

All we need to know is that an electrical decoding system (called the *logic circuits*) carries out the specific instruction—for instance, by shutting off the path to the memory, turning on the adding circuit, and opening paths through the adding circuit and back to the main register.

Now that the program counter holds the number of the next instruction, it in turn is accordingly fetched and executed.

And so it continues.

When an instruction calls for a jump or branch in the program, what happens?

The jump command causes a new number to be stuffed into the program counter, that's what, and so that's where the program goes next. ➡

## Core Memories Are Made of This

Memories come in many sizes and speeds, to be tailored to an application. You should know the differences between:

ROM: Read-Only Memory. Contents can't be changed, costs less than changeable (at any given speed).

RAM: Random-Access Memory. Also called read-write memory. Same as core memory: May have its contents changed.

EL6RMM: Read-Mostly Memory. You can get out its contents fast, but change them only very slowly.

## EXTERNAL MEMORY

*This is where you put information when it overflows core, or when you want to take it off the machine for storage.*

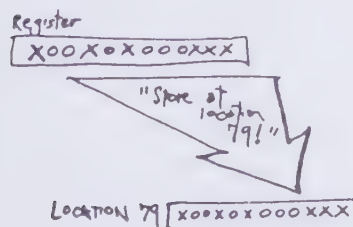
DISK: Platters of magnetized material that go around and around. May be erased and used over. Information is put on them by means of read-write heads, which either read information from them or write information onto them.

TAPE: Holds much more and costs less, but takes much longer to get at. Computer tape machines, unlike audio tape machines, can be reversed automatically while in use, to read material again, write over it, or whatever.

LASERDISC: A laser is a device producing a beam of ultracontrollable light. The beam can be made very tiny. So-called videodiscs (see p. DM 60) contain television that has been recorded on disc with a laser (at the factory). The TV signal is likewise read off the disc with a laser. (Note that for no good reason magnetic computer disks are spelled with a K, videodiscs with a C. How to spell it now that combined units are available is a question.)

The computer laserdisc is a new animal, but one long anticipated: it writes with a laser and reads with a laser, but the bits can only be written once. So you start with a big memory all zeroes and fill it up, adding byte after byte. If there is something you no longer need, you arrange to ignore it. If you don't want anybody to see it, you write all ones over it, placing it in final stagnation. Vast amounts of stuff can be stored very cheaply on such discs.

These are now called WORM disks (Write Once, Read Many) or sometimes WOMB (Write Once, Many Back).



Register  
"store at location 79"  
Location 79

and gotten back out of specific places:



# Then What Are The Differences Between Computers

The word length (number of bit-spaces in a main register and memory slot);

The number of main registers and what they can do; i.e., how they are set up and what operations can take place in and among them; i.e., the Instruction Set (see nearby);

The amount of memory;

The accessories or peripherals;

The cycle time.

EVERYTHING IS DEEPLY INTERTWINGLED.

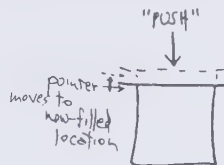
## THE MAGIC OF THE STACK



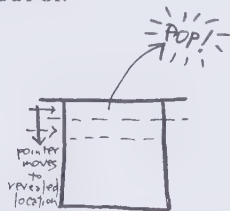
The Stack is a mechanism—either built into the computer (“hardware”) or incorporated in a program (“software”) which allows a computer to keep track of a vast number of different activities, interruptions and complications at the same time.

Basically, it is a mechanism which allows a program to throw something over its shoulder in order to do something else, then reach *back* over its shoulder to get back what it was previously working on. But no matter how many things it throws over its shoulder, everything stays orderly and continues to work smoothly, till it has resumed everything and finished them all.

It goes like this: if the program has to set aside one thing, it puts that one thing in core memory at a place specified by number called a *stack pointer*. Then it adds one to the stack pointer, to be ready in case something else has to go on the stack. This is called a **PUSH**.



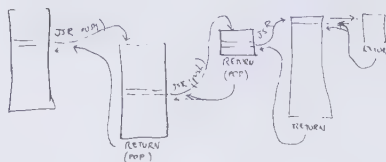
When a program is ready to resume a previous activity, it subtracts and fetches whatever that stack pointer points to. This is called a **POP**.



It may not be immediately obvious, but this trick has immense power. For instance, we may stack any number of things together—the addresses of programs, data we are moving between programs, intermediate results, and codes that show what the computer was doing previously.

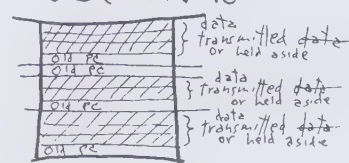
Using stacks, programs may use each other very freely. It is possible, for instance, to jump among subroutines—

independent little programs—willy-nilly, using a stack to keep track of where you've been.

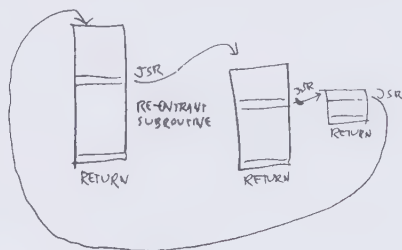


In this case the stack holds the previous locations and intermediate data, so that the program follower can go back where it came from at the end of each subroutine.

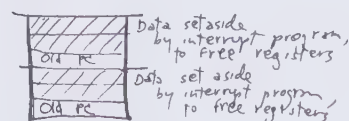
### STACK SUBROUTINING



This even makes possible “re-entrant” programs, meaning subroutines that can be used simultaneously by different programs without mixup, and “recursive” programs, meaning programs that manage to call themselves when they themselves are in progress.



Stacks are also used for handling “interrupts”—signals from outside that require the computer to set aside one job for another. Having a built-in *hardware* stack enables the interrupts to pile up without confusion:

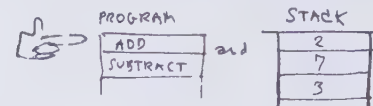


Finally, *stack arithmetic*, like that done on the Burroughs 5500, enables arithmetic (and other algebraic types of activity) to be handled without setting aside registers or space in core memory.

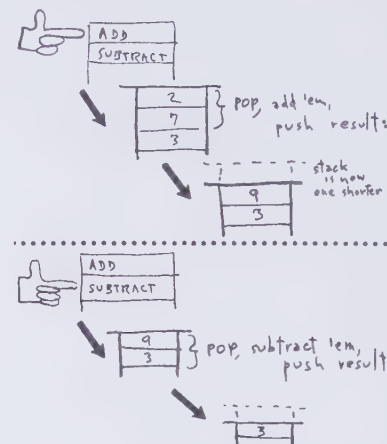
As a simple-minded example on a hypothetical machine, suppose we wanted to handle

$$2 + 7 \times 3$$

On this machine, let's say, this gets compiled to a program and a stack:



Then the operations are carried out on the stacks itself:



Stack programming tends to be efficient, particularly in its use of core memory.

Some computer companies, such as IBM, resolutely ignore stack architecture, though hardware stacks have become widely adopted in the field.

Some languages, such as ALGOL and TRAC Language, require stacks.

*FORTH* is a stack language and uses two of them constantly; *C* makes it easy to set up all the stacks you like. But if these languages run on machines that don't have hardware stacks, they have to take care of stacking in software instead. That still leaves the programmer free to concentrate on *using* the stacks, without having to build them from scratch.

## COMPUTER DESIGNS

The particular structure of registers, memories and pathways among them is called the *architecture* of a computer. The binary instructions available to the programmer are called the *instruction-set* of the particular computer. (The word “architecture” is often used to cover both, including the instruction-set as well, see p. 84.)

The principal variations among computers are the word length (in bits—see “Binary Patterns,” see p. 65) and the number and arrangement of main registers. Then come the details of the ways in which items are selected from core memory—the addressing structure. Then the instruction-set, whose complications and subtleties can be considerable indeed.

The individual computer is the complex result of all of these. If they fit together well, it is a good design. If they fit together poorly, it is a bad design. A bad design is usually not so much a matter of overt stinky features as of ramifications which fit together disappointingly. (*Glitch* is a term often used for such stinky features or relationships.)

The possible ways of organizing computing hardware are vast, and only partly explored. (See p. 76.)



This is a PDP-11, one of the world's best-designed minicomputers (see p. 90). The PDP-11 is a 16-bit machine. Shown is Model 45, the fastest PDP-11, which has various special features. Stripped, with 4K of core memory (that's 4096 locations), it costs about \$13 grand. A smaller PDP-11 goes for some \$5000.

## The MINI

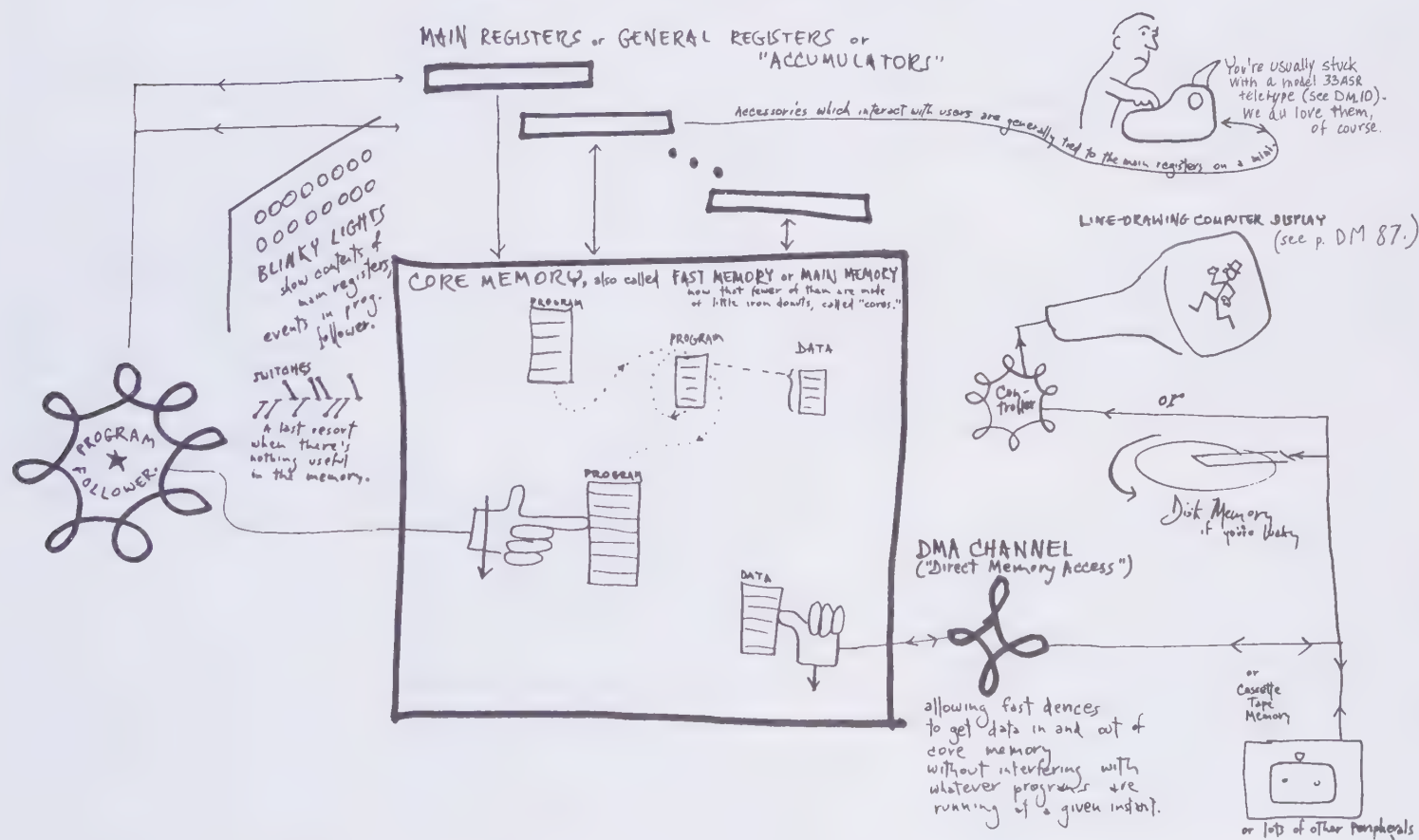
A minicomputer simply means a small computer, no different in principle from the big ones (to be discussed shortly), and it can do all the same things except as limited by speed and memory capacity.

(Mind, we are talking about real *computers*, not the little calculators you hold in your hand that just do arithmetic. A real computer is one which works on stored programs and all kinds of data, working not merely on numbers but on such other things as text, music and pictures if supplied with appropriate programs; see flip side.)

There is some argument over what constitutes a minicomputer; basically we will say it's any computer with a word length of 18 bits or less (see “Binary Patterns,” p. 65). (Some companies, like Datacraft and Interdata, are trying to

## BASIC DESIGN OF SIMPLE COMPUTER or MINICOMPUTER

Bigger computers are the same but more so.





## DINKIES: an overview

There is great confusion as between various types of small computer, with the latest stupid term, "microcomputer," adding to the confusion. We have:

### minicomputer or mini

Traditionally, any computer having an architecture (memory and main registers) of 18 bits or less. Lately, unfortunately, some people have been advertising their 24-bit and even 32-bit computers as minis. This is just confusing.

(They base this on the fact that "minicomputer" has also referred to a machine sold without a lot of programs. But that's really a separate issue.)

### microprocessor

Any program follower sold on a chip. At this writing, the popular ones are the 8086 and 286 series series from Intel and the 68000 series from Motorola.

### microcomputer

Crummy term apparently being used to mean any tiny computer, regardless of its structure. Thus all computers will be "microcomputers" in a few years. This clarifies nothing as to their structure or use.

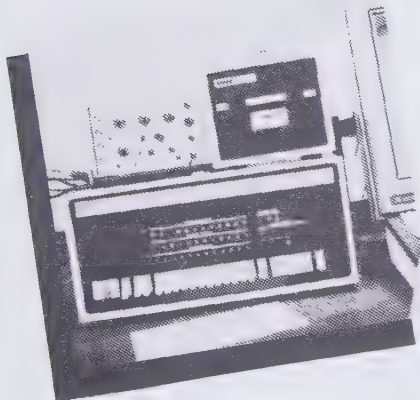
### midi computer

Remember midi skirts? Well, this term has been used for computers larger than 16 bits or faster than usual, by people seeking to give the impression that their machines are bigger than minis and less than biggies. Even the PDP-10 (a genuine biggie) has sometimes been called a midi.

## WHAT DO THESE SIZE DESCRIPTIONS MEAN EXACTLY?

Not much exactly. Compare SUBCOMPACT, COMPACT, STANDARD, MIDSIZED, FULLSIZED and LUXURY CARS.

We know SORT OF what these terms mean, but they are twisted for whatever advantage by whoever needs to.



The good ol' PDP-8, perhaps the most popular minicomputer (12 bits). Shown here with a cassette tape deck and a screen display (see *Dream Machine* side). Courtesy Princeton University & R.E.S.I.S.T.O.R.S. (see p. 110)

peddle their worthy computers as "mini-computers" even though they're 24 and 32 bits, respectively, but that's very odd. Interdata says any computer under ten thousand is a mini—which means *all* computers will be minis by and by; a vexing thing to do to the term.)

Traditionally, minicomputers come with much less. In the old days pretty much all the programs you got with it were an assembler (see p. 80) and a debugger and a FORTRAN compiler (see p. 59) if you were lucky. Today, though, with minis having highly built-up software like the PDP-8, the PDP-11 and the Nova, you can get a lot of *different* assemblers; together with FORTRAN, BASIC, and a little disk or cassette operating system (see p. 99) to make your life a little easier.

**The idea of owning a computer may seem strange to some people, but with prices falling as they are it makes perfect sense.** Numerous individuals own minis, and as the price continues to drop the number will shoot up. For several families with children to pool together and buy one for the kids makes a lot of sense. One friend of mine has an 8, another is contemplating an 11. (I've been trying to get my own for years; perhaps this book...) Anyhow, the general price range is now \$3000 to \$6000 plus accessories, and that's dropping fast. Rental is usually a great mistake: prices are very high and after six months or so you'll have paid for it without owning it. (But names of rental places will be found in this book, and some of them may offer good arrangements.) Minis may now be had in quantity for \$1000 each—price of the PDP-8A in May 1974—and soon that will be the consumer price.

Unfortunately, the price of the computer itself is dropping faster than that of the accessories, such as the basic terminal you'll need, which still weighs in at \$1000-\$5000. Moreover, as soon as you want to do anything serious you'll need a disk (starting around \$4500) or at *least* a cassette memory (starting around \$1500). **But these prices too will come way down as the consumer market opens.**

*Nobody uses a terminal anymore, since you can now use a \$500 computer (which comes with its own screen and keyboard). And you can also use your per-*

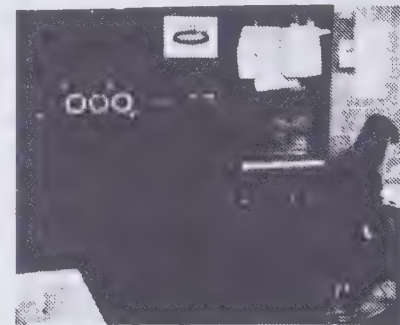
*sonal computer to connect to a bigger one, the way terminals used to. Disk drives start at a couple of hundred dollars; so do printers.*

**Some of us minicomputer freaks see little real need for big computers.** Minicomputers are splendid for interactive and "good-guy" systems (see p. DM 10); as personal machines, to handle typing and bookkeeping; even for business systems, if you recognize the value of working out your own in BASIC or, say, TRAC Language.

Minicomputers are being put inside all manner of other equipment to handle complex control. (However, for repetitive simple tasks, the latest thing is microprocessors (see p. 96), which cost less but are harder to program.)

Minicomputers are now being found in high schools; active marketing to high schools is now being done by both DEC and Hewlett-Packard.

Children's museums in Brooklyn and Boston have recently obtained PDP-11s for the kids to interact with. In the future, networks of minis may be the systems to offer low-cost information services to the home (for speculations, see p. DM 141). **But minis will also start to make bigger and bigger incursions on the territory of the big machines.**



Here's that selfsame PDP-11 in its overall setting. With peripherals shown, plus the magnificent Vector General display (see p. DM 90), this setup cost well over a hundred grand. (This is the Circle Graphics Habitat, otherwise known as the Chemistry Department Computer, U. Illinois at Chicago Circle. Why do chemists need such things? See p. DM 91 and p. 151)



The operator muses at the console of the main computer at the University of Illinois at Chicago Circle. It is an IBM 370 model 158, which rents for about \$50,000 a month, including all accessories and a dozen or so terminals—in the parlance of big-computer people, a “medium-sized installation.”

## THE BIGGIE

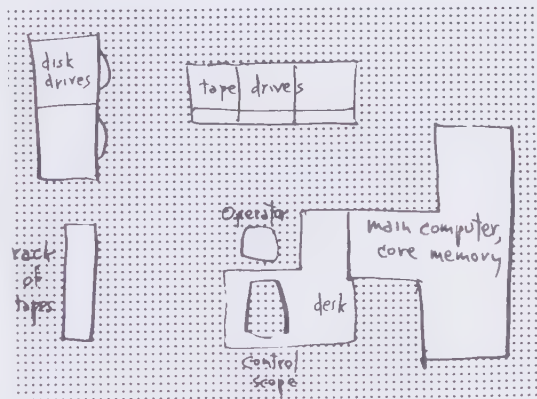
This is a big computer.

In principle it's no different from a small one; but it has bigger memories, more registers, more program followers. There are more specialized parts and more things happening at once. (Thus the term “digital computer complex” is sometimes used for a big computer.) It comes supplied with a monitor program or operating system (see p. 99) and a variety of other utility programs and language processors.

Biggies have many ominous and seemingly incomprehensible things to scare the layman.

For one thing, *where is the computer?* All you see is a lot of roaring cabinets. Which is it?

Answer: all of them. “The computer” is divided among the different cabinets (note diagram and cluster of pictures locating the operator among them, next column). The external devices or peripherals (see p. 50) are usually in separate housings. Usually there is one single box or “mainframe” containing core memory, main registers, program-following circuitry, etc., as in the machine



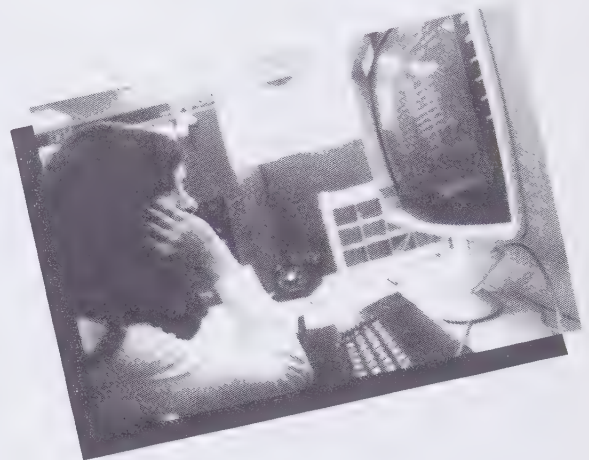
illustrated, but these things don't *have* to be in one box, and sometimes aren't.

The parts of a computer are set up to be *gotten at*, to be refilled and repaired. Their innards swing open like refrigerators. Similarly, the wiring of computers is in separate sections or modules (“module” merely being today's stylish term for “unit”), having very orderly connections among them. Individual circuits are on circuit sheets or “cards” which plug in sideways and may be replaced easily.

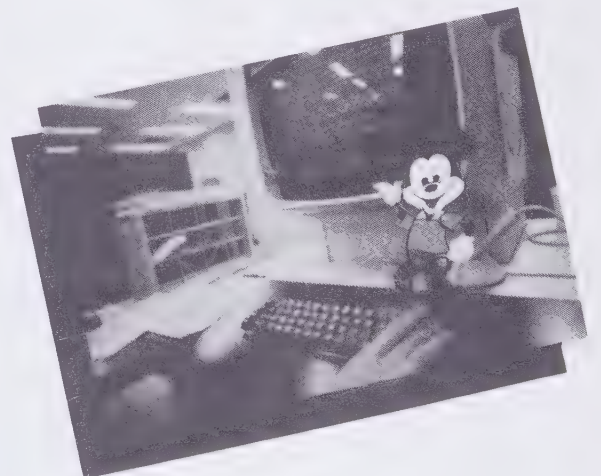
## AN OPERATOR IS NOT A PROGRAMMER

Cindy Woelfer is the day-shift operator of Circle's big computer. The job mainly consists of changing disks and tapes, starting and stopping different jobs listed on the scope, and restarting the computer when the system crashes (gratuitously ceases operation).

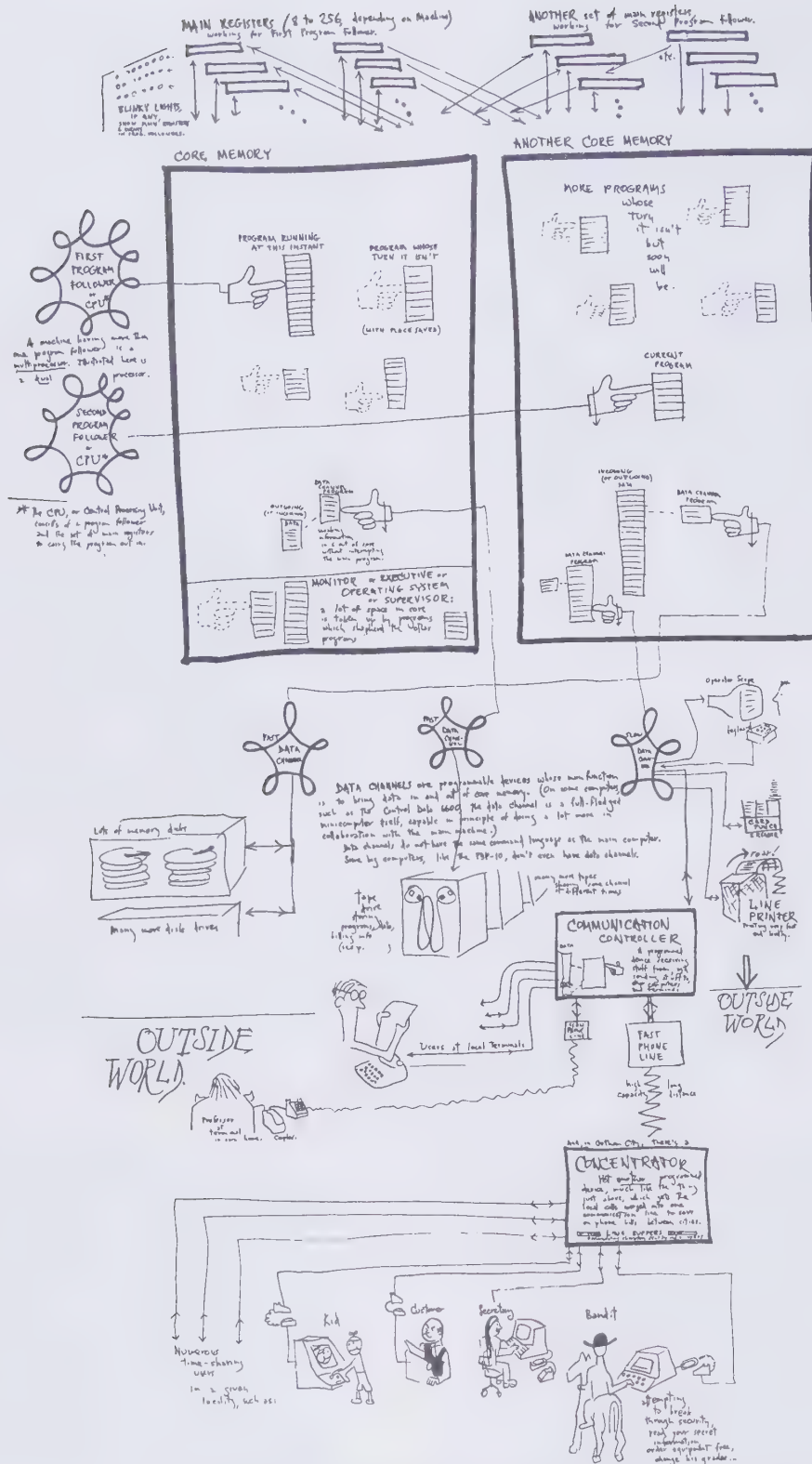
Ms. Woelfer, a thoughtful person, says she does not find her job very stimulating. She *can* program, but the job doesn't involve programming. It's also a lonely job. Non-systems people, except Mayor Daley, aren't ordinarily allowed around. About the only people to talk to are the systems programmers who stop through to look at the scope and see whether their programs are up next.



Operator's console of this particular setup. The operator may use the keyboard or light pen (see p. DM 88 ) to select among waiting programs, submitted by various programmers and departments.







TO BE MORE SPECIFIC, descriptions of some prominent big computers will be found on pp. 85-96.

There's nothing really computerish about this, it's merely sensible construction; but it is traditional in other fields to build something as a tangle of wires. (When TV makers follow these rational practices, they call it "space age construction.")

Why are the different parts so far apart? So there's room to swing them open, refill or change them, sit down and repair them. Refrigerators could, and perhaps should, also be built in separate sections, but it's not traditional. Automobiles can't be spread out because they have to endure the jostles of the road. But computers like this baby aren't going anywhere.

Also intimidating is the fact that you have to *step up* as you enter a computer room. That's because computer rooms ordinarily have raised floors, permitting cables to be run around among the pieces of equipment without your tripping.

Computer rooms are generally lit by millions of fluorescent bulbs, making them garishly bright. This is simply tradition.

Big computers can have millions of words of core memory. Moreover, there are usually several disk drives and tape drives, as seen in the pictures, used to hold data and programs. (Some of the programs are the system programs, especially the language processors and the operating system — see p. 99—but other programs and most of the data belong to the users.)

It used to be traditional for machines like this to have many, many rows of blinking lights, showing what was in all the main registers at any fraction of a second. But there's really no point in seeing all that, since about all you can tell from it is whether the computer is going or not (if it's not, the lights are stopped) and other high-level impressions. For that reason some big computers, beginning with the CDC 6600, started doing away with the fancy lights and bringing written messages to the operator on a CRT scope instead (for lots more on the glories of CRTs, see the flip side, pp. DM 85-93).

Big computers can have multiple program followers and sets of registers (a program follower and its main registers are together called a CPU, Central Processing Unit). A computer with two CPUs, i.e., two sets of program followers and registers to carry the programs out, is

called a dual processor; a computer with more than two CPUs is called a multiprocessor.

Separate independent sections of core memory may be put in one computer, allowing separate program followers and data channels to work at the same time. (Note: a "bank" or core memory is an independent section. Except in this sense of "core memory bank" or "core bank," there is no other correct usage of the layman's vague term "memory bank." Computer people only say "memories," and distinguish further among core, disk, tape, etc. Note that "data banks" are a separate issue—see "Issues," p. 165.)

## DINOSAURS?

Many computer people, the author included, entertain certain doubts about the long-term usefulness of big computers, since minicomputers are cheaper, especially in the long run, and can actually be in the offices and homes where people create and use the information. Big computers are necessary for time-sharing (see p. 103) and huge "number-crunching" jobs (see "Grosch's Law," below). However, it will soon be cheaper to put standardized number-crunching jobs in stand-alone or accessory hardware; see "Microprocessors," p. 96.

Fans of big computers also argue that they are necessary for business programming, but that only means traditional business programming—non-interactive and batch-oriented. For tomorrow's friendly and clear business systems, networks of minis may be preferable. But makers of big computers may be unwilling to admit this possibility.

## GROSCH'S LAW

Minicomputers are so nifty that we may ask why have big computers at all. The answer is that there are considerable economies, especially in applications that require many repetitive operations and don't need interaction with users.

A hypothesis about the economy of big computers was formulated a long time ago by Herbert J.R. Grosch, onetime director of IBM's Watson Lab and now a heavy detractor of IBM. Thus it is called Grosch's Law. The idea is basically that there is a square-law relationship between a machine's size and its power (narrowly defined in terms of the cost of

millions of operations, and *without* considering the advantages of interactive systems or other features which may be of more ultimate value). Anyway, when I asked him recently for *his* formulation of Grosch's Law, I got the following:

"Grosch's Law (formal): Economy in computing is as the square root of the speed.

(informal): If you want to do it ten times as cheap, you have to do it a hundred times as fast.

(interpretive): No matter how clever the hardware boys are, the software boys piss it away!"

## A GROSCH IRONY

Herb Grosch, now editorial director of Computerworld, is perhaps IBM's worst enemy. Once he worked for old man Watson, and was the only IBM employee allowed to have a beard. Now, among other things, he gives speeches and testimony wherever possible about the Menace of IBM, at conferences, at governmental hearings, and in letters to editors.

Yet IBM's main computer sales strategy today is to stress the advantages of big computers with lots of core memory (and persuade you you don't want highly interactive systems or independent minicomputers).

And the fundamental rule stating the advantages of big computers is called Grosch's Law, formulated years ago by none other. See nearby.

## WHERE ARE THEY NOW?

I think I last saw Herb on a plane back from Australia with his family, or maybe it was at a SIGGRAPH where we discussed the funspots of Amsterdam, or maybe he had just joined Mensa. Anyhow, he gets around.



## PERLIS' COROLLARY

"Each breakthrough in hardware technology leads to more massive programming enterprises, new organizational principles, and an enrichment of abstract models."

Alan J. Perlis, foreword to Harold Abelson and Gerald Jay Sussman, Structure and Interpretation of Computer Programs. MIT Press/McGraw-Hill

Dinkies and Mainframes are just ends of a continuum in which there are few sharp distinctions.

Little machines have one program follower, big machines may have many; little machines may move their data in small chunks (like 4 or 8 bits at a time), big machines may move it 32 or 64 bits at a time.

Big machines may be faster or have more instructions. But in principle they're the same—more or less.

A mainframe is essentially just a lot of program followers and a lot of memory.

## LONG SHADOWS

We can boil down the main computer designs to only a few; and they have influenced each other in fascinating patterns.

Now let's consider a few of those specific designs.

Computers have three aspects, as far as I'm concerned:

INTERACTION,  
STORAGE,  
CRUNCH.

INTERACTION is how you get along with the computer, what you and your program can say to each other as you both proceed, the opulent swoop and zoom of tomorrow's flight-simulator workstations and playstations (see p. DM 23).

STORAGE is all the information that's saved so that the computer—and thus the program—can get at it. More and more storage is cheaper and cheaper; and there will be public-access storage systems as well (see p. 141).

CRUNCH is the raw processing power of the machine. On dinkies, like the Apple II and the PC, crunch is measured in cycle time, or how many millionths of a second it takes a byte to move.

The most-often-heard measurement is MIPS, or Millions of Instructions Per Second. Unfortunately the "instructions" of different computers are hardly comparable; one VAX instruction may be comparable to a thousand instructions on a particular RISC. (Thus one Stanford wag has redefined MIPS as Meaningless Interpretation of Processor Speed.)

On bigger machines, like VAXen and IBM mainframes, performance is often measured in Whetstones (and more recently Drystones), which are packages of mixed programs designed to give a measure of overall "average" performance.

On Crays and their little brethren, though, the measure of performance is the megaflop, or one million floating point operations. A "megaflop machine" is a big fast one, like the Cray, that can deliver at least a megaflop per second.

## SOME GREAT COMPUTERS

Here, then, are some thumbnail descriptions of some great, classic or popular computers, expanding our basic diagrams as needed.

Individual computers represent variations of the patterns shown so far.

The particular structure of registers, memories and pathways among them is called the *architecture* of a computer (see p. 76). The binary instructions available to the programmer are called the *instruction-set* of the particular computer (see p. 66). (The word "architecture" is often used to cover both, including the instruction-set as well.)

The principal variations among computers are the word length (in bits—see "Binary Patterns," p. 65) and the number and arrangement of main registers. Then come the details of the instruction-set, especially the ways in which items are selected from core memory—the addressing structure. Then the instruction-set, whose complications and subtleties can be considerable indeed.

The individual computer is the complex result of all of these. If they fit together well, it is a good design. If they fit together poorly, it is a bad design. A bad design is usually not so much a matter of overt stinky features as of ramifications which fit together disappointingly. (*Glitch* is a term often used for such stinky features or relationships.)

The possible ways of organizing computing hardware are vast, and only partly explored.

The machines mentioned here are an arbitrary selection. Some of them are the Great Numbers, computers so important that folks use their numbers as proper nouns, with no brand name:

"Do you have a 360 up there?"

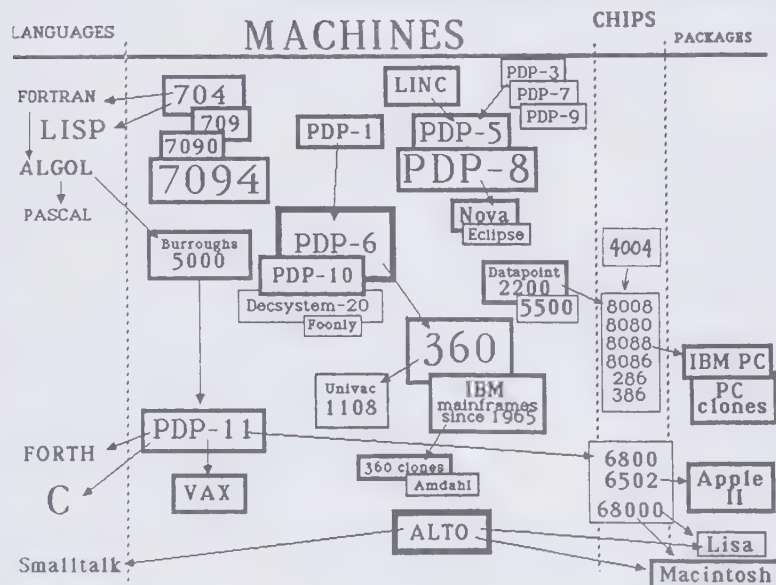
"No, but there's a 6600, a 10 and a bunch of 8s."

"Personally, I'd rather work on a 5500."

Here is what they are talking about.

Ironically, you DON'T HAVE TO KNOW THIS STUFF ANYMORE.

WHAT INFLUENCED  
WHAT?



The action is all in languages. As languages become the center, the structure of the machine becomes less and less important.

The messy complications you do have to know are in the operating systems.

**BIBLIOGRAPHY**

Byte magazine, which tells you more than a sane person could ever want to know about new hardware architecture.

Tracy Kidder, The Soul of a New Machine. Avon, 1985.

Tells of the exhaustion and agony that goes into designing a computer—even

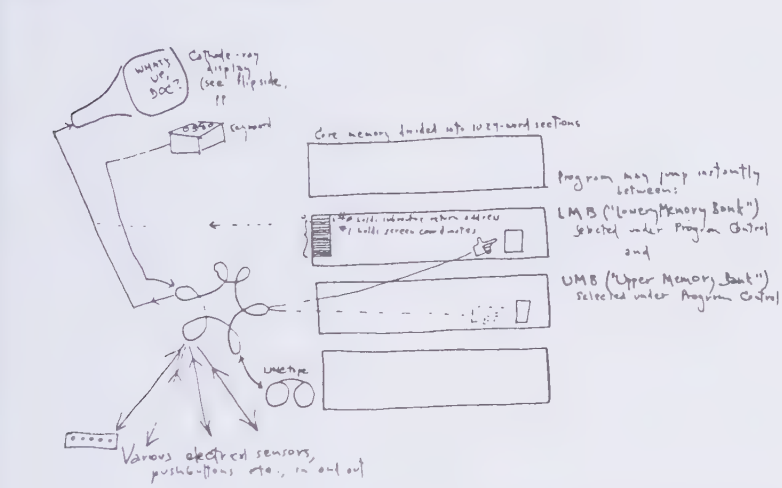
a fairly conventional and uninteresting one.

The classic book: C. Gordon Bell and Allen Newell, Computer Structures: Readings and Examples. McGraw-Hill, 1971.

Note that Bell designed various of the PDPs, and Newell pioneered in list processing (see p. 61).

B. Beizer, The Architecture and Engineering of Digital Computer Complexes. Plenum Press, 2 vols., \$40. Heavier than Bell and Newell. A catalog of thousands of structures and tricks, emphasizing the tradeoffs among them.

**THE CLASSIC LINC (12 bits)**



A computer named the LINC, now usually referred to as "the classic LINC," was perhaps the first minicomputer. It was an important forerunner of our highly interactive systems of today, notably including today's graphic displays with double program followers (see p. DM 88), which offer the highest interactive capabilities.

Perhaps most importantly, it was designed with none of the biases that creep in from the traditions of business computing.

It was called the LINC because it was designed at Lincoln Laboratories (about 1960) for "biomedical research"—actually it was the sort of computer you'd want for hooking up to all sorts of inputs and outputs, to make music, to run your dark-room, but only medical scientists could afford it, so that's what they said it was for.

The LINC had two interesting innovations. It was probably the first computer to be designed with a built-in CRT display (see flip side). It also came with a funny little tape drive, designed for reliability and high response, that was supposed to perform almost as conveniently as a disk and be reliable even in dusty or messy environments. This was the LINCtape, still offered as an accessory by one company. DEC adapted it somewhat and made it the DECTape, handy pocket tape unit of the PDP computer line.

It was never sold commercially. A dozen or so were made up specially out of DEC modules and dealt out to various scientists, and the general hope was that DEC would take the machine up as part of its product line, but that's not what happened.

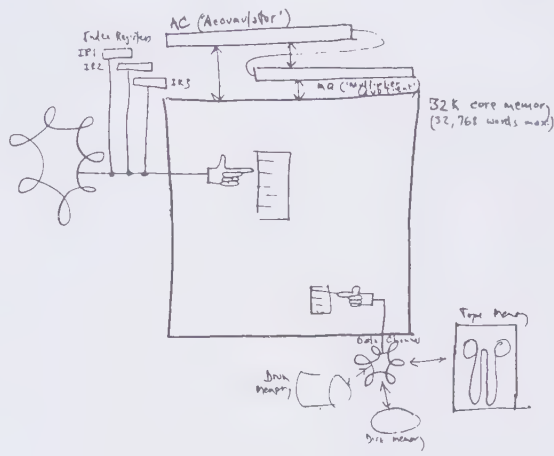
**THE 90 & 94 (36 bits)**

The IBM 7090 was the classic computer. Introduced about 1960 and mostly gone by '66, it was simple and powerful, with clean and decent instructions. With its daughter, the 7094, it became virtually standard at universities, research institutions and scientific establishments. At many installations that went on to 360s they long for those clearminded days.

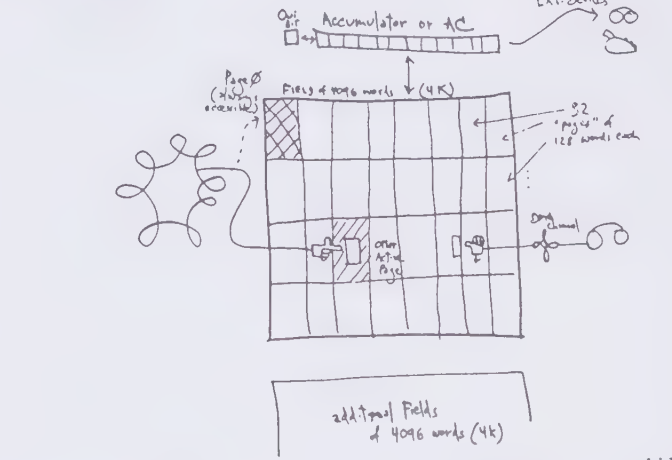
The 90 had three index registers and fifteen bits to specify core addresses. (This meant, of course, that core memory

could ordinarily be no longer than 32,768 words ("32K"—see "Binary Patterns," p. 65.) A later model, the 94, went up to 7 index registers, since there were three bits to select them with.

Though these were million-dollar machines ten years ago, you now hear of them being offered free to anyone who'll cart them away; partly because they needed a lot of power, air conditioning and so on. But they were great number crunchers.



**THE 8 (12 bits)**



The PDP-8 was designed by Gordon Bell (in its original version, the PDP-5) about 1960. Originally it cost about \$25,000; as of May 1974 that price is down to about \$3000, or less than a thousand dollars if you want to buy the circuits and wire it all up yourself. Yup, here comes that Heathkit.

The PDP-8 has been DEC's hottest seller; you'll find them in industrial plants

and museums, or even hidden in the weirdest equipment, from typesetting devices to big disk drives. At universities all over there are kids who know them inside out.

Today the PDP-8 seems archaic, with its one accumulator and awkward addressing schemes: you can only get to 256 different addresses in core memory directly, and it's chopped up into pages. But for its time it was a brilliant design,



packed like a parachute, and even today there are people who swear by it. (But look at what Bell's done lately: the PDP-11, see p. 90.)

*The 8 is pretty much gone now. (Inter-sil put it out on a chip, but it wasn't a hit.) The place of the PDP-8—the useful little machine that kids hack, that works all the time and isn't noticed—has been taken by Apples, PCs, 8080 boxes—the gamut of today's dinkies, which are all its children.*

*And those kids who knew it inside-out now have their own computer companies all over the map.*

*(And in fact Heathkit did offer a PDP-8 interface about a year after Computer Lib first came out.)*

### THE LINC-8 (12 bits, a marriage of the PDP-8 with the LINC.)

DEC was offered the option of building Lincoln Laboratories' classic LINC,

but decided instead to *combine* it, in the mid-sixties, with the already successful PDP-8. That way all the PDP-8 programs and most of the LINC programs would work on it. The result is kind of strange, but very popular in biomedical research: two computers in one, handing control back and forth as needed. You can write programs on the LINC with sections for the 8, and vice versa. Hmm. A more recent and slicker version is called the PDP-12.

The 5000 was designed about 1960 by Edward Glaser and Bob Barton. It was designed to be used *only with higher languages*, not allowing programmers access to the binary instructions themselves. Indeed, it was particularly designed to be used with ALGOL, which would have been the standard language if IBM had allowed it (see p. 60) and is still the "international" language.

## ► Two CLASSICS and Two Classic Problems ◀

Two of the most classic computers illustrate some fundamental aspects of computer architecture. Let's consider IBM's great 7090 from the early sixties, and DEC's great little PDP-8 (late sixties).

## ► Address Space: Integral Vs. Cut up ◀

Gordon Bell has been quoted as saying that in each generation of computer design—mainframes, minis and processor chips—the same mistakes are made over and over. The worst of which is memory segmentation. You can correct anything during the product life of a processor architecture, says Bell, except too few address bits. That makes it necessary to carve memory in pieces, as he did so cleverly in the PDP-8.

The 7090 had an unbroken, single address space in which to put things. (That's because the 7090 had a 36-bit word, with room for fifteen address bits, enough to address 32,768 memory locations—all you could get in those days. You can get about 3 1/2 times that much memory in a PC now.)

The PDP-8, however, only had a 12-bit word, of which only seven bits were free for addressing. That meant dividing memory into segments, or pages, of 128 words (!!!) There was one more bit of the address (not shown on p. 69) that specified whether it was in the "current page" (previously selected) or "page zero" (always available). You could also do an indirect jump in various ways, expanding the total possible memory to 32K, but with further effort. It all had to be done indirectly and by stages.

That is the problem of memory segmentation. The same problem continued on in both big and small machines: the problem is built into both the IBM 360 series (still today's standard mainframe systems) and, at the small end, in the 8088 family, heart of the PC.

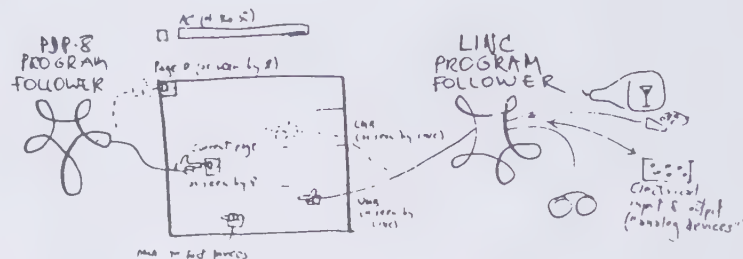
On the other hand, large and clean address spaces are part of the PDP-6 and the Motorola 68000, another classic mainframe and contemporary chip, respectively.

## ● Program Followers: Many Vs. One ●

The IBM 7090 had more than one program follower. The main program follower, following the program itself, concerned itself with the official work of the machine. But information was brought in and out of the computer by so-called "data channels"—minicomputers really, with their own separate program followers, whose work was only concerned with bringing stuff in and out of memory. (Having all those butlers simplified and sped up the major machine a lot.)

The PDP-8 had it down to one wired-in program follower, and all other functions—including input and output—had to be done by that program follower as well.

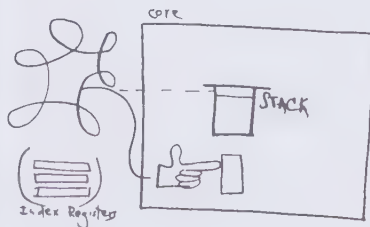
This highlights a fundamental variety among computers. You can add additional program followers to add more power, and it never ends: Ivan Sutherland called it "the great wheel of reincarnation." As you add more function to your program followers, they become computers too, like the data channels of the 7090-4.



While you might half-think that both sides of the computer could work simultaneously, giving you double speed, it doesn't work that way. There's only one core memory, and that sets the basic speed; either a PDP-8 instruction or a LINC instruction can be underway at once, but not both.

Nevertheless, we see here the double structure that plays such an important part in highly interactive computer displays (see p. DM 88). Indeed, LINC programmers often use the machine just that way: the PDP-8 running an actual program, the LINC part running the CRT display in conjunction with it.

### THE VERY GREAT 5000 & 5500 (bit length obscure)



I have heard no computer more widely praised among computer people than the Burroughs 5000 (replaced by the 5500).

Because of this approach, its main registers were to be hidden from the programmer, and attention centered instead upon the stack, a high-level programming device (see nearby about Stacks). However, index registers were added to make it better for Fortran.

The 5000 was marketed as an "all-purpose" computer with an operating system, anticipating IBM's 360 of a few years later. Indeed, after the 360 was announced, Burroughs sales picked up, because IBM salesmen were at last promoting the concepts that customers hadn't understood when they heard about them from Burroughs salesmen years before.

Bigger machines in the line are now the 6500, 6700...

The Burroughs Corporation continues to be an acknowledged leader in computer design. Apparently their sales force is something else, unfortunately. I once spent some time with a Burroughs salesman who not only knew nothing about the magnificent structure of the machine he represented, but would not get me further information unless I demonstrated that the company I represented (a large corporation) was *seriously* interested. He wore very fancy clothes.

EVERYTHING IS DEEPLY  
INTERTWINGLED.

*The 5000 came out in the early sixties, and was too different. The notion of a computer set up for the programmer's*

convenience was too radical, so was the notion of an operating system.

The 5000 was built around the idea of programmer convenience, an unheard idea at that time.

Now at last the programmer's convenience is recognized as one of the most important aspects of a computer system. However, a lot of programmer convenience has migrated into the operating system, at least if it's UNIX.

## THE FOONLY!

ALSO THE DECSYSTEM-20,  
FORMERLY

THE 10, FORMERLY  
THE 6 (36 bits)

DEC's PDP-10 is in some ways the standard scientific computer that the IBM 7094 was in the sixties.

The PDP-10 is excellent for making highly interactive systems, since it can respond to every input character typed by the user.

It is a favorite big computer among research people and the well-informed. The ARPANET, which connects big computers at some of the hottest research establishments, is largely built with PDP-10s. There are PDP-10s at MIT, U. of Utah, Stanford, Yale, Princeton and Engelbart's shop (see p. DM 16). The Watkins Box (see p. DM 104) hooks to a 10.

Digital Equipment Corporation, aware that its computer trademark "PDP" connotes minicomputers to the uninformed, now wants the 10 to be called DECSYSTEM-10 rather than PDP. We'll see if that catches on.

Who designed it is not entirely clear. I've heard people attribute it variously to the Model Railroad Club at MIT, to Gordon Bell, and one Alan Kotok.

Originally, it was the PDP-6, which appeared about 1964, and was the first computer to be supplied with a time-sharing system, which worked from the beginning, if rockily. Now it's good and solid. DEC's operating system for it (see p. 99) is called TOPS, but BBN sells one called TENEX, also highly regarded. The 10 does time-sharing, real-time programming and batch processing simultaneously, swapping to changeable areas of core memory. (This feature should soon be available, at last, on IBM computers ("VS2-2").)

PDP-10 time-sharing works even if you don't have a disk, using DECTape (DEC's cute little tapes). Of course, without disk it's really hobbling, but this capacity is nevertheless noteworthy.

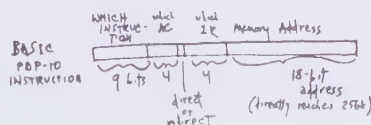
The PDP-10 has debugging commands which work under time-sharing and with all languages, and hugely simplify programming.

Unlike the IBM 360, whose hardware protection comes in options, the 10 has seven levels of protection: the user can specify who may read his files, run them, change them, and do four other things. The PDP-10 does have job control commands, but they are not even comparable in cumbersomeness to IBM's JCL Language (see p. 63), and they are the same for all three modes of operation: time-sharing, real-time and batch.

The PDP-10 has 36 bits but has instructions to operate on chunks, or bytes, of any length. It has sixteen main registers, as does the 360, but uses them more efficiently.

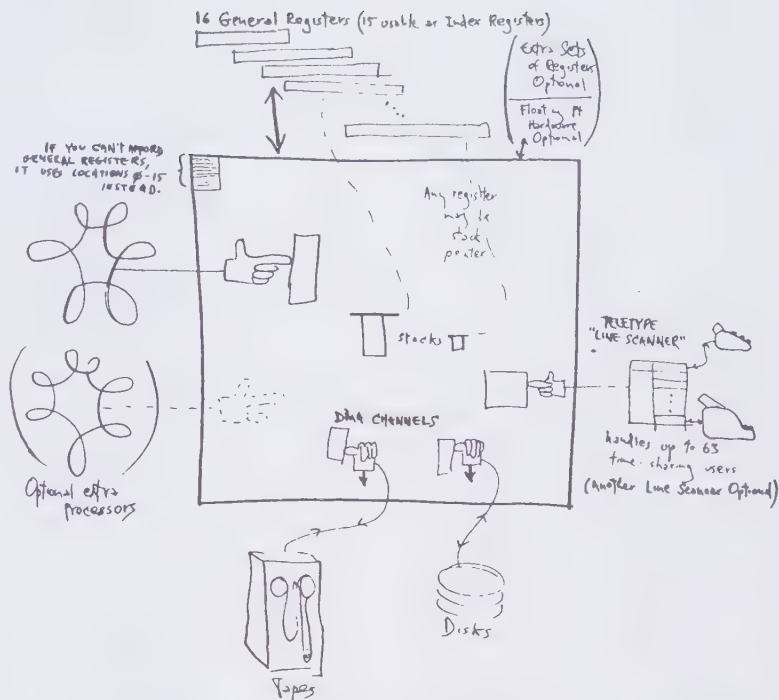
The PDP-10 also has unlimited indirect addressing: an instruction can take its effective address from another location, which can in turn say to take its effective address elsewhere, ad infinitum. For your heavy tight elegant stuff.

Perhaps most important, the 10 has a full set of stack instructions (see "The Magic of the Stack," p. 78), allowing programmers to use multiple stacks for purposes of their own. (The operating system's own stacks are protected.) Programmers do not have to save each other's registers, as on the 360. Programmers are relatively safe from each other.



Some think of the PDP-6 and 10 as a glorified 7094 (with 18 addressing bits, instead of 15). In this case we might consider the 360 a stripped-down version of the 6, since IBM threw out the stack and in most models the memory mapping.

PDP-10s are ordinarily sold where the views of scientists and engineers are considered important, and controllers do



Laphod Beeblebrox, tricky slicker of Douglas Adams' Hitchhiker's Guide teratology, happens to have two heads.

## BEEBLEBROCTICS: Coprocessors

Coprocessors—second processors or program followers—can share a computer's box and memory, effectively making that computer into two machines.

The LINC-8 was a coprocessor of the PDP-8, spliced to a Classic LINC.

Various coprocessors are available throughout the personal computer world, for the Apple II, the PC, the Macintosh, even the Commodore.

EVERYTHING IS  
DEEPLY INTERTWINGLED®



not have first choice. Nevertheless, some say that its business-programming facilities (i.e., COBOL, duh) are just as good as those of companies who claim to have designed computers "for all purposes." First National City Bank of New York has found that the PDP-10 makes a splendid banking computer for internal use, profitable at an internal charge of \$3.75 an hour plus processing charges. Prices for a PDP-10 system with disk start about \$500,000, or \$15 grand a month, and go up into the millions.

However, DEC salesmen are not like IBM's, who can reputedly sell Eskimos to iceboxes. For one thing, DEC salesmen are on salary. That fits DEC's demure, aw-shucks image, but it doesn't exactly sell big computers.

(For you Firesign Theater fans, the mutterings of the dying computer on the "Bozos" album are various PDP-10 system things, artistically juxtaposed.)

Digital Equipment Corporation, aware that its computer trademark "PDP" connotes minicomputers to the uninformed, now wants the 10 to be called DECsystem-10 rather than PDP. We'll see if that catches on.

*It's always sad when people use the wrong word because others misunderstand the right one. But the term PDP-10 is still used by old-timers.*

DEC has been trying to kill the 10 (and the 20) in favor of its VAX line, but the 10 refuses to die. Good computer.

*(Hackers in California hand-built some 10s under the brand name Foonly, out of Mad Magazine, but they went out of business after selling a few.)*

## THE 360 & 370 (32 bits, 16 bits, 8 bits, 4 bits)

More recently the 3300, 4300, Sierra, etc.

"No corporation except IBM could sell a computer like this." — A friend.

The IBM 360 (now called 370 because we're in the 70s) is the commonest and most successful line of computer in the world. This does not necessarily mean it is the best. There are those who appreciate IBM typewriters but not their computers.

360s are bought because the repair service is great; because IBM has very tough salesmen; and possibly for other reasons (see p. 139).

A strange unseen curse seems to haunt the 360 series; indeed, some cynics even

think it results from deliberate policies of IBM! Yet the 360 (and its software) seem somehow organized to make programs inefficient and slow; to make programs big, needing lots of core memory (with numerous enticements for the programmer to take up more); to prevent the compatibilities that are so widely advertised, except through expensive options; to make things excessively complicated, thus locking in both its customers and the employees of its customers to practices and intricacies that are somehow un-

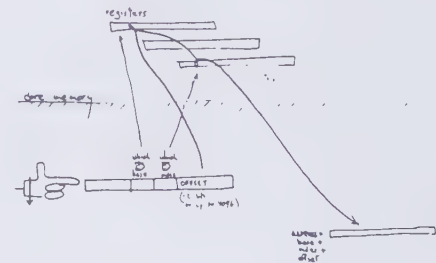
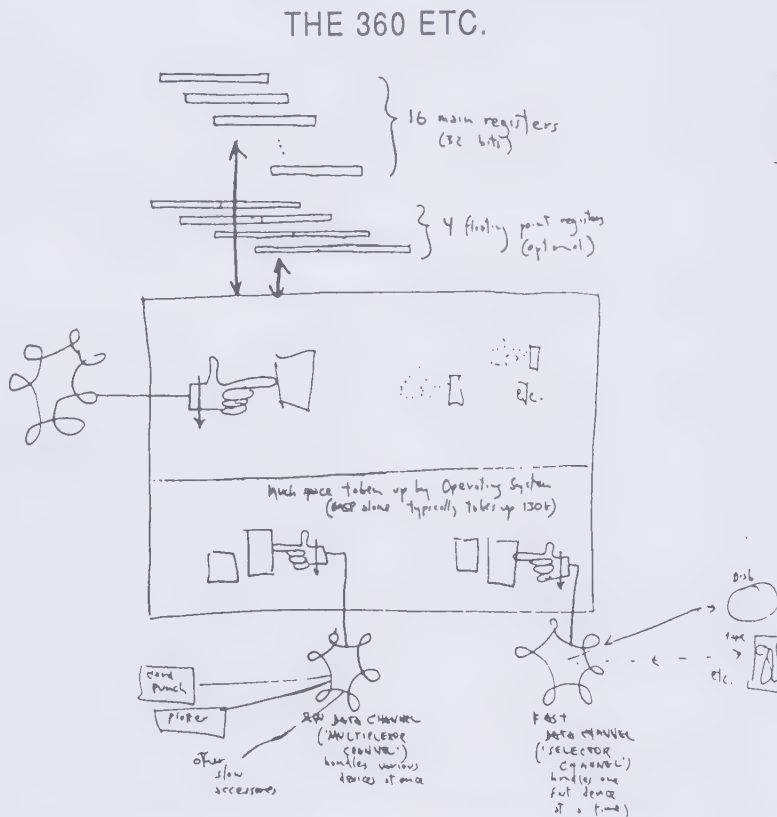
necessary on other brands of computer.

The design of the 360, which was basically decent, is generally attributed to Amdahl, Blaauw and Brooks. Those who hate it, and there are many, base their complaints largely on the restrictions and complications associated with its operating system OS, which is notoriously inefficient (see p. 102).

The architecture of the 360 was quite similar to the PDP-6 (now the PDP-10), designed about the same time: sixteen main general-purpose registers of over

thirty bits, and using the 16 main registers as either accumulators or index registers.

A curious form of addressing was adopted, called "base-register addressing." This had certain advantages for the operating system that was planned, and was thought to be sufficiently powerful that you wouldn't need Indirect Addressing. Two main registers were required, one holding a "base" more or less equal to the program's starting address, and an "index register," whose contents are added to the base to specify an address.



Often a third number, or "offset," is added as well.

The idea of this technique is that programs can be "relocatable," operating anywhere in core memory. A few instructions at the beginning of each program can ascertain where it is running from, and establish the Base accordingly.

The basic idea of the 360 seems to have been doped out for multiprogramming, or the simultaneous running of several programs in core, a feature IBM has pushed heavily with this computer.

## WHAT'S WRONG WITH THE 360?

The main differences between the 360 and the PDP-6 and 10 represent conscious and legitimate and arguable design decisions. To fans of the PDP-6 and 10, here are the 360's main drawbacks:

**NO INDIRECT ADDRESSING.** This was because, within the addressing scheme adopted, indirect addresses could not be adjusted automatically. (But it also makes programs more inefficient, thus more profitable to IBM.)

**NO STACK.** Why? Too expensive, said Amdahl, Blaauw and Brooks in the *IBM Systems Journal*. Funny, they have stacks on \$5000 PDP-11s—and it would have saved everybody a lot of money on programming.

*"Burroughs made a business failure out of a technical success, and IBM made a business success out of a technical failure."*

— Vic Poor

NO MEMORY MAPPING (except on certain models). Where the PDP-6's successor, the PDP-10, automatically takes care of redistributing addresses in core to service every program as if it were operating from location zero on up, the 360 left this general problem to local programmers and (on certain levels) to operating systems.

Handling this automatically in the PDP-10's hardware obviates the complications of base-index addressing and makes possible the efficiencies of indirect addressing.

## LOOKALIKES

360 lookalikes were sold by RCA and Univac. Now that RCA no longer makes computers, Univac is servicing the ones they made.

And Amdahl, no longer with IBM and now head of the Amdahl Corp., is coming down the pike with a super-360 of his own, in part backed by Japanese money. It will be bigger than IBM's biggest—and cheaper.

## THE AMDAHL COMPUTER

The Amdahl computer or System 470, a supercomputer of the 360 series by one of the guys who designed them originally—see p. 88—is now available from Amdahl Corporation, 1250 East Arques Avenue, Sunnyvale CA 94086. (They are now advertising for systems people who know the insides of OS/MVT, VS, etc.) The first 470 is up and running at NASA's Institute for Space Studies, Columbia U. But IBM is said to be readying one of their famous "knockout" machines to do it in. (*Datamation*, July '75, p. 96.)

## THE NOVA (16 bits)

The Nova came out in the late sixties. Basically the story was this: some of the

higher people at DEC, perhaps dissatisfied with DEC's soft sell, perhaps out for their own personal share of things, broke out and started their own corporation. They had in hand the design for a hot, solid minicomputer—some say it was the rejected design for the as-yet-non-existent PDP-11—and since then they have built it reliable and sold it hard.

The basic design of the Nova is sleek and simple: four main registers, no stack, well-designed instructions. Moreover, it was (I think) the first computer to be built around a Grand Bus (see nearby), a design which has caught on rather widely.

Data General (the company mentioned) has used a very interesting marketing strategy. Instead of bringing out a variety of new computers as time goes on, they concentrate on making the Nova faster and smaller. They began by competing against DEC—especially in "the OEM market," purchasers who are buying minicomputers in larger equipment they in turn make—but more recently they have actually started to market against IBM with business systems.

Data General ads have ridiculed the complexity and mystery of IBM systems, arguing quite rightly that minicomputers programmed in BASIC are a reasonable alternative for a wide variety of business applications.

The Nova's instruction-set is clean and straightforward. Key examples (first bits only):

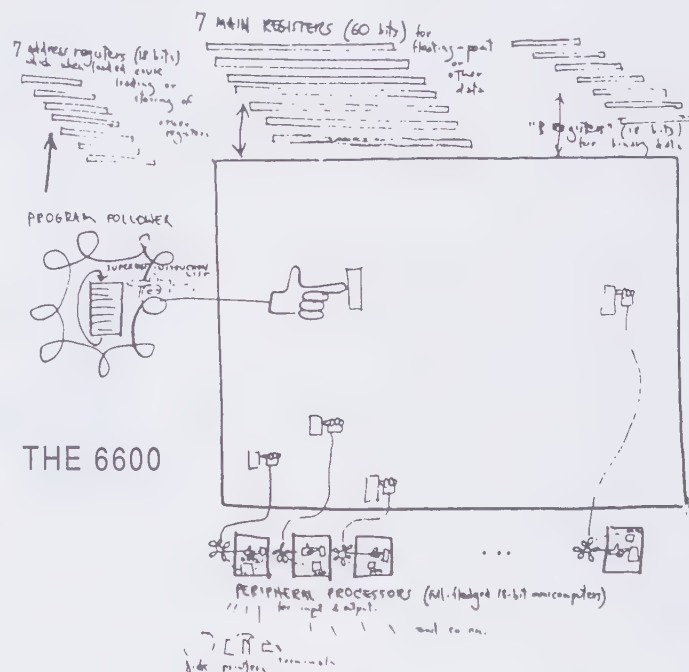
00000	Jump (thus an all-zero instruction jumps to loc 0)
0000X	Subroutine jump
000X0	Increment, skip if zero
000XX	Decrement, skip if zero
00X	Load AC
0X0	Store AC
X	Instructions among registers.

One competitor, Digital Computer Controls, sells a Nova lookalike. Whether Data General will sell you its programs to run on it is another question.

## THE 6600: FIRST OF THE SUPERCOMPUTERS (60 bits)—also 6400, 6800.

Control Data's 6600 computer was the first really big computer. The first one was delivered around 1965. The machine and its operating system, CHIPPEWA, were created by Seymour Cray and his team in hinterland Minnesota.

Extreme speed was designed into the computer in a number of ways. The main computer has no input or output at all; this is handled by data channels which have been built up into full-scale minicomputers or "peripheral processors" of 18 bits.



Instructions can be executed at lightning speed, much faster than the usual microsecond or so. However, since core memory is much slower than the main registers, a trick is used: program instructions are drawn from core into a superfast instruction list (often called a *cache*), and any jumps or loops within this seven-word cache can be executed at unthinkable speeds, perhaps tens of millions of times per second.

The machine is especially geared for floating-point numbers (see p. 71). Because of the intense speed of the fast instruction cache, many instructions (such

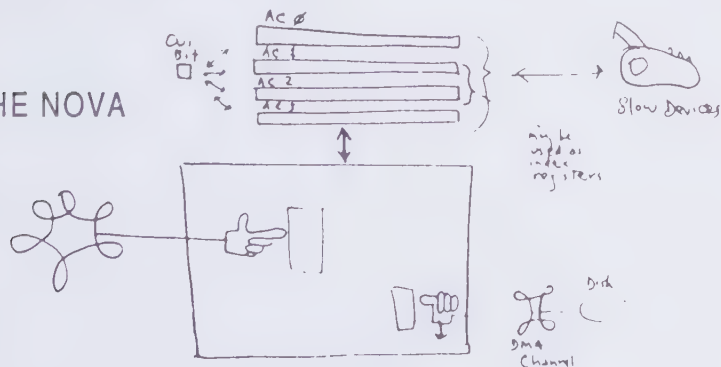
as multiplication and division of integers) can be accomplished faster by a *short program* than if they had actually been wired into the computer.

The 6600 became the start of a whole line, including the 6400, 6800 and others. The 6400 is used by PLATO (see p. DM 93).

## THE CRAY COMPUTER

Seymour Cray, master computer builder, created the 6600 system for Control Data. Indeed, he had the audacity to require CDC to build the computer factory on the property adjoining his own estate in Chippewa Falls, Minnesota. Now that he's broken off to start his own company (with money from CDC, among others), the new computer factory adjoins his estate on the *other* side. The Cray-1,

## THE NOVA



another supercomputer, is nearing completion there.

## Thundercrunch: THE CRAY-1

As we go to press, the Cray-1 is still the preeminent supercomputer. Although a larger one has appeared from Japan, the same Cray architecture is being cranked up to larger systems and should be with us to the end of the century.

It's a "scientific vector processor," that is, its principal use is for trains of numbers (vectors) which are in floating point notation ("scientific"—very big or very



little, but concerned with controllable accuracy rather than a specific number of binary points).

The Cray does not work like a standard computer. It's a streaming processor with incredible throughput of number streams. Its principal processing unit is set up to do arithmetic on streams of 64 numbers at a time shooting through the vector processor. There are eight vector registers (each holding 64 numbers), and those not involved in the current operation can be shooting in and out of memory.

The control unit (or "scalar processor") spends most of its time setting up the operations of the vector processor and the peripheral processors.

This beast is so fierce it doesn't even have a "divide" instruction (dividing takes too long). Instead, it uses something called a *reciprocal-approximate* instruction followed by a multiply; if that's not accurate enough it can repeat.

Sounds like a bear to program, right? Fortunately it's mostly programmed in Cray FORTRAN, which is set up to utilize all this power efficiently.

The Cray-1 has two and a half million electronic parts, but the different types of parts are only seven in number. This decision makes things vastly simpler; in the parts department there are *bins with scoops*, a unique approach to procurement and backup.

You don't see many Crays; indeed, it was peculiar to see the Cray-1 publicized as the effects machine for "The Last Starfighter." Its conventional customers are more or less limited to places like Los Alamos, Jet Propulsion Laboratory, Boeing and the National Security Agency. Of late, though, places like Digital Productions and Lucasfilms have begun to use machines of this caliber for image synthesis and movies (see p. DM 101).

## THE CRAY-2

The Cray-2 is cylindrical (actually shaped like its initial), 45 inches high and 53 inches in diameter. It is awash in fluorocarbon coolant, and you can see bubbles through a window. (It is sometimes called The Bubble Machine.)

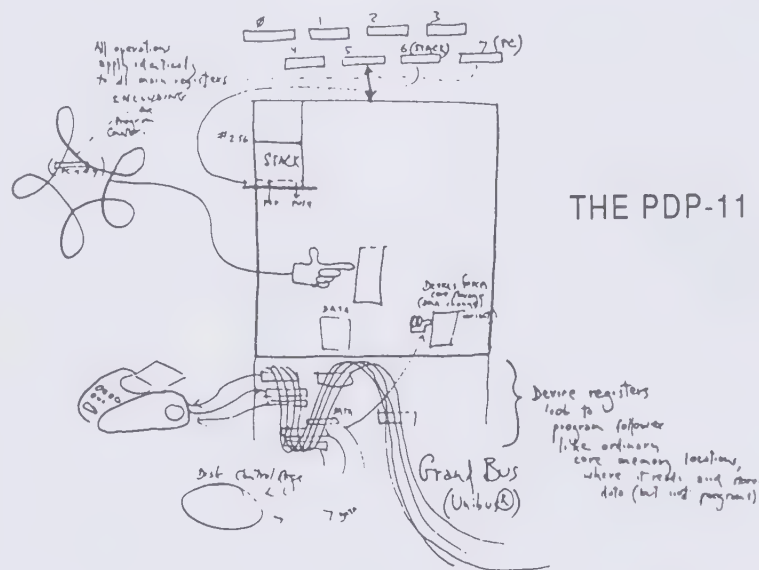
The potential danger of a Cray Meltdown is nontrivial.

Its system clock is 4.1 nanoseconds.

It costs over fifteen million dollars, and you can't buy it if you live behind the Iron Curtain.

It comes with FORTRAN, C and UNIX.

Even if you don't make it your personal computer, you *can* get FORTRAN, C and UNIX.



## THE 11 (16 bits)

The PDP-11 is not a beginner's computer. But the power and elegance of its architecture have established it, since its introduction in 1970, as perhaps the foremost small computer in the world.

Actually, though, we can't be too sure about the word "small." Because as successive parts of the line are unveiled, it becomes increasingly clear that this line of "small" computers has been designed to include some very powerful machines and coupling techniques among them; and it would seem that we haven't seen everything yet.

In other words, DEC's PDP-11, which has already cut into sales of their PDP-8 12-bit series and PDP-15 18-bit series, may soon cut into its PDP-10 36-bit series—as designer Bell unveils (perhaps)

There are eight main registers. Two, though, function specially: the program counter (that part of the program follower that holds the number of the next instruction), and the hardware stack pointer, both follow the same programming rules as the main registers—an unusual technique. Thus a jump in the program is simply a "move" instruction, in which the next program address is "moved" into main register 7, the program counter.

In addition, all external devices seem to the program to be stored in core memory. That is, the interface registers of accessories have "addresses" numerically similar to core locations—so the program just "moves" data, with MOVE instructions, to doorways in core. (This is facilitated by the automatic handling of previously bothersome stuff, like Ready, Wait and Done bits.)

Physically all devices are simply attached to a great sash of wires called a Unibus. (See Grand Bus, p. 98.)

## NEWS OF THE 11

The PDP-11 has now become the first computer to range in size, genuinely, from the tiny to the grand.

## SMALL

The small 11 is a special chip on a small board, called the LSI-11, with a different bus. PDP-11s get moderately large, but there's that 64K address limitation—making it small in principle—which means you have to map memory segments and all that.

## BIG: THE VAX

After the PDP-11, what could Digital do for an encore? The VAX, that's what. It is a very different machine from the PDP-11, although the VAX will run 11 code. DEC calls the VAX a PDP-11, but it isn't, although it's descended from it. Where the 11 had been stark and simple, exactly what the programmer needed (reduced to a clean new set of abstractions), the VAX now added more of what the programmer needed in every direction. Hundreds of new options came in the instruction-set.

The VAX has a squillion different commands for all kinds of stuff you might want to do ("scientific" instructions). The machine had individual instructions to do

monster PDP-11s in arrays or double word-length or whatever.

The PDP-11 was designed by C. Gordon Bell and his associates at Carnegie-Mellon University. In designing the architecture, and especially the instruction set, they simulated a wide variety of possibilities before the final design was decided. The resulting architecture is extremely efficient and powerful (see "The 11's Magic Modes" nearby).

Basically it is a 16-bit machine, with most instructions operating on 8-bit data as well.

## "Little Crays" or (Yaargh) MINISUPERCOMPUTERS

**Crays are multi-million-dollar computers, but a number of firms are now making machines that do the same as a Cray but slower. The minisupercomputers presently start at about a quarter of a million, and deliver good megaflops for the money.**

everything, frozen in the microcode just as if they were wired in.

In a way this was "everything the programmer could want:" hundreds of functions that normally had to be done in software had been absorbed into the basic instruction-set of the computer. The plural of VAX, in hacker slang, is VAXen. (VAXen and Macintoshen are hackers' favorite computers.) VAXen come big (million-dollar class) and small (the chip version is in the ten-thousand-dollar class, a high-end personal machine). You can even get a **portable** VAX, encased by another vendor. (The bad news: it's over twenty grand.)

The VAX has been the general-purpose scientific workhorse now for about a decade. But the handwriting has begun to appear on the wall: while the VAX is a swell machine, it's expensive. Building in all those instructions costs a lot and is hardly necessary for most applications, especially, say, in a UNIX environment, where a lot of it would be available in software anyway.

**MORE! MORE! FASTER! FASTER!**

### THE RISC

Okay, so you want a cheap computer? After a lot of study and simulation, it has

become clear that the price-powerful way to go is to build a computer with not the **most** instructions, like the VAX, but the **fewest**. By making very few instructions, going to the starkest essentials, the machine can be made extremely fast, because the special cases and off-timings are eliminated, as well as the layer of built-in programs, called microcode, that implement all the extra instructions.

Such a machine is called the RISC, Reduced-Instruction-Set Computer. By eliminating the complicated variations and leaving all the extras to software, the RISC appears to be the most bang for the

buck, RISC machines do not invite the compiler-writer to be clever, which is a lot simpler for everybody.

The IBM RT is a RISC machine, but its price-performance has not been impressive. Certain rather special-purpose machines contain RISCs, like RCA's new Digital Video, Interactive system (see p. DM 99). Wheee! 🗑️

### FOUR AGES OF MAINFRAMES

(quotations from Juergen Winkelplatz, Kunstwissenschaftliche Geschichte, tr. Francine Przwiroclaw)

#### THE CLASSICAL MAINFRAME: THE 7094

*Classic means stark, simple, clean, and what the People Yearn For Abstractly.*

#### THE BAROQUE MAINFRAME: (PDP-6 and 10 family, 360 and 370 and so on)

*In the baroque era, the classic structures become embellished and adorned. While the original ideas are remembered and still recognizable, the embellishments shift and enhance their apparent qualities.*

#### THE ROMANTIC MAINFRAME: VAX

*What is the Romantic? It is the system of ideas that reorients and redirects. While the old, which previously seemed eternal, is remembered, it now seems superfluous, superseded, in the onrush of up-gathering newness.*

#### THE MODERN MAINFRAME: RISC

*Comes at last then the Modern. At this culture-time becomes apparent a system of shapes amongst the old more stark, more rarefied and more intrinsic than previously suspected could have been. Out from the confusion of the more recent eras, reaching beyond the classic to a primeval hypothesized simplicity, they arrive, newness condensing oldness into a boldly reduced Reformation.*

## WHAT THE ELEVEN STARTED

Three important innovations began with the PDP-11, I believe, and have become quite widespread.

### 1. SHARED ADDRESS SPACE

Before the Eleven, your computer had a memory space and a completely separate world of input and output devices, which the program would get to in a very different fashion.

The PDP-11 put them all together in a common address space: the same system of addressing included memory and external devices.

This meant that the program could treat outside devices exactly as if they were locations in memory, rather than a specific case. This greatly helped with the modularity and transportability of code.

### 2. GRAND BUS

The PDP-11 did not have a fixed number of places to plug accessories. Instead it had the Unibus, first of the grand parallel busses, which was a system of connectors allowing new devices to be put anywhere along the bus's length. Each external device looks to the program as just a set of locations in memory.

### 3. ADDRESS MODES

The older machines had just a couple of index modes, that is, different ways to divert its operations in memory by varying a given instruction. This let programmers bounce their way through memory in many startling and effective new ways. These instruction modes have been copied in various places, most recently on the 68000 chip.

### 4. NEW LANGUAGES

Interestingly, the PDP-11 inspired two major computer languages, each of which was first programmed on the PDP-11: C, and FORTH.

C borrowed ideas from the 11's addressing structure, especially the ability to point at, and step through, blocks of pointers.

Forth used a specific PDP-11 hardware instruction to end a command and thread back to its main body of code.

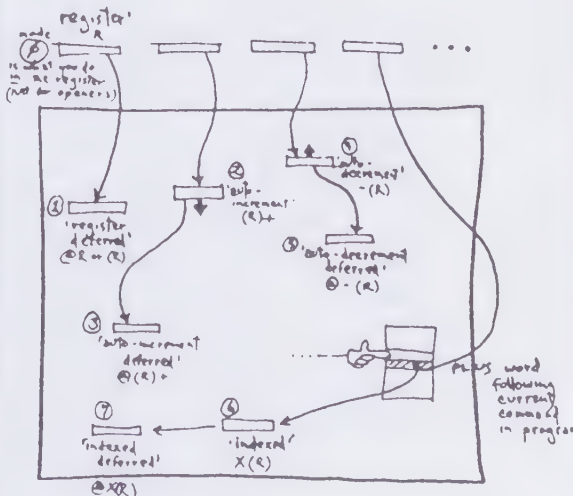
### Bell peels out:

## THE 11'S MAGIC MODES

Minicomputers are cramped, and so the basic problem in mini architecture is how to cram into the instruction enough choices for getting around in core memory.

In designing the PDP-11, Gordon Bell and his co-workers systematically sought a powerful solution, simulating various possible structures by computer program, trying out a variety of different combinations and structures.

The elegance and power of the solution are little short of breathtaking. Basically the PDP-11, the final design, provides seven different types of indirect addressing. The computer's main registers may be used both to operate on information (the usual technique, here called mode zero), or to point to locations to be operated on (indirect modes 1 through 7). These provide extremely efficient means for stepping through tables, PUSH and POP, dispatch tables, and various other programming techniques. The following diagram is meant for handy reference.





## THE ATARI 800 (and 400) and THE AMIGA

Interesting but difficult machines have been created by a brilliant graphics guy and chip designer named Jay Miner.

In the early eighties he created the main chip for the Atari 800 (also sold in a crippled form as the Atari 400)—a small computer (up to 64K) with a lot of arcade-like graphics capability.

The bad news: it was a bear to program, and nearly half the memory had to be given to screen refreshment techniques.

Now Miner has created the chip for another machine with arcade-like graphics capabilities: the Amiga.

The bad news: it, too, is a bear to program.

Nevertheless, in principle it does fabulous things: millions of colors, multiple-bit-plane animation, real-time processing and video (which can be behind or in front of the computer animation), and so on and so on.

The graphics chip also has remarkable scrolling features, based on the structure of video (see pp. DM 84-86); different programs can control different horizontal stripes of the screen.

I've heard that it can only scroll up, but that's hard to believe. Potentially the Amiga could be great—if it catches on.

## WEIRD AND SEXY COMPUTERS

All computers can in principle do the same things, some faster. However, some are too slow or too small ever to do what others can, though the types of their operations are similar.

Some computers (and their languages and facilities) are much more convenient for programmers than others, because their instruction-sets are better.

This is no small matter.

(But it's a big matter of taste and argument among computer people.)

## RETRACTION

Computers are no longer all alike.

The proliferation of new ideas and models and structures is now so great that we can no longer go on saying there's only one basic structure.

However, most computers are alike; and the rest, so far, are special-purpose. Here are some to think about.

## LISP MACHINES—especially, THE GREENBLATT MACHINE

Unsatisfied with the structure of normal computers, they are building at MIT's AI Lab a computer whose native language is LISP. It will have 32 bits with virtual memory, and execute LISP like a bat out of hell.

In a refreshing reversal of trends, it will be for one user at a time. "Time-sharing is an idea whose time has gone," chuckles one participant. (Project MAC, where time-sharing grew up, was there.)

Why a special machine to run LISP? Because conventional machines are so inefficient for it. LISP requires "garbage collection," the sweeping up of old garlands of discarded information, which is helped enormously by special hardware. Furthermore, because LISP scatters and intertwines its data and programs so thoroughly, it runs very badly under conventional virtual-memory operating systems.

There are two LISP-machine companies going, each with a product growing out of the original LISP machine Greenblatt built at MIT. (Greenblatt's first name, Richard, is seldom used.)

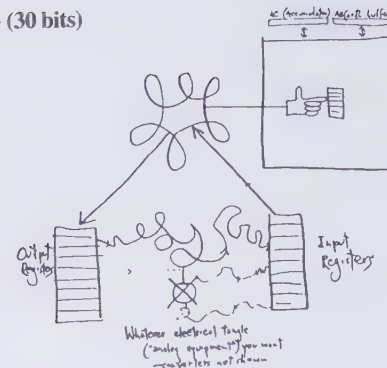
One is that offered by Symbolics, Inc. (and so vigorously campaigned against by hacker-ethicker Richard Stallman, the Chuck Yeager of hackers, as outlined in Steven Levy's *Hackers*). It's an excellent machine, with superb interactive graphics, but as it comes in at over a hundred thousand dollars you can see why others would spring up.

Greenblatt, the Hacker's Hacker—the Lindbergh of Hackers (again, see the Levy book), now builds LISP machines at LISP Machines Inc. (Texas Instruments is manufacturing and marketing their processors.) The latest model will have the 68000 as a coprocessor.

One of the special features of the LMI LISP Machines is that you can compile your favorite routines into microcode—underlevel fast instructions—in order greatly to speed up code that depends on them.

Xerox also sells LISP machines, the D series—Dolphin, Dandelion and Do-rado. It's clearly a growth market.

## THE AMBILOG (30 bits)



An interesting but little-known computer was the Ambilog, made by Adage, Inc. of Boston, a most innovative machine first marketed in the mid-sixties.

The Ambilog is a hybrid computer, i.e., both digital and analog. "Analog computers" are any electrical circuits set up to produce a result according to some formula. For certain types of repetitive functions, analog makes a lot of sense. Thus the Adage people put this machine together for highly efficient hybrid computing.

The essential idea was to have a highly ventilated machine that could take in and put out measurable electric signals at high rates. What they created was a rather straightforward digital computer with a lot of registers and converters to send analog information out and bring it back in. This meant that problems suited to repetitive electrical twisting and measurement could gush out through special analog circuits, and the "answers" or doctored signals could gush back in.

The instruction-set was designed for this high-speed management of input and output.

The principal applications this equipment has been used for are three-dimensional display (see Adage Display, p. DM 89) and Fourier analysis for sound and other applications (see p. DM 61).

## THE ESS

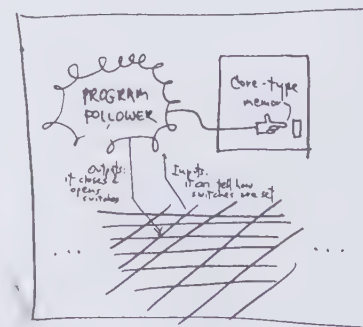
The national phone company (usually called affectionately, "Ma Bell") has drastically changed its switching methods in the last few years. They are replacing the old electromechanical switches, or "crossbars," with a new de-

vice called the ESS, or Electronic Switching System. If there's one in your area you may hear about it in their jolly news sheet that you get with the bill.

In the old crossbar days, a phone connection was a phone connection and that was that. Now, with the ESS, all sorts of new combinations are possible: the ESS has stored programs that determine its operation. If you dialed a non-working number, it jumps to a program to take care of that. It does all sorts of things by special program, and new programs can be created for special purposes. Now the phone company is trying to find the services that people will pay for. Having calls rerouted temporarily to other numbers? Linking up several people in a conference call? Storing your most-called numbers, so you can reach them with a single or double digit?

These particular services are now being offered experimentally.

The way it works is this: there are a number of programs stored in a core memory; the only "output device" of the system consists of its field of reed switches, arranged to close circuits of the telephone network.



Depending on the numbers that have been dialed, and whatnot, the ESS jumps to a specific program, and that tells it to connect an incoming call to particular other circuits, or to ring other lines, or whatever.

It's really neat.

There are only a couple of things to worry about. One is that it makes wire-tapping not a complex bother involving clipped wires and men hunched over in cramped spaces, but a simple program.

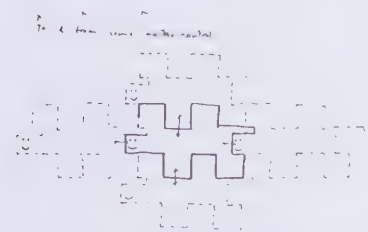
Another is that some people think that blue-boxers (see nearby) may be able to program it, from the comfort of their own homes. Meaning that not just court-authorized wiretaps, but Joe Schmoe wiretaps, would be possible. Let's hope not.

## OPTICAL COMPUTING

Beams of light do things that it's very hard to simulate. This means that if you want certain types of computation done that light beams do naturally or as lasers (such as Fourier transforms), there is a case for building a special machine that actually carries out the process by light. Different guys are working on various stuff.

It is also the case that as computers get bigger, it makes sense to tie their parts together with paths of light rather than wires. This avoids stray electrical needles and interferences and also speeds it up.

## CELLULAR SYSTEMS



Now that integrated circuits are getting cheap, the distinction between registers (where things happen to information) and memory (where nothing happens to information) can be reconsidered. Storing information in cells that can themselves perform actions, or having numerous subsystems in which computation takes place, leads to a fascinating variety of possible architectures. These are generically called "cellular" computers; this is slightly ironic considering that the living cell itself is now known to be at least a digital memory, and probably more (see p. 94).

Examples of cellular computers more or less include the ILLIAC IV. But this type of architecture has barely begun.

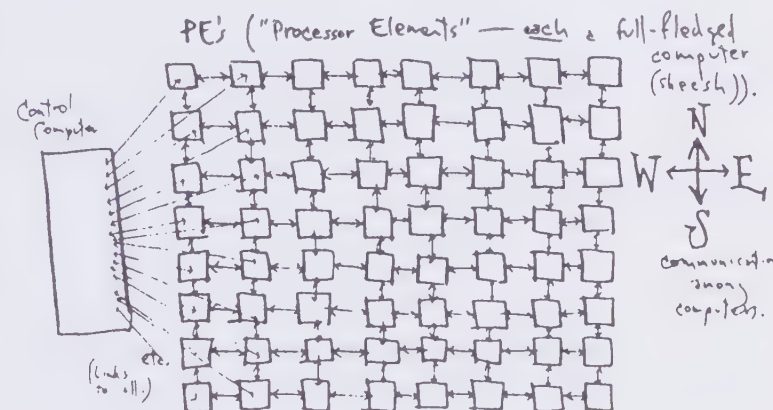
A new and vital cellular chip is the Transputer, permitting great pour-through programming (see p. 99).

Incidentally, the new "cellular telephones" don't have anything to do with cellular computers, they just divide cities up into sections with separate phone transmitters.

## THE ILLIAC 4

The Illiac IV is the biggest and most extraordinary computer in the world, knock wood. To most computer people it's about as big as anything they want to think about.

The Illiac 4 consists of sixty-four big-gish computers, all going at once under the supervision of yet another big com-



puter, typically all working on a single problem. It is the brainchild of Daniel Slotnick, who worked on the theory of array computers and pressed for its creation for years; eventually built by Burroughs, it sits at an airbase but is available to outside users through the ARPA network.

In principle the idea is this: certain classes of problems, especially those involving very large arrays and matrices, can be run only rather slowly on ordinary computers. If, however, a computer is built which itself is an array, certain operations can take place very much faster because they happen in parallel units simultaneously. Matrices, particular formal kinds of array, are used in a great variety of mathematical-type applications. For instance, weather prediction. It seems that the theory of weather prediction has been well worked out for de-

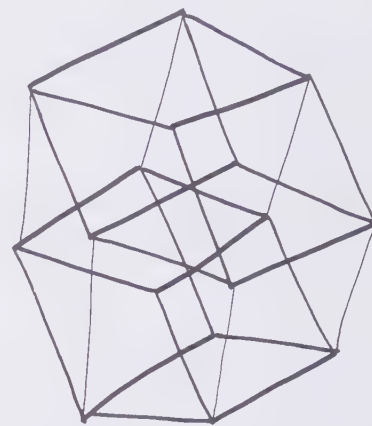
acades, but because the swirly behavior of the atmosphere is so intricate, actually calculating out everything involves billions of operations. At one conference session I believe it was explained that it used to take twenty-five hours to predict the weather twenty-four hours in advance, which means you get the answer an hour after it's happened already; now it is possible, using Illiac IV, to do the whole planet's weather in an hour and a half, said the speaker.

Some say that may be its only use and the whole project was inadequately thought out. Others suspect it's really intended as a radar-watcher for the ABM systems.

Anyway, there it is. And the individual briefcase-sized Burroughs machines, if they're ever marketed, may provide a new price breakthrough for small high power systems.

Sure. "Dimension" simply means yet another direction at right angles to the others. You can have all the dimensions you want. We live and perceive in three dimensions that are homogeneous, meaning you can rotate in them (time, a fourth dimension, does not afford the same movements and rotations except on Einsteinian levels). But conceptually any number of dimensions are possible: wherever you need to branch out in one more direction of anything, you're talking dimensionality.

So the cube that comes after the normal cube is called a hypercube. And its connections look like this:



(Just as a square connects the endpoints of two lines separated by another dimension, and a cube connects the corners of two squares separated by another dimension, a 4D hypercube connects the corners of two cubes separated by another dimension, and so on for any number of dimensions. Hypercubes are extended to additional dimensions by connecting all their corners to the corners of other hypercubes, just like lines and squares.)

So the Hypercube, around for years, is lots of processors whose interconnections would be in the form of a hypercube. Each processor would be able to communicate or share memory with those connected every whichwise. Surely this would solve many problems of the universe, just as soon as, uh, the software could be worked out.

One extraordinary young dreamer, Danny Hillis, has put real punch into the theory. At Thinking Machines, Inc., in Cambridge—a lavishly-funded outfit—Marvin Minsky's former righthand wit, good ol' Danny that we've known and loved all these years, has built the

Incidentally, "Illiacc" is the traditional name for computers built at the University of Illinois. Will the series end with this one?

The Illiac IV was expected to revolutionize computerdom. It didn't. It just slid off to the side.

## BIBLIOGRAPHY

Daniel J. Slotnick, "Unconventional Systems." Proc. SJCC 1967, pp. 477-481.

## HYPERCUBE COMPUTERS

As you may know, a line is the extension of a point in the next dimension. A square is an extension of a line in the next dimension. A cube is the extension of a square in the next dimension. And after that—can it go on in "other dimensions"?



Connection Machine, a multicomputer of immense power. The separate computers aren't very powerful, but **every one is connected to every other**, leading to extraordinary new powers. Redesigning the software world for it is another problem, however. (It programs in LISP.)

The darn thing works. With over **sixty-four thousand** little computers all interconnected, each as powerful as (say) an Apple II, the Connection Machine already is the most powerful computer in the world. And the first **real** departure from classic computer architecture. It looks as if Daniel Hillis ranks with Babbage, Mauchly, Cray, Noyce and Bell as one of the great computer designers.

But just wait. Danny is talking a million dimensions. And now people take him seriously.

## BIBLIOGRAPHY

W. Daniel Hillis, "The Connection Machine." Scientific American, June '87, 108f.

## OCTOPUTSCH

Of course you've thought that the hardwired setups were for sloppy analog types of things. But here now we have THE CHESS MACHINE, under straightfaced construction at the MIT AI Lab, which will provide HARDWIRED THREAT ANALYSIS. Yes, its advanced perceptron architecture will supposedly be capable of analyzing threats to any given position in a GRAND PARALLEL FLASH. The impact of this astonishing development on the world of Electronic Chess, or anything else, for that matter, is totally impossible to predict.

**Speaking of Grand Parallel flashes, consider a super cellular computer, the Pixar, which operates on EVERY BIT OF A PICTURE SIMULTANEOUSLY.**  
(See p. DM 113)

## DATAFLOW MACHINES

are a new kind of architecture, not yet commercial, where data sequences slither through the machine in shift registers, and are gated together for shared operations when the compiled program recognizes their joint readiness.

Dataflow machines might just knock the pants off the Cray. But I'm sure Seymour Cray is on top of the matter.

## TRENCHANT MOORINGS

Perhaps the **most** unconventional piece of gear is based on a highly controversial mathematics called "the calculus of indications," first presented in G. Spencer Brown's strange book The Laws of Form. Its notation consists of a lot of gammas, under one another recursively in odd combinations; and its author asserted that it covered both symbolic logic and dynamic digital systems. Part of its outrageousness was that it derived from nothing else in mathematics, putting beginners on an equal footing with PhD's, and making them all equally uncomfortable.

Anyway, somebody named Trenchard More has now built a bat-out-of-hell machine for the processing of calculus-of-indication models. And where that will lead is a mystery.

## THE MITIEST COMPUTER?

The focus of attention in genetics and organic chemistry has for a decade now been the remarkable systems and structures of the molecules of life, DNA and RNA.

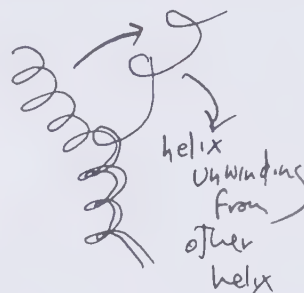
DNA is the basic molecule of life, a long and tiny strand of encoded information. Actually it is a digital memory, a stored representation of codes necessary to sustain, reproduce, and even duplicate the creature around it.

It is literally and exactly a digital memory. Its symbols are not binary but *quarternary*, as each position contains one of four code molecules; however, as it takes three molecules in a row to make up one individual codon, or functioning symbol, the actual number of possible symbols is 64—the number of possible combinations of four different symbols in a row of three. (I don't know the adjective for sixtyfourishness, and it's just as well.)

The basic mechanism of the system was worked out by Francis Crick and James Watson, who understandably got

the Nobel Prize for it. The problem was this; how could living cells transmit their overall plans to the cells they split into?—and how could these plans be carried out by a mechanical process?

The mechanism is astonishingly elegant. Basically there is one long molecule, the DNA molecule, which is really a long tape recording of all the information required to perpetuate the organism and reproduce it. This is a long helix (or corkscrew), as Linus Pauling had guessed years before. The chemical processes permit the helix to be duplicated, to become two stitched-together corkscrews, and then for them to come *apart*, unwinding to go their separate ways to daughter cells.



As a tape recording, the molecule directs the creation of chemicals and other cells by an intricate series of processes, not well understood. Basically, though, the information on the basic DNA tape is transferred to a new tape, an active copy called "messenger RNA," which becomes an actual playback device for the creation of new molecules according to the plan stored on the original.

Some things are known about this process and some aren't, and I may have this wrong, but basically the DNA—and its converted copy, the RNA—contain plans for making all the basic protein molecules of the body, and anything else that can be made with amino acids. (Those molecules of the body which are *not* proteins or built of amino acids are later made in chemical processes brought about by these kinds.)

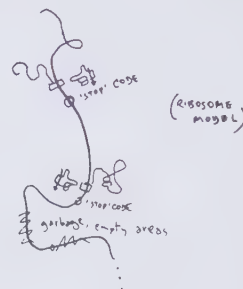
Now well you may ask how this long tape recording makes chemical molecules. The answer, so far as is known, is extremely puzzling.

As already mentioned, the basic code molecules (or nitrogenous *bases*) are arranged in groups of three. When the RNA is turned on, these triples *latch onto* the molecules of amino acid that happen

to be floating by in the soupy interior of the cell. (There are twenty-seven amino acids, and sixty-four possible combinations of three bases; this is fine, because several different codons of three bases can glom onto the same passing amino acid.)

Now, the tape recording is divided into separate sections or *templates*; and each template does its own thing. When a template is filled, the string of amino acids in that section separate, and the long chain that results is a particular molecule of significance in some aspect of the critter's life processes—often a grand long thing that *folds up* in a certain way, exposing only certain active surfaces to the ongoing chemistry of the cell.

One theory about the mechanics of this is that a sort of zipper slide, called the *ribosome*, chugs down the tape, attaching the called-for amino acids and peeling off the ever-longer result.



Now, here are some of the funny things that are known about this. One is that there is a particular codon of three bases that is a *stop code*, just like a period in ordinary punctuation. This signals the end of a template. Another is that the templates on the tape are in *no particular order*, but distributed higgledy-piggledy. (Geneticists engaged in mapping the genes of a particular species of creature find that the gene for eye color may turn out to be right next to the gene for length of tail—but where those are *really*, and what the particular molecules do that determine it, are still mysterious sorts of questions.)

Here is some more weird stuff about this.

Large sections of the DNA strand are "dark," it turns out, just meaningless stretches of random combinations of bases that don't mean anything—or ever get used. This ties in, of course, with the notion that genetic change is random and blind: the general supposition is that genetic mutation takes place a base or two at

a time, and then something else activates a chance combination in a dry stretch that turns out to be useful, and this is somehow perfected through successive 1-base changes during the process of successive mutation and evolution.

Amazing use is made of these mechanisms by some viruses. Now, viruses are often thought of as the most basic form of life, but actually they are usually dependent on some other form and hence more *streamlined* than elemental. Well, some viruses (but not all) have the capacity for *inserting* themselves in the genetic material: breezing up to the DNA or RNA, *unhooking* it in a certain place and lying down there, then being *duplicated as part of the template*, then unhooking themselves and toddling away—both parent virus and copy. I can't for the life of me think of an analogy to this, but I keep visualizing it as happening somehow in a Bugs Bunny cartoon.

## CONTROL MECHANISMS

Now, all cells are not alike. From the first beginning cell of the organism (the zygote), various splits create more and more specialized, differentiated cells. A liver cell is extremely different from a brain cell, but they both date back by successive splitting from that first zygote. Yet they have different structures and manufacture different chemicals.

One simplification may be possible: the "structure" of a cell may really be its chemical composition, since cell walls and other structures are thought to be special knittings of certain tricky molecules. Okay, so that may reduce the question slightly. How then does the cell change from being an Original (undifferentiated, zygotic) cell to the Specialized cells that manufacture particular other complex chemicals?

One hypothesis was that these other cells have different plans in them, different tapes. But this theory was discarded when John Gurdon at Oxford produced a fresh frog zygote from the intestinal cell of a frog (which accordingly, in due time, became a frog *de facto*). This proved, most think, that the whole tape is in every cell. Thus there must be something-or-other that blocks the different templates at different times (You there, now you're a full-fledged epithelial cell, never mind what you did before) and selects among all the subprograms on the tape.

Much pressing research in molecular biology, then, is concerned with searching for whatever it is that switches different things on and off at different times in the careers of the ever-splitting cells of our bodies. Not to mention those of all other living creatures, including turnips.

## COMPUTERISH CONJECTURES

The guys who specialize in this are usually chemists, and presumably know what they're doing, so the following remarks are not intended as butting into chemistry. However, new perspectives often give fresh insight; and the matters we've covered so far might seem to have a certain relevance.

DNA and RNA, as already remarked, may without distortion be thought of as a *tape*. Indeed, on this tape is a *data structure*, and indeed it is a data structure which seems to be involved with the execution of a *program*—the program that occurs as the organism's cells differentiate. There is evidently some sort of *program follower* which is capable of branching to different selections of (or *subprograms* in) the overall program, depending on various factors in the cell's environment—or perhaps its age.

Now, it is one thing to look for the particular chemical mechanisms that handle this. That's fine. On the other hand, we can also consider (from the top down) what sort of a program follower it must be to behave like this. We may consider both its *logical* structure—its mechanisms and effects, considered from a computer-man's point of view—and its chemical structure, or what is really happening. (This is like the difference between tracing out particular circuitry and trying to figure out the structure of a program from how it behaves.)

At any rate, the following interesting conjectures arise:

0. The genetic mechanism really seems to be a list processor (see pp. 72, 61-62), using associative rather than numerical addressing. The gene is now thought to be divided into four segments, called Promoter, Initiator, gene proper, and Terminator. As I understand it, the promoter and terminator zones contain codes which mean, simply, Start and Stop. The initiator zone, however, is a coded segment which effectively *labels*

the gene. This initiator area contains a chemical code unique for every gene.

The genes are turned *off* by grabbing



molecules, or *repressors*, whichglom onto the initiator sections of the genes which they have been *specifically coded to repress*.

1. The mechanism of somatic reproduction is a *subroutining program follower*—not unlike the second program follower of the subroutining display (see p. DM 89). That is, it steps very slowly through a master program somewhere, and with each new step directs the blocking or unblocking of particular stretches of the tape.

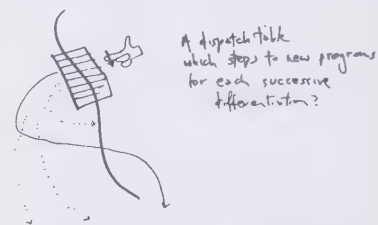
As the program is in each cell, presumably it is being separately followed in each cell. (This is sometimes called *distributed computing*.)

2. In each cell, the master program is directing certain *tests*, whose results may or may not command program *branching*—successive steps to new states of the overall program. It may be testing for particular chemical secretions in its environment; it could even be testing a *counter*.

3. (This is the steep one.) If this were so, we might suppose that this program too was stored on the DNA, in one or more program areas; and it would therefore be necessary to postulate some addressing mechanism by which the program follower can *find* the templates

to open and close. (And perhaps further sections of the program.)

4. Indeed, it makes sense to suppose that such a program has the form of a *dispatch table*—a list of addresses in the tape, perhaps associated with specifications of the tests which are to cause the branching.



These wild speculations are offered in the spirit of interdisciplinary good fellowship and good clean fun. Whether (1) and (2) have any actual content, or are merely paraphrases of what is already known or disproven, I don't know; somebody may find the rest suggestive.

Two more observations, though. These are not particularly deep, and may indeed be obvious, but they suggest an approach.

5. There is definitely a Program Restart: to wit, whatever it is that turns an old differentiated intestine cell into a fresh zygote.

6. Cancer is a runaway subroutine.

Research in this area must now find the specific coding of molecules which block and unblock specific genes, and how these fit in the overall graphs of metabolism, immunology, development, and so on. If there is anything to make an old atheist uneasy, it is the extraordinary

## There are Consoles For Building Genes (Data Sequences) Now

The gene designer sits at a screen and selects codons. The DNA or RNA strand is built on request.

Highly-miniaturized digital memory. But you don't want this on your personal machine.

Alan Kay tells this story from when he worked at a video-game company.

He was trying to explain to an associate the immense memory capacity of DNA and RNA storage. For example, he said, the human digestive bacterium *e. coli* contains the information equivalent of twelve million game cartridges in its DNA.

"How can I get some of them *e. coli*?" the other asked eagerly.

"You can either use a feather or look at yesterday's stock reports," said Alan.





beauty of this clockwork.

From all this, one last speculation creeps forward.

Ivan Sutherland, in considering the structure of subroutining display processors, has noted that as you get more and more sophisticated in the design of a display program follower, you come full circle and make it a full-fledged computer, with branch, test, and arithmetic operations.

If the somatic mechanism should turn out to have a program follower as described, it is not much of a step to suppose that it might have the traits of an actual computer, i.e., the ability to follow programs, branch, and perform manipulations on data bearing on those operations.

In other words, the digital computer may actually have been invented long before von Neumann, and we may have billions of them on our persons already.

It may sound far-fetched, but the mechanisms elucidated at this level are so far-fetched already that this hardly seems ridiculous.

## THE COMPUTER FRONTIER

Regardless of what's actually in the cell, it is clear that being able to adapt molecular chemistry, especially DNA and RNA, to computer storage is a beckoning computer frontier. This would make possible computer memories which are far larger and cheaper than any we now have.

Basically we can separate this into two aspects:

The DNA Readout. This part of the system would create long molecules holding digital information.

The DNA Readin. This would convert it back to electrical form again.

GULP

You frequently hear the rumor that the AIDS virus was constructed by, e.g., the CIA. Probably not, on technical grounds. First, it's been around at least since 1978, when genes could not be built as they can now; secondly its center of infection appears to have started in the middle of Africa; and third, people who have such capabilities are immensely wary of their dangers.

Weird possibilities follow. One is that (if chemical memory is generic, rather than idiosyncratic to an individual's neural pathways) knowledge could be set up somehow in "learned" DNA form, whatever that might turn out to be, and injected or implanted rather than taught. Weird.

As our ability to create clones improves, we could clone new creatures, or genetic "improvements"—which, considering the racehorse and the Pekinese, means "those sorts of nonviable modifications supported in human society." And of course that ghastly stuff about building humans, or semi-humans; having traits that somebody or some organization, ulp, thinks is desirable. . .

But the real zinger is this one. It might just be a small accidental printout meant to test the facility, or maybe just a program bug—

—but the system could output a virus that would destroy mankind.

## BIBLIOGRAPHY

James D. Watson, *Molecular Biology of the Gene*. Beautifully written; meant for high-school science teachers. But potentially formidable; if so, start with his autobiographical *The Double Helix*, which is a gas.

Lewis Thomas, *The Lives of a Cell*. Viking, \$7. Eloquent writing to popularize, among other things, the New Genetic view that your modern animal cells, and mine, actually contain various fungi and other stray ding-a-lings that slid into one of our ancestors and found useful work, joining the basic genetic program.

## MICROBOTS

*Speaking of chipping away at the future, the most radical ideas of the future,*

*bar none, are those propounded by K. Eric Drexler, MIT-area physicist and polymath, and a valuable member of the Xanadu team (see p. DM 141.)*

*Drexler attracted note a few years ago with his analysis (published in Coevolution Quarterly) showing that the best way to get around the solar system, once such matters are up for consideration, will not be rockets but solar sails. Pound for pound, the solar sail will do far more, and not get used up in the process. On the first day with a sail, says Eric, you generally get as much thrust as with the equivalent mass of rocket fuel, but then you get it again the second day. (Drexler has also analyzed the proprietary Xanadu algorithms and pronounced them fit, also suggesting variants of his own.)*

*But Drexler's main thrust is in another direction entirely, as discussed in his book Engines of Creation (Anchor/Doubleday).*

*He posits that we will be able to create microscopic self-reproducing robots, precisely engineered for exact purposes and not liable to mutate and go haywire.*

*These microbots require no particular breakthrough, says Drexler. All they require is the step-by-step continuance of today's miniaturizing technologies, as pressing on in the integrated-circuit field, together with the adaptation of partial genetic mechanisms through gene-splicing to do specific tasks.*

*Drexler sees these self-reproducing microbots as creating perfect cloth and unbreakable rope, bringing immortality, transforming planets, and more.*

*Drexler is brilliant, knowledgeable, and a hardliner physicist, so his theories will be getting a great deal of attention in the coming years. Which of us will live to see such an era is an interesting question. By Drexler's estimate, it's only a few decades away.*

*You may recall the idiocy of the term "microcomputer" as commonly used, which I have pointed out elsewhere in this book. But here is a more reasonable definition of a "microcomputer," from the Drexler perspective:*

*A microcomputer is any computer too small to be seen with the naked eye.*

# IN THE CHIPS

What makes possible the computer counterculture and everything else is, of course, the spectacular development of electronic-chip technology, the techniques of shrinking great electronic circuits to almost no size. Electronic rigs that were shoebox-size ten years ago are typically now etched on chips the size of your thumbnail and sold for a few dollars, no matter what they contain.

## HERE THEY COME—THE MICROPROCESSORS!

Microprocessors are what's happening.

Computers cost several thousand bucks on up. Microprocessors cost several hundred on up, and that price range is falling fast.

Microprocessors are already on *integrated circuits*, postage stamp-sized electronic tangles that are simply printed and baked, rather than wired up; this means there is effectively *no bottom limit* to the price of microprocessors. Mark this well. It means that in a few years there will be a microprocessor in your refrigerator, your typewriter, your lawnmower, your car, and possibly your wallet. (If you don't believe this, look what happened to pocket calculators in the last couple of years. The chip those are built around costs five bucks. But next come the *programmable* chips, the microprocessors.)

Microprocessors should not be called *microcomputers*, a term that seems to have captivated Wall Street lately. "Microcomputer" just means any teeny computer.

Ignore the prices above. Every chip comes out at \$500 or so and comes down to \$5 or so three years later.

Indeed so. And now, in 1987, the price of memory chips is down to about .001 cents per bit. Depends where you buy, though. There are discounters everywhere.

A few years ago, the chips only contained building blocks, such as registers—units for holding information temporarily. But now in the mid-seventies

they have come to contain whole computers, or large sections of them.

The first biggies were from Intel: the 8008 and then the 8080, a chip that has become the heart of the Altair, as well as rival computers.

New computer chips keep coming out, people keep telling me to mention specific ones, but I can't keep track of them.

Because of the chips, the price of computer main memory is collapsing space. Something like a dollar a word in the sixties, it is something like a dime a word now. But Intel now offers a storage chip holding 16k bits for \$55, which is 3 cents a bit, and a friend of mine estimates that memory chips will cost  $\frac{1}{10}$  of a cent per word in 1976.

These cost collapses cause many to predict the end of disk and tape. But that's premature. While these zappier chips hold a lot for a little, their contents disappear when the lights go out. Until laser-punched tape comes along, disk and magnetic tape will be very much with us as long-term backup and storage devices.

*Transistors were invented at Bell Labs in 1947 by Bardeen, Brattain, and Shockley; but yes, the computer chip was created by the arms race. These tiny inscribed and incredibly filigreed circuit chips are the hearts and minds of today's electronics, and the inferno of defense preparation, or perhaps inferno preparation, that has gripped our world since that time, has polished the technology into its remarkable state.*

*And most of the computers on sale are built out of these pieces.*

*With chips, as with computers, there are a few Great Numbers. Let's do them.*

## TWO FAMILIES

*Basically there are two main families of chip computer.*

## THE 8080 FAMILY

### SUPERCHIP!

The history books ten years from now, if any, will note that the first computer-on-a-chip was produced by Intel. Intel, an astutely managed company, chose to make a microprocessor that would be suited to placement in others' machines at low cost. This means that if you make a fancy bulldozer or bake-oven, and want it to have some form of intricate pre-planned behavior, you'll put "the Intel chip" in it.

Actually the Intel chip is a number of separate chips, which start low in cost—a fairly complete set can be had for under \$500—and can be assembled into a full computer. (Indeed, various firms do offer complete computers built out of Intel chips. Including one the size of an Oreo cookie, guaranteed for 25 years.)

While these individual chips cost under a hundred dollars each, memories and other necessary sections cost extra. For people who want to develop systems around these chips, Intel has cannily prepared a number of setups.

Crafty and clever Intel, which has captured much of the overall market already, has now brought out much faster versions of these chips. Rah.

*The design was originally created by Vic Poor and Harry Pyle at Datapoint. According to the industry legend, Intel contracted to put it on a chip but couldn't deliver. (Miffed Datapointers point out that Intel brought out the 8008 all too soon after that.) The contract was cancelled; but both companies went their ways peacefully, and prospered. Datapoint continued to build from the original design out of discrete components, Intel brought out faster and faster new versions with bigger and bigger architecture.*

*Like the Roman Republic, which changed by extension into the Roman Empire, the Byzantine Empire and the Holy Roman Empire, this family of chips went from one thing to something else entirely. The design was never meant to be a computer; just a chip that would run a terminal, and let the user select options.*

8008 (the superchip above) was first, and remained quite like the Datapoint machine. Then came the 8080, a more powerful design that could address a whole 64K reasonably, had a proper stack, and better instructions.

This was built into several computers before the Altair. But it was the Altair that made the great publicity splash (see p. 17).

For a time a double model called the Z-80 was popular.

## THE 8088

*and its later models: 8086, 286, 386*

*The perky little 8080 chip, chugging through the seventies, generated so much software momentum that it had to be kept alive at any cost. Now the 8088, too, is immortal, having been anointed the Official Chip of the IBM PC. That stuck us with it forever. The 8086 is a Stretch 8080, and the 286 and 386 are cosmic expansions of that.*

*Also IBM has bought part of Intel.*

*So the world will be stuck with this baby for a long time, cranked up and sped up and elongated and inflated and whatever else they can think of.*

*The 286, which has privileged instructions, and can thus time-share properly, is in the IBM AT.*

*The 386, which runs very fast, is now in new machines from Compaq and others.*

## THE 6800 FAMILY

*The Motorola 6800 and its descendants are the other main family of chip computers today. Actually the 6800 stems from the PDP-11; it uses basically the same types of address modes (see "The 11's Magic Modes," p. 91). Thus it does not have the memory-segmentation uglies of the Intel family.*

## THE 6502

*The 6502 (a variant of the 6800), originally made by MOS Technologies, was at the heart of the Apple II, and the Apple IIgs has a larger offspring of the chip.*

*The 8088 design is what happened to the 8080 when they had to make it bigger; its initially clumsy design was cleverly expanded. The good news is that they increased the instruction set (giving it multiply and so on), making it more symmetrical. The bad news is its broken address space.*

*Tragically, the 8080 had 16-bit address pointers, only enough to select among 64K at a time. That means that segments of either programs or data can only be 64K in length, and have to be combined inside the software by the juggling of four segment registers.*

*As the salesmen say, "That's just a software problem."*

*But as the programmers say, "(!)!"*

*(Fortunately, programmers in higher languages don't have to deal with the messiness and segmentations of its architecture, but those who build the language processors do.)*

## CHIP POWERS ARE DOUBLING ROUGHLY EVERY TWO YEARS

And the price of a given chip goes down—first a few hundred dollars, then a few dozen, then a couple of bucks.

The capacity of an individual chip has now doubled every couple of years for about two decades.

This extraordinary rate of progress is what is driving the computer world so furiously.

## HOW MANY ORDERS OF MAGNITUDE ARE LEFT?

Power and speed can go on doubling every couple of years for—how long?

Some say six years. Some say ten. If Eric Drexler is right (see p. 96), the limit is in storing bits with just one or two atoms, which should take about twenty years from now at the going rate rate.



## THE FAIRLY CLEAN MACHINE: THE 68000

The big one, the 68000, is essentially today's top-of-the-line processor, to be found in the Macintosh, the Amiga, the Sun and other tough small computers.

It's a reasonable design, with a true 32-bit address space, and thus can support enormously large bodies of programs and data without register-fiddling. In principle, billions of bytes of core could be put on board, and indeed the Mac II has that much addressing.

UNFORTUNATELY, in the hurry to get the machine out, various design glitches got put in the 68000 permanently,

and so they've had to stay in the later versions, so the 68000 is not as good as it should be, but it's a lot better from a programmer's point of view than the 8088. And this will in turn mean that operating systems for 68000 systems can be more general and less oppressive than operating systems for the 8088 family.

## THE SEPULCHRE. THE BEST, but not the most successful: NATIONAL'S 16000

The National 16000 chip is everything the Motorola 68000 should be, but it came about a year too late. It will probably never catch up with the momentum

of the Motorola. A friend of mine calls it "a study of architectural elegance," much as the PDP-11 was in 1970. And so it will have a market niche determined by those designers who prefer to go with the best rather than the standard.

## BUSSES AND GLUE: THE REVOLUTION IN PACKAGING

But the revolution is not just the chips by themselves. By themselves they would just be jewelry. Think of chips as a revolution in **packaging**, especially of computer parts; and the chips are just part of this revolution.

COMPUTER CHIPS ARE NOW WHOLE SECTIONS OF COMPUTERS. But only chips do not computers make. To go with them there is a whole new system of packaging.

## GLUE

The term "glue" refers to the interconnectors—the parts that allow chips to be hooked together. In the 68000 series, for instance (see below), a variety of "glue" chips provide the interconnections. Put all together, the main chips and their lieutenant chips and their glue chips are called the **chip-set**.

But the term "glue" also applies to the connections between parts from outside, chips from other manufacturers, signals from over the hill. Whatever enables a chip-set to talk to the common chips of the

field—memory chips, clock chips, calendar chips, input-output chips—is all glue.

## BUSSES

With the packaging of chips have come the pathways for putting them together. These essentially evolved from DEC's Unibus.

## THE GRAND BUS

In electronics, a "bus" is a common connector that supplies power or signals to and from several destinations. In computers, a "bus" is a common connection among several points, carrying a complex parallel signal.

The Grand Bus, a new idea among computers, is catching on. (The term is used here because the colloquial term, "Unibus," is a DEC trademark.)

Basically, the Grand Bus is a connector of multiple wires that goes among several pieces of equipment. So far that's just a bus. But a Grand Bus is one that allows the different pieces of equipment to be changed and replaced easily, because signals to any common piece of equipment just go out on the bus.

This means that the interface problem is deeply simplified, because any device with a proper bus interface can simply be plugged onto the bus.

The idea is great in general. For your home audio equipment, for instance, Grand Bus architecture would simplify everything.

## WORLDS BEYOND

So far the chips have not changed the architecture of computers, though many think so. In fact computers are still designed out of registers, program followers, logic circuits, core memories ("core" is a fighting word, see p. 1). But it makes sense to look at other and better ways we could be doing it.

## BUBBLECHIP

Lately overlooked is the **bubble chip**, a maze for so-called magnetic bubbles. These tiny circular fields are guided through mazes, sluices and raceways; their collisions can do all the things other circuit logic does. They're slow, but very reliable and require little power. Bubble memories are popular, but processors can be bubbled just as well. Another option for parallel architecture.

**Key computer busses presently include:**

**The Unibus on the PDP-11.**

**The S-100 bus (started by the Altair, did not catch on greatly in the professional engineering market).**

**The Apple II bus, designed by Steve Wozniak.**

**The IBM PC bus.**

**The Intel Multibus (allows multiple processors).**

**The Q-bus on the LSI-11 and MicroVax.**

**The Massbus on the VAX.**

**The VME bus on Sun computers.**

**The NuBus on the Mac II.**

THE

## UNIFIED Address Space

Part of DEC's original Grand Bus concept was the fact that everything was on the bus—memory and external devices both. This meant that what you put into a location either went into memory or into some external device, and treating them alike made programming a lot cleaner. And that, too, has remained a part of the Grand Bus concept of this packaging revolution.

Because of the action in chip technology, a potentially important movement in computer design may have been passed over: the "macromodules" developed at Washington University in St. Louis by Wes Clark (father of the original DEC modules) and associates.

The basic idea of the macromodule approach was to have computer sub-sections that were completely interpluggable. With them you can build any computer, to your own design, in a couple of days. The system exists now and it works just fine: counters, registers, memories can be attached quickly by cable.

Unfortunately, the cost is high and they haven't found a manufacturer. With chip prices falling, and chip know-how widespread, it's hard to justify charging ten or so times as much for components just because they can be plugged together faster. (Just as, unfortunately, everything in the macromodule system is built on sections of twelve bits.) For this reason the St. Louis folk are having trouble getting commercial sponsorship. However, perhaps some bright hungry chip company, reading this, would like to get into the macromodule game. And presumably whittle the module down to the now-universal 8 bits.

## HOT-WEATHER SUPERCONDUCTORS

Incredibly fast transistors, and thus circuit chips, can be built using superconductors, those odd substances that lose all electrical resistance in the deep cold. But lately some materials have been found to superconduct in much warmer circumstances—like 200 below—and now (it is rumored) even **room temperature**.

That could blow the computer field wide open, leading to superpower super-speed computers at very low cost in less than a decade.

## THE TRANSPUTER

The Transputer, from the English firm Inmos, is one of the cellular machines mentioned above. It should be a powerful new performer in the next generation of computers—both personal and scientific.

Conventional computer design is still the basic von Neumann machine—one main program follower, lots of memory. But people have said: hey, now the processor chip is as cheap as the memory chip—so why can't we have a lot of processor chips going at once?

The best machine designs are done by people who know software, a principle pioneered by DEC long ago. So the Transputer was designed partly with the help of C.A.R. "Tony" Hoare, analyst of structured programming, and the Transputer was designed for the resulting Occam language (see p. 63).

Occam and the Transputer are also designed for pour-through programming. The programmer **needn't know how many Transputers there are in the array**; the program falls through the menage as fast as it can, with additional Transputers speeding it up like additional holes in a sieve.

Making your own chips is getting increasingly practical, though not terribly. In principle, making a chip is no different from compiling a program. **BUT VERY EXPENSIVE**—especially mistakes—so you want to simulate thoroughly in **some language**, then compile the chip from the already-successful program without modifications (which might introduce bugs).

Last I heard, a firm called Silicon Compilers was the leader—because their chip-design language was so good.

# ADVANCED PROGRAMS

In the early throes of computer enthusiasm, it is easy to suppose that *anything* can be done by computer, that is, anything involving the chewing or diddling of information. This is decidedly not so.

For instance, it is easy enough, and often practical, to have a computer do something a few *million* times. But it is almost never practical to have a computer do something a *trillion* times. Why? Well, let's say (for the sake of simplicity) that a certain program loop takes 1/1000 of a second. To do it a thousand times, then, would take one second, and to do it a million would take a thousand seconds, or about seventeen minutes. But to do it a *trillion* times, now, would mean doing it 17,000,000 minutes, or over thirty years.

Now, you will note that even if you speed up that loop to 1/1,000,000 of a second, a trillion repetitions will take almost twelve days, which is obviously going to need some justifying, even assuming that it is otherwise feasible.

(For problems of this type people begin thinking about building special hardware, anyway.)

In fact, with today's supercomputers, there are now an elect few who have written programs that have run trillions of operations (see "The Teraflop Club," p. DM 101).

## COMBINATORIAL EXPLOSIONS

One kind of thing that's too much to do is generally called a *combinatorial explosion*—that is, a problem that explodes into too many things to do. For instance, consider the game of chess. Just because you can write a program to look ahead at all the possible outcomes of, say, tic-tac-toe, that doesn't mean you can consider all the possibilities of chess. To look at "all" the possibilities just a few moves ahead involves you in *trillions* of calculations. Remember about trillions? And it turns out that there are a lot of problems like that.

Then there is the problem of "Turing impossibility." Turing was a mathematician who discovered that some things *can* be done sequentially in a finite amount of time, and some things *can't*, such as proving certain types of mathematical theorems. In other words, anything that has

to do things in sequence—whether a computer or a mind of God, if any—cannot possibly know anything which is not Turing-computable. Another important limitation.

On a more practical level, though, there are just lots of things which nobody has figured out how to do in any feasible way, or are just now figuring out different systematic ways of doing. (For a favorite such area of mine, compare the different computer half-tone image synthesis systems described on pp. DM 101-119.) Thus you see that *figgering out ways of doing stuff* is still one of the principal aspects of the computer field. (Whole journals are devoted to it, such as CACM, JACM and so on.)

But then of course, every few years there comes a new movement in the field that bodes to make us start all over.

One such trend is called *structured programming*, being promulgated by a Dutch researcher named Dijkstra, among others. The idea of structured programming is to restrict computing languages in certain ways and "eliminate the GO TO," i.e., no longer have jumps to labeled places in programs. By dividing computer programs up only in certain ways, goes this school of thought, the programs can perhaps be *proven* workable, in the mathematical sense, rather than just *demonstrated to work*, as they are now—a notoriously error-prone situation. If the Dijkstra school is correct, we may have to start all over again with a new bunch of programming languages.

These remarks give you the flavor of some restrictions and lines of development.

## OPSYSTEMS, perhaps better called OOPSYSTEMS

THE GREAT SOFTWARE PROBLEM Basically, an *operating system* is a program that supervises all the other programs in a computer. For this reason it is also called a *supervisor* or a *monitor*. Because the operating system is supposed to be in charge, many computers now offer special wired-in instructions that only the operating system can use. This prevents



# Systems People

are the folks who bring you the computer.

That is, they're the ones who try to keep the operating system working. And make the changes it needs to adapt to new equipment and working rules and schedules and software. And change the parts through which mischievous users crash the system.

Systems people often look like dirty rats to users of computer systems. To each other they often look like harried, overworked, unsung heroes, their fingers (and whatever else) in the dike, trying to hold back the tide of Disorder.

Systems people deserve more thanks than they get. Thank you, systems people.

## Operating Systems Trickery

Swapping means transferring one user's program out of core memory in order to move in somebody else's program. This can happen very rapidly, and even when it's done to you every turn, your terminal may seem to respond as though you are in continuous possession of the entire computer.

Paging is one of the Great Abstruse Problems of modern operating systems. The problem is this: you've always got fast expensive memory and cheap slow memory. How can the operating system store most of your program in cheap slow memory and still predict which parts you'll need soon enough to get them in there for you? In the hotter systems, indeed, the operating system tries to predict what's most important and move it to a fast little memory called a cache. This area is so bizarre and complicated I prefer not to think about it. "Minis for me," says Mr. Natural.

## Operating SYSTEMS In Art

Remember the Firesign Theater, back in the sixties? Their album "We're All Bozos on This Bus" contained, according to my sources, recitations of commands for the PDP-10 operating system.

The more recent movie "Terminator" featured the Apple DOS listing in Assembler. As you see the world through the eyes of the evil robot, played by Arnold Schwarzenegger, what should be playing through his head but the commands of our perky little chum, the li'l old, good old Apple II?

other programs from taking complete control of the machine.

Operating systems are big and hard to program. They take a lot of the computer's time: for instance, Dartmouth's time-sharing operating system, taking as much as 23% of the computer's time, is considered efficient.

*From the user's point of view, the operating system is the program that always hangs around, even when you think you're dealing with some other particular program. You see it as the commands you give the computer between what you think of as "programs."*

*The operating system is in an important sense the basic personality of the machine: the tangible, outstretched hand of your computer, with which you have minute-to-minute contact, ready to do your bidding or to slap you if you don't watch out. Most of it's on disk, but some usually stays in permanent fast memory.*

*In the Apple II, it's unobtrusive; on the Macintosh it's visual and quirky. On biggies, and on the PC, it's obtrusive and requires a lot of work.*

Operating systems come in all sizes. The bigger ones take up a lot of computer

time because they have to do a lot. The smallest kind, which are really kind of different, are just to help a single programmer move quickly between his basic programs. This system is really a kind of butler that keeps track of where your basic programs are stored on disk and brings them in for you quickly.

*(Typical operating systems of this type are Apple's DOS, used with the Apple II, and MS-DOS, which you get with a PC.)*

## Of this type are the OPERATING SYSTEMS OF THE DINKY WORLD

### APPLE DOS—Its Own Tradition

Woz designed the Apple II (just for fun, he claims, and to impress the guys down at the Homebrew Club—see p. 18). He not only did a marvelous job of it, but he gave it an excellent operating system as well. The commands were simple. It was precisely the right kind of easy system for that friendly pup of a computer. While it didn't have the power or concision of CP/M, it allowed long filenames and had a certain cheery informality.

There are two kinds of people, those who can learn to tolerate existing operating systems and those who cannot. The latter comprise 85% or more of humanity.

*Some people say that operating-system programmers are anal-retentive. Nonsense. This is the sort of remark that's made by irresponsible users who aren't careful about what's in their*

**A colon:**

**B colon:**

**C colon:**

**D colon:**

Q: How is MS-DOS like MSG?

A: Both of them raise your blood pressure and give you a tightening sensation around your forehead.

Q: Which is better?

A: It depends on the application.

## CP/M

Soon after the Altair came out, Gary Kildall created CP/M, the operating system for it. This was very much in the DEC tradition: crisp, concise commands for moving files back and forth, printing, deleting and copying. CP/M was a felicitous adaptation of the DEC-type operating system. Like the ancient operating systems of the sixties, it was able to work with a very small paper-tape setup on up to a disk system. CP/M quickly became a standard in the dinky world.

CP/M is basically a set of routines which can be called by your programs or by you at the keyboard. It is strongly related to the architecture of the 8080 chip: programs under CP/M must interact with specific registers of the 8080. This has hindered its portability.

## MS-DOS

MS-DOS was originally created by Seattle Computer Products. Microsoft adapted it for the original IBM PC. Thus baptized by IBM, it became the standard of the PC world. It is important because the PC is so widespread. Too bad it isn't better.

MS-DOS is ungainly, complicated, dangerous, inconsistent, and certainly not for beginners. Great concentration is required to keep from losing your files. It is impossible for a beginner to set up and change, though it must be changed for every change of printers, storage devices, or accessory boards.

Microsoft is not really to blame for the more painful features of MS-DOS. They had to be able to field a product in a short time.

We may think of MS-DOS as CP/M the way it might have been modified under rumors of UNIX, augmented with a labyrinthine directory structure that can't be changed, commands conspicuously absent like missing teeth.

But you have no choice. Within the PC world, either you have to stick to one program or you have to know a lot about MS-DOS.

Various products exist to alleviate some of its problems (for example, with menus instead of commands) but to install any of these products, you have to know all the things they're intended to shield you from. ■

## MS-DOS In A NUT SHELL:

**COPY A + B A**  
**will combine**  
**A and B into**  
**a file called**  
**A;**

**COPY A + B B**  
**will destroy**  
**B, replacing**  
**it with A.**

## PIP, Ancestor of the Dinky System

From the beginning DEC computers were sold with paper tape. (Magnetic storage was a big extra that most people didn't have.) So DEC supplied utility programs that helped you punch new copies of your programs and data onto more paper tape. PIP was the name of some of these routines, for moving files between paper tape, magnetic tape, disk, and printers. Its convention that printers and other equipment are "virtual files" passed into UNIX.

# OOPS, what I meant was- GOOD WAYS TO LOSE DATA. (THEY'VE ALL BEEN DONE.)

**COPY THE OLD VERSION OF A FILE ONTO THE NEW.** Note that this is very easy to do if you use both CP/M and MS-DOS. Horrifyingly and amusingly, the command in CP/M,

**COPY A=B**

does the opposite of the MS-DOS command

**COPY A B**

In the first case B will be copied under the name of A (equals means "Gets"). In the second case A will be copied under the name of B.

**BACK UP THE OLD DISK ONTO THE NEW.**

**ACCIDENTALLY FORMAT A FULL DISK.** Best of all, when you meant to format a floppy in the B drive of a PC, type

**FORMAT C:**

and format the hard disk instead.

**TAKE OUT THE DISK** to lend to a friend (for just a minute) before you've exited your program.

**HIT THE RETURN KEY** accidentally before your command is finished.

**HIT ANOTHER KEY** along with the Return key, changing the command accidentally.

**TYPE**

**DEL filename**

when you meant

**DIR filename**

**HIT THE WRONG MODE KEY** (Shift, Alt, Control)—carrying out the wrong command.

**HIT THE ASTERISK KEY BY MISTAKE**, as in

**DEL filename.\***

thus deleting all copies, rather than the ones you wanted.

**RENAME SOMETHING BY ACCIDENT**, thus losing track of it.

**HIT RESET** on the Apple II or Control-Alt-Delete on the PC.

Then there are the things that can go wrong inside an applications program. For instance,

**SELECT THE WRONG MENU COMMAND.**

**QUIT BEFORE SAVING** (in a program that doesn't save automatically).

**GET A "DISK FULL"** message when you try to exit your program.



## Datapoint's DOS and RMS

Datapoint was, in 1970, a spunky little company that made a clean little terminal at a low price (about \$2500). Actually it was a computer, designed by Vic Poor and Harry Pyle just to allow people to reconfigure its terminal characteristics in software.

Much to everyone's surprise, customers saw the potential for fully programming it as a computer, and orders swept in. This minimal computer was in fact the prototype of the 8008 and grandfather of today's PC (see pp. 97, 48).

Harry Pyle, the Tesla of Datapoint, then created an easy and extensive user language, Databus. That was a big hit too. Then Databus was expanded to a

time-sharing language, DataShare, under a simple operating system, which they called DOS.

Then, in 1977, Datapoint brought out the ARCNET. In one of the two mistakes that killed the company, they wouldn't tell anybody what it was, except for VERY qualified customers.

But the ARCNET was the first and best local area network, no doubt because it was created—and programmed—by Harry Pyle. Its DOS was a fairly simple variant of the old DEC-type system—except now they managed to make it work over a network.

This was likewise a spanking success, a *simple* operating system that could be used across a network of up to 255 computers.

Fred Brooks, one of the designers of the 360 and the man who built the operating system, has set forth his apology in a book called *The Mythical Man-Month*. They learned that putting hundreds of programmers on the problem did not get it done faster or make it better.

Note that UNIX was done by two guys.

But these things are much clearer to people now. A big programming project like this combines all the problems of the design of ideas (see *Virtuality* section, p. DM 68) with the real-world problems of making things work. And the bigger the team, the harder it is to consider the choices reasonably or pull the strands together.

## The Problem Of Separate Files▼

You want to keep your work all in one piece? Sorry. You have to break it up into separate entities, called files, and give each one a name. You must be a mother hen to hundreds of these things, the disappearance of any of which is irrevocable. Most ironic of all, you are well advised to make paper lists, outside the computer, of the names of these files and where they are stored.

If you think of your work as a large whole, you may wonder why you must divide this unified whole of your concerns up into separate files and give them a lot of stupid little names—a hornet-swarm of names buzzing around your head. And the answer is, that's how it's always been done.

## Cleaning Up Your Files▼

The problem of file management is horrendous (they proliferate like bedbugs, you can't tell what's in them with those bloody short names, and so you have to keep running programs to look at them or compare them).

"Cleaning up your files"—making sure they're in the right catalogs, and you're rid of what you don't need, requires endless finagling and primping. The people best suited to it have the personalities of stamp collectors, tweezing little scraps from place to place.

Most people and offices just let them proliferate helplessly and keep buying new disks.

But then came the other decision that killed the company: the decision to build a much grander and more versatile operating system. Thus was born RMS, or Resource Management System.

The design of RMS was too hard to use: close to MS-DOS, with a forest of complex options that had to be typed in by users—despite the fact that **most Datapoint workstations were used by secretaries and clerks, not programmers**.

However, RMS could deal with huge files, worked reliably over a network, and had enhancements lacking in MS-DOS (like a MOVE command that would move a series of files, deleting only those originals which had been successfully copied). But it was too different and far too complicated. (In setting up passwords, there were so many options that you were more likely to lock yourself out of your files than protect them from others.) Datapoint's attempt to wallpaper over the system came too late (and was itself too complicated). They brought out VistaView and VistaScript, a nice window package and a good system for choosing options for a complex command. But these came too late, and were too complicated for most people to install. And the window package, for various reasons, could not be adapted to their enhanced word processor.

Given the RMS operating system, Datapoint's gradual collapse in the mid-eighties was inevitable.

## OPERATING SYSTEM/360, or OS/360, or OS

We have no space here to discuss OS, the operating system of the IBM 360 and 370, which is just as well: it is a notoriously heavy-handed system, elaborated with what some would call devastating messiness. Kinds of convenience taken for granted by users of such computer systems as the Burroughs 5000, the PDP-10, DTSS and others aren't there. The programmer has to concern himself with intricacies having names like ACONs, VCONs, TCBs, ECBs, and the complications of JCL (see p. 63). (While these other systems may have equivalent complications, the programmer need not mess with them to create efficient programs, as the 360 demands.) The programmer must even set aside the previous programmer's information in "SAVE AREAS," which is like a restaurant guest having to clear the dirty dishes on

sitting down—and return them when he leaves. Several of the 360's sixteen general registers are confiscated. Time-sharing requires its own JCL-type language. And so on.

[The basic flaw with IBM's OS was that it began as a] Batch Monitor, or operating system set up for Batch Processing (see p. 102). In batch processing, programs go through the computer as if on a conveyer belt, one at a time (or in some systems several at a time). The operating system shepherds them.

Batch processing is used when programs don't need any interaction with human users. (Or, and this is more common, when human users want time-sharing but can't get it; see nearby.)

Despite all the Known Bugs, IBM has managed to patch its mainframe operating system to a certain level of reliability: a task roughly comparable to repairing the Titanic as it was sinking. It now has numerous manifestations and varieties, but their distinctions, as the old joke says, "would be of interest only to another rhinoceros."

## "DESKTOP" OPERATING SYSTEMS

On the Macintosh and systems that copy it—like Digital Research's GEM, on the Atari—you don't type commands. You see *The Desktop*—that is, a strange arrangement that perhaps looks to some people sort of like a desktop on the screen. Your files are represented by little file folders (third cut, center tab).

The beginning Mac user doesn't have to Learn an Operating System in the same sense as a user of either the Apple II or the IBM mainframe—typed commands to make the machine do things. But the operations of moving the pictures of files around are equivalent to some of the same operations.

The basic idea of *The Desktop* is correct: give the user a visualization that corresponds to what the user wants and must think about. The only problem with the "desktop" is its shallowness as a visualization. What is needed is **more profound and extensive** visualizations, not just for moving and copying files, but for searching, ordering, merging, and directing bunches of programs to operate on bunches of files.

Such visualizations are not yet known. (See "Virtuality Design," pp. DM 68-74.)

## IT DOESN'T HAVE TO BE THIS WAY

If you enjoy juggling straight razors, then you'll enjoy working with today's operating systems. They require the concentration of a jet pilot or a brain surgeon, and you are in constant danger of losing everything you're working on. Eternal vigilance is the price of survival. (The "Desktop" systems are slightly better—but even there you have to think hard about which disk you want to copy to the other.)

**Operating systems as now constituted are a principal evil of the computer field, and one of the principal, proper reasons a lot of people have for hating computers. The design of today's operating systems is highly punitive, and they constantly distract you with matters not of your current concern.**

The technoids don't see this, nor how greatly they are imposing on non-computer people. ANYBODY WHO DOESN'T SEE THAT OPERATING SYSTEMS ARE A MONSTROUS PROBLEM FOR CIVILIANS AND BEGINNERS,

## SHOULDN'T BE ALLOWED TO PROGRAM FOR CIVILIANS AND BEGINNERS.

Programmers can walk in the Valley of the Shadow of Death if they want to; they need extensive access to operating-system facilities with the blade-guards off. But this should not be required of civilians.

## WHITHER?

Operating systems have to change, two ways.

One is in the design of the interface and virtuality (see p. DM 68). The other is internal change, to make operating systems more robust, and able to operate over networks and in multi-network situations around the world—circumstances prone to both foul-up and mischief.

In the old operating systems, a program got what it wanted inside the computer. It might have to supply some kind of a password, but it could often dodge around these barriers—if it could get access to all memory, for instance.

In the newest experimental operating systems, security is handled by a new kind

of element called a *capability*. A capability is the operating system's private memo saying you're entitled to use a thing. If you ask for something without the capability for it, you don't get the information required even to find it. This mechanism controls access to memory, files, and running tasks.

The capability approach has been worked on in England and, in the United States, at Carnegie-Mellon.

## TIME-SHARING

Then there is time-sharing. Time-sharing means the simultaneous use of one computer by several different users at once.

In principle this is like a lazy Susan. The central computer works on one user's program for a while, then another's...until it is back to the first user.

There are basically two kinds of time-sharing: time-sharing where you can only use certain facilities or languages, and time-sharing where you can use all the facilities of the computer (including programming in the computer's assembly language).

Examples of restricted time-sharing are the various minicomputer systems that are available which time-share the BASIC language.

Some examples of unrestricted time-sharing are the PDP-10 (see p. 87), Dartmouth's DTSS, Honeywell's MULTICS, IBM's TSO, and General Electric's MARK III.

Restricted time-sharing, with only one or a few languages offered, is much easier to provide for than full time-sharing.

Full time-sharing is always *shared* with batch. In other words, the computer, darting among users, still finds some time to devote to the batch stream.

Time-sharing is self-limiting. That is, the more users that are signed onto a time-sharing system at a given moment, the more slowly the system responds to all of them.

The importance of time-sharing is not in terms of "raw" efficiency, that is, the cost of each million operations, but in terms of human efficiency, the ability of each user to get so much more out of the computer by using interactive programs and languages.

## The Tyranny Of The File

In a recent paper, "The Tyranny of the File" (Datamation, 15 December 1986) I argued that conventional file structures—which go with conventional operating systems—are implicitly responsible for the structure of computer programs, in much the same way that the time slot is implicitly responsible for the structure of schools and television (see p. DM 58).

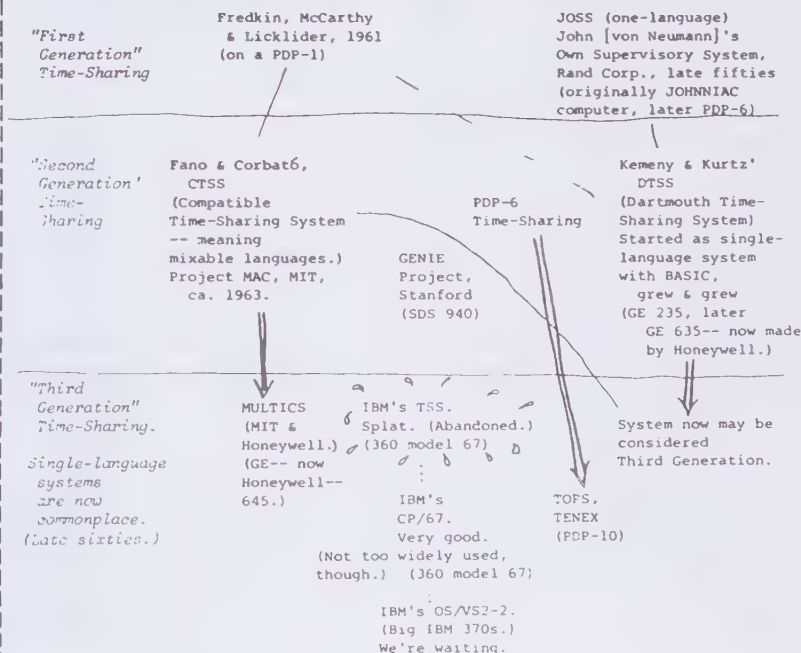
The way it works is this: specific forms of files have evolved for convenience, and as these file types evolved, corresponding program types evolved around them—"word processing," "databases," and so on. It is hard to combine these types of programs because files weren't done right in the first place.

The radical alternative is a storage system that allows the sharing of materials between files, shows their origins, and delivers the materials as needed from their original locations. This is the Xanadu system (see p. DM 141.)

Since, under Xanadu, only additions and pointers are stored, the vast waste of space that goes on with conventional operating systems does not occur; new documents and versions may be made out of old material at little space cost; and so the problem of "purging" files becomes unimportant.

Unfortunately, the Xanadu system does not work with conventional software, so somebody may have to write an operating-system simulator which delivers Xanadu files to conventional programs. Or better, develop new software to work with this new model.

## SOME IMPORTANT TIME-SHARING SYSTEMS (very incomplete)



(from the 1974 edition)



*“UNIX  
is the  
joint  
product  
of every  
university  
and  
college  
in the  
country.  
And it  
all ended  
up under  
the trade  
secret of  
AT&T.”*

—A Cynic

**I**n one university environment where I used to hang out, they had an early model IBM 370 mainframe. Sometimes, in the middle of the night, the computer's operating system would inform the guys I was working with that their account had run out of money. They would then have to go to the console and look at a map of main memory until they found the slot that said how much “money” they had. They would put in more “money” with the light pen, then proceed. It was an amazing thing to watch.

## DTSS

DTSS is the Dartmouth Time-Sharing System, and let it be an example to us all. It was created by Kemeny and Kurtz, who created the BASIC language to be used on it (see p. 59). Their computer arrived in fall '63. Their time-sharing system went into operation in spring '64, programmed mostly by Dartmouth students, and has grown and improved continuously since then. On that basis: programmed by students.

It's great.

The Dartmouth computer philosophy—i.e., the idea carried through by John Kemeny and Tom Kurtz—was that a computer is like a library: its services should be free to all in a community, paid for through some general fund. Students and faculty at Dartmouth use it free (unless they have grants). You can use it too, if you pay.

The result: everybody at Dartmouth uses the computer. It's always running, (ahem) six days a week. There are almost two hundred terminals around the campus; peak afternoon usage is about a hundred and fifty. Freshmen learn BASIC first thing, after which the com-

puter is a standing facility, to be used in courses in music, sociology, literature, history, engineering, or whatever; for independent research; or just for fun and games and showing off to visitors.

The entire Dartmouth system is built for simplicity and clarity with explanations of all the facilities available at terminals. (The command *explain JGK* causes the terminal to type out a picture of Kemeny.)

Many fuddy-duddies insist that computer usage should be *billed*, as it is on most college campuses. That is essentially the Calvinist view. But what if we treated libraries like that? It would probably cost \$10 *just to borrow any book*. The point is that if we believe that certain conditions are a social good, then we should be flexible about how to implement them. (See Arthur W. Luehrmann and John M. Nevison, “Computer Use under a Free-Access Policy,” *Science*, 31 May '74, pp. 957–961. This article continues this line of argument and further describes the Dartmouth billing system.)

The most enjoyable session at the 1974 National Computer Conference was the Nostalgia session on the Dartmouth system, DTSS. The Old Hands were there, guys who as kids worked on the original

time-sharing system, and have now become grownups of one sort or another.

An alarming statement was made at that session by Jerome B. Wiener, who said he had been the liaison man between the Dartmouth effort and the computer manufacturer (not IBM). He stated that he had been ordered by his company to stop the Dartmouth “experiment” any way he could, or lose his job in three months. He did no such thing, and (he said) after being fired continued to help the Dartmouth effort, holding weekend meetings with others from that company in his home. He deserves the Frances O. Kelsey we-do-our-real-job medal.

## MULTICS!

MULTICS was announced in 1965 as the Time-Sharing System of All Time, to be created jointly by MIT, General Electric and Bell Labs.

It took a lot longer to get going than they expected—I have a 1968 button that says, **YOU NEVER OUTGROW YOUR NEED FOR MULTICS**—but now it's available from Honeywell. People say it's the greatest, all right—its fascinating facilities include the ability to execute parts of other people's programs, if you have permission—but it's also said to be awfully expensive.

Interestingly, the MULTICS operating system is largely programmed in the PL/I language (see p. 60).

## UNIX

Now there has arisen a far grander operating system, UNIX.

UNIX—the name's suggestiveness of harem guards is deceptive—is really the son of MULTICS (see above). But it was finished in much less time. Like MULTICS, it's a beauty. Like MULTICS, it was programmed in a higher language; the language it's programmed in, however, is simply “C.” The language was created by Brian Kernighan, author of a widely praised book which he audaciously compiled out of incorrect programming examples from other people's books on programming. UNIX itself was programmed in “C” by Ken Thompson and Dennis Ritchie, at AT&T's Bell Laboratories.

UNIX is a demon.

*Note that LIKE MS-DOS, IT IS TOO DANGEROUS AND COMPLICATED FOR CIVILIANS. It does not solve the problem of protecting the innocent. But programmers need and love it.*

EVERYTHING IS  
DEEPLY INTERTWINGLED.

*“An option  
means the pro-  
grammer didn't  
have a clear  
idea of what the  
module was  
supposed to  
do.”*

—Roger Gregory

## UNIX HISTORY

UNIX started out as a small time-sharing system, designed especially for programmer convenience, and especially well suited to clean programming with small modules. Now it is also used on big machines as well. Programs built with UNIX need UNIX forever.

It was the first major portable operating system; it has been spread around by cheap academic licensing, and carried along by many devout UNIX hackers as they left the colleges. (It has also become fairly standard in Europe and the Soviet Union.)

Unfortunately, the AT&T lawyers slowed down UNIX's distribution, and it's too expensive for most people (still about a thousand for XENIX on your PC, for example). But cheaper and free-type versions are gradually becoming available.

The IBM PC cannot run full UNIX because it has no hardware instructions that can be used only by the operating system ("privileged instructions"). Microsoft XENIX is the best UNIX that can be produced under those circumstances.

## MAIN UNIX CONCEPTS

### PHILOSOPHY

Individual programs are specialized single-function pieces. Each does one thing very well in a simple and understandable fashion. These programs are then chained together into bigger operations through Pipes (see below), which carry data between successive stages of operation. (This is called the "tools" approach.)

A full-bore UNIX program is lots of separate processes with data gurgling between them, connected and synchronized through pipes.

### FORKING AND TIME-SHARING

Aside from all the usual features, UNIX allows programs the magic property of splitting.

This is how tasks are spawned.

A process that calls the *fork* instruction is cloned into two processes. They either go on in parallel or one of them calls something else that is supposed to happen in parallel. Simultaneous copies of the

same processes can be off working on different things. Since multiple simultaneous tasks are intrinsic to UNIX, **time-sharing is not much more than the addition of terminals.**

This sorcerer's-apprentice structure comes from SIMULA (see p. 62).

### PIPES

The **pipe** is an easily-arranged connection from the output of one process to the input of another. If the output of one process comes out sequentially, the next process can immediately begin work on it. Processes can pipe to each other in parallel almost without limit.

### GENERALIZED INPUT AND OUTPUT

There are no special cases of input and output; data may be transferred freely from anything to anything else; meta-physically, something that can give or receive data is called a **file**, even if it's a printer.

### SHELLS

You can create your own UNIX command names with a **shell**.

A UNIX shell is a generalized command interpreter: a program which recognizes command names and calls routines. A **shell script** is a set of commands to a UNIX shell, as if typed in.

Shells may call shells. (Several shells are popular, especially the "Bourne Shell" and the "C Shell.")

### FILES

UNIX files are the usual hierarchical kind, but at least very cleanly usable with a lot of utilities.

### DENOMINATIONS OF UNICES

Various people sell UNIX, but if it's called UNIX it's always under license from AT&T.

The two main suppliers of UNIX are now AT&T, parent of Bell Labs, and, for some reason, University of California at Berkeley.

There have been lots of Unices. The first ones (versions 1 to 7) were built at Bell Labs. Then UNIX itself forked: Version 7 has become AT&T UNIX, Berkeley UNIX, and XENIX. (See diagram, "Know Your Unices.")

## THE SPIRIT OF UNIX CURRENT MORALISMS

Hackers are divided on Who is Faithful to the True Spirit of UNIX. The clean programming style is being muddled, some say, by the addition of dirty options that lead to bad programming.

Berkeley UNIX 4 and its successors (available from various companies) have extra cluttering options on various commands, getting away from what some call the "True UNIX" style of small and exact code modules.

UNIX CONFERENCE JOKE: "Berkeley UNIX is two to three years ahead of Bell UNIX. Bell is way behind on putting in unnecessary features."

**Example.** Suppose you want to list your files on the screen in columns. The traditional way to handle this was by piping the output of the LS command (for LIST) to a column formatter, and thence to the screen.

This was classic UNIX style. The column formatter could be used by any program, and would put any stream of text into columns. But Berkeley made column format an option on LS that can't be used with anything else.

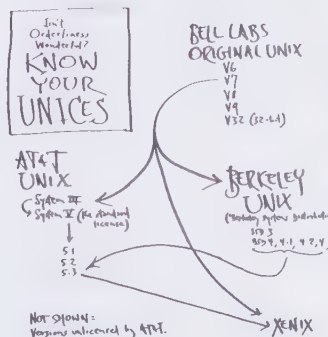
There is a popular license plate worn by programmers on the front of their cars (in one-plate states).



That expresses strong feelings, hard to explain to non-programmers.

We look forward to GNU—the heroic FREE UNIX from high-credibility superbacker Richard Stallman. It will be ready soon.

Meanwhile, versions of UNIX are available for as little as \$200 for the PC.





"Time-sharing is an idea whose time has gone."

## HAND ME THE PLIERS, IGOR

Tanenbaum's book (see Good UNIX Books) describes a small version of UNIX, called MINIX, for which complete source code is available on disk for under a hundred dollars. (Compiler is extra.) This runs on the PC—with a lot of swap delay.

It was designed to be understood, not to be efficient; thus it is excellent for those who want to understand UNIX-type operating-system structure.

On a sunny day, I give humanity a fifty-fifty chance of survival in the near term.

My friend Eric Gullichsen asked if I knew anyone more pessimistic.

I named someone more pessimistic, a very great programmer and teacher, who thinks humanity has less than twenty years, period.

Eric's reply:

"That's because he uses VMS instead of UNIX."

### GOOD UNIX BOOKS (there are lots more):

Understanding UNIX: A Conceptual Guide from Que, Indianapolis, IN.  
Andrew S. Tanenbaum, Operating Systems: Design and Implementation. Prentice-Hall.

### THE ELEVEN-VAX OPSYS QUANDARY MESS

Multiple operating systems are, indeed, the bane of the PDP-11 line. Not only are there DEC's own, like RSTS, RT-11, DOS and RSX, which suffer from a lack of file compatibility and sometimes won't even run the same object code, but now there has arisen a far grander operating system, UNIX.

Unfortunately DEC has been ambivalent about UNIX. Rather than adapt

UNIX, the world favorite, for the VAX, DEC created their own new operating system, VMS, which they said would be everything for everybody.

The sophisticated and scientific houses operate VAXen under UNIX, as do many universities; computer centers are more likely to stick with VMS.

### TSO

IBM's "TSO," for Time-Shared Operating System, is an odd sort of time-sharing they have come up with for the 370. It is a sort of interactive batch programming. That is, it allows the user at a terminal to communicate with programs running in batch mode.

While this is a form of true time-sharing, (though its detractors tend to compare it with what they call "true" time-sharing, such as that on the PDP-10), it has a number of drawbacks.

The bad feature of TSO most often mentioned is its slow response time. That is, response may be sometimes good, sometimes execrable.

IBM is urging its fans to believe that its next operating system, called OS/VS2-2, will be much better.

There have been improvements in IBM's time-sharing, and a continual dance of name changes, but the problems have remained the same. The basic architecture of IBM's OS could not change much, and thus improvements in time-sharing performance had to come mostly from stepped-up hardware crunch. Fortunately, most time-sharing has been able to migrate to smaller machines with UNIX—or to personal computers.

### THE LOGIN HEARD 'ROUND THE WORLD: GE'S MARK III

Some time-sharing systems are local, others have "concentrators" allowing users in other cities to log into them with local telephone calls.

Perhaps the most far-reaching time-sharing system, though, is General Electric's Mark III, with concentrators in many of the major cities of the world (mostly Europe). The main computer is in Ohio, but the overall system may be thought of as an octopus around the globe. Besides hundreds of cities in the USA, the GE system offers local access in Australia, Austria, Belgium, Canada, Denmark, Finland, France, Italy, Japan,

Netherlands, Norway, Puerto Rico, Sweden, Switzerland, United Kingdom and West Germany. What this basically means is that if a company has offices in these places, it can do its internal communication through General Electric's computer system.

This presents obvious merits and difficulties, which there is no room to discuss here. The service is said to be expensive.

The I.P. Sharp time-sharing network is another round-the-world time-sharing system that companies use to communicate among their offices.

### THE SOURCE AND COMPUSERVE

are sometimes called "networks," but in fact they are good old-fashioned time-sharing systems, dressed up in seventies polyester. Turns out they're used very heavily for text, but that's fine. Too bad, though, that the people who originally invented time-sharing didn't realize how much of its use would be for text handling, or things might've been done differently.

### WHERE TO GET IT

No way can we here get into the prose and cons (both senses) of the myriad time-sharing services that are available.

Time-sharing prices are a mix of lotsa stuff:

1. Connect time. This you pay by the hour.
2. "Core charges"—essentially the price of processing itself; depends on amount of number crunching. PDP-10 bills this in kilocore-seconds, i.e., how many thousand words of core memory your program really turns out to need, for how many seconds.
3. Storage, which costs much. Example: 1000 characters for a month for a buck. (Typical.) That adds up fast. Five bucks an hour overall is a pretty good rate. Note that time-sharing usually costs less in non-business hours—but some exceptions charge more.

### BIBLIOGRAPHY

- M.V. Wilkes, *Time-Sharing Computer Systems*, MacDonald/American Elsevier Publishing Co.  
All About Timesharing Service Companies. Datapro Research (1 Corporate Center, Moorestown, NJ 08057), \$10.



# THE HEARTS AND MINDS OF COMPUTER PEOPLE

Computer people are a mystery to others, who see them as somewhat frightening, somewhat ridiculous. Their concerns seem so peculiar, their hours so bizarre, their language so incomprehensible.

Computer people may best be thought of as a new ethnic group, very much unto themselves. Now, it is very hard to characterize ethnic groups in words, and certain to give offense, but if I had to choose one word for them it would be *elfin*. We are like those little people down among the mushrooms, skittering around completely preoccupied with unfathomable concerns and seemingly indifferent to normal humanity. In the moonlight (i.e., pretty late, with snacks around the equipment) you may hear our music.

Most importantly, the first rule in dealing with leprechauns applies *ex hypothesi* to computer people: when one promises to do you a magical favor, *keep your eyes fixed on him until he has delivered*. Or you will get what you deserve. Programmers' promises are notoriously unkept.

But the dippy glories of this world, the earnestness and whimsy, are something else. A real computer freak, if you ask him for a program to print calendars, will write a program that gives you your choice of Gregorian, Julian, Old Russian and French Revolutionary, in either small reference printouts or big ones you can write in.



Computer people have many ordinary traits that show up in extraordinary ways—loyalty, pride, temper, vengefulness and so on. They have particular qualities, as well, of doggedness and constrained fantasy that enable them to produce in their work. (Once at lunch I asked a tableful of programmers what plane figures they could get out of one cut through

a cube. I got about three times as many answers as I thought there were.)

Unfortunately there is no room or time to go on about all these things—see Bibliography—but in this particular area of fantasy and emotion I have observed some interesting things.

One common trait of our times—the technique of obscuring oneself—may be more common among computer people than others (see “The Myth of the Machine,” p. 31, and also “Cybercrud,” p. 27). Perhaps a certain disgruntlement with the world of people fuses with fascination for (and envy of?) machines. Anyway, many of us who have gotten along badly with people find here a realm of abstractions to invent and choreograph, privately and with continuing control. A strange house for the emotions, this. Like Hegel, who became most eloquent and ardent when he was lecturing at his most theoretical, it is interesting to be among computer freaks boisterously explaining the cross-tangled ramifications of some system they have seen or would like to build.

(A syndrome to ponder. I have seen it more than once: the technical person who, with someone he cares about, cannot stop talking about his ideas for a project. A poignant type of Freudian displacement.)

A sad aspect of this, incidentally, is by no means obvious. This is that the same computer folks who chatter eloquently about systems that fascinate them tend to fall dark and silent while someone *else* is expounding his own fascinations. You would expect that the person with effulgent technical enthusiasms would really click with kindred spirits. In my experience this only happens briefly: hostilities and disagreements boil out of nowhere to cut the good mood. My only conclusion is that the same spirit that originally drives us muttering into the clockwork feels threatened when others start monkeying with what has been controlled and private fantasy.

This can be summed up as follows: NOBODY WANTS TO HEAR ABOUT ANOTHER GUY'S SYSTEM. Here as elsewhere, things fuse to block human

communication: envy, dislike of being dominated, refusal to relate emotionally, and whatever else. Whatever computer people hear about, it seems they immediately try to top.

Which is not to say that computer people are mere clockwork lemons or Betheheimian robot-children. But the tendencies are there.

## BIBLIOGRAPHY

Gerald M. Weinberg, *The Psychology of Computer Programming*. Van Nostrand Reinhold.

Systematic treatment in a related vein. Sherry Turkle, *The Second Self: Computers and the Human Spirit*. Simon & Schuster. A powerful and troubling analysis of computer folks' mentality.

There are many kinds of computer people, living in many different computer subcultures, though if you've just met one bunch you may not suspect it. Some of the different groups have practically nothing in common—no more than, say, skateboarders, bus drivers and airline cabin attendants, who are all “transportation people.”

But the one thing we all have in common may be the *COMPUTER-INTENSIVE LIFE*. And that means that every hour, sometimes every minute, we have *Informatic Thoughts*: concerning ourselves with the input, or moving, or storage, or twiddling, or output, of information. Just the way a *Transportation Person* (as most of us are) thinks about, or uses, cars quite frequently.

The strange language of computer people makes more sense than laymen necessarily realize. It's a generalized analytical way of looking at time, space and activity. Consider the following.

“THERE IS INSIGNIFICANT BUFFER SPACE IN THE FRONT HALL.” (Buffer: place to put something temporarily.)

“BEFORE I ACKNOWLEDGE YOUR INTERRUPT, LET ME TAKE THIS PROCESS TO TERMINATION.”

“COOKING IS AN ART OF INTERLEAVING TIME-BOUND OPERA-

TIONS.” (i.e., doing parts of separate jobs in the right order with an eye on the clock.)

## BECOMING A COMPUTER PERSON

—*You Don't Feel It Happen*

Becoming a computer person is an insidious process. *YOU AREN'T AWARE OF ITS HAPPENING.*

You don't notice any change in yourself. All you notice is that you can no longer talk to your old associates: they don't know, or care, about the things that you now recognize as basic and important.

“...programmers, in my experience, tend to be painstaking, logical, inhibited, cautious, restrained, defensive, methodical, and ritualistic.”  
Ken Knowlton, “Collaborations with Artists—A Programmer's Reflections,” in *Nake & Rosenfeld* (eds.), *Graphic Languages* (North-Holland Pub. Co.), p. 399.





## Some Computer People To Distinguish Among

Computer operators turn 'em on and off, change programs, change disks and tapes, select modes of operations for programs that can do more than one thing.

Input typists (also called *keypunch operators*) are clerks who copy information into the computer (on terminals) or onto something the computer can read (punch cards, magnetic disk, etc.).

NOTE: these jobs may end in a few years when nothing else has to be copied anymore because users put things in themselves.

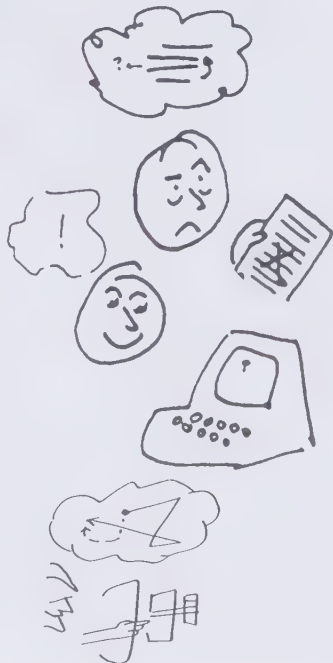
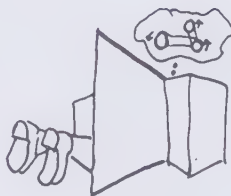
A great deal of keypunch work has moved off shore, especially to the Bahamas.

Computer repairmen, or "field engineers," fix computers and their accessories when something goes wrong electrically or in the gears. They always wear tie clips, at least if they wear ties, so as not to get pulled into rotating machinery.

Computer programmers create exact plans for what the computer is to do, then change them till they work.

A NAIVE USER (no offense) is an ordinary person who doesn't need to know any of these things in order to do something useful with the computer. Creating programs to help him is the frontier of computing.

Interactive system designers are the new movie-makers—designing the interactive movies of the mind for our computer screens: everything from office systems to video games.



## COMPUTER PEOPLE AND SLEEP

Back in the early fifties, a New York disk jockey named Jean Shepherd discovered that there were Day People and Night People. This startling disclosure greatly relieved some of us Night People, who were tired of being hassled about it and doubting ourselves.

When you work late, you decouple from the world: there are no errands, appointments, little deadlines. It's a good way to avoid interruption, distraction, confusion; to pursue loose ends; to get it right. That's what night people like.

Many computer people are not just night people, but have unusual ways of treating sleep—living on other than a 24-hour day, for instance. (I have always preferred about a 30-hour day.) Unfortunately this creates many inconveniences in dealing with Mundanes, offices, restaurants and whatnot.

How should the human body sleep anyway? French scientists lowered volunteers into mineshafts with lots of books and supplies and no clocks, with the instructions: work out your own schedule. Some quite startling body-rhythms emerged, I think involving sleep-cycles of fifty hours or more.

### NEAT READING

Murray Melbin, *Night as Frontier: Colonizing the World After Dark*. Free Press.

### A Hacker's Hacker's Hacker

MARK SAMUEL MILLER

There are many Mark Millers in computerdom, but there is at most one, I am sure, who spells his middle name with a dollar sign, or has conducted weddings as an ordained minister of the Universal Life Church of the High Frontier.

He knows thirty computer languages, and is proud that none of them is COBOL. He also points out that "Mark Miller" is his profession as well as his name: a programmer is really a symbol grinder.

I met Mark Miller in '77: he called up to talk about Computer Lib, and he turned out to be the first reader to take any interest in the Historical Tree section (see p. DM 43). I had him down to lecture my seminar at Swarthmore. He apprenticed himself to me that summer, and by 1979 had become the principal theorist and



programmer of Project Xanadu in its new transformation (see p. DM 141).

Then, seduced by the real world, he went to Datapoint. Until Datapoint's recent collapse, he was that company's top R&D programmer, best known for implementing their VistaView windowing package. At the present writing, he is at Xerox PARC.

Mark clarifies and condenses at the same time. He insists on clean, elegant design in everything, and generally manages to simplify and bring unity to whatever he does, sometimes even retrofitting unification to some preexisting chaos, which is clearly impossible in principle.

The work he is presently concerned with is at once his recreation, his professional research field, his personal salvation and his religious duty. Mark is working on an actor-based, capability-based, market-model network operating system, and is probably one of the few people who can explain what that means.

But his most unusual aspect is his personal plan for immortality. He genuinely expects to live forever. It is Rev. Miller's belief that he will be converted to, and reborn as, a computer program, to live long and prosper, in superior, immortal and generalized form. He will, he believes, be translated into a program (in what language we cannot now know) that preserves his intelligence and personality. By whom and why is not yet decided either. (But if his brain has to be minced up for the conversion to software, so be it. "That's just an I/O problem.") He actually believes and intends this.

So you can see that his current work is an expression of his religion, his profession, his intellectual involvements, and his personal destiny. Mark Miller is building the environment in which he expects to spend eternity, the architect of his own Heaven. And there, dear reader, is the apex of computer religion.

After he had been hired by Datapoint, Mark was asked to fill out a medical form. Where the form asked if he had any physical defects, he wrote:

CARBON BASED.

## LIBERTARIANISM

The Politics of Alienated Computer People

Libertarianism is where the political left wraps around to meet the political right.

political eccentrics  
dation of government  
s ago, some of them  
ver from the Spanish  
e more right wing—  
' science-fiction fans,  
puter people. Their  
list Ayn Rand (The  
science-fiction guru

agreement on Liber-  
ugh it may perhaps  
"There are no public  
ate problems." Gov-  
itimate, since it does  
exact consent of the  
ere should be no laws  
rkets for everything  
slaves). Individuals  
y, required—to work  
ivate negotiation and

ertarian Party, and  
y field a presidential  
poll in New York City  
half of the registered

LIBERTARIANS were computer people. This probably holds elsewhere as well.

Why do computer people go in this political direction? Well, the field is full of talented loners who have felt stifled by school and held back by undeserving fools; in a sense that is their natural political position.

If Libertarians ever really enter the political process, they could be a force. But their doctrine pretty much rules that out.

Many women aren't looking for men; some are. This is addressed to those who are.

## HOW TO ATTRACT COMPUTER MEN

Thousands of computer guys are dying to meet women they can talk to about their work.

It is very easy to meet them. Just go to conferences.

This is not my advice. It comes from a happy young woman I know who goes to lots of computer conferences. Though she happens to be a programmer, she says that's not the important thing. I paraphrase what she has told me.

"I don't necessarily know anything that the conference is about, but conferences are always interesting. And because there are so few women there, the guys swarm around me. I take their numbers and don't give them mine.

"After each conference I end up with maybe forty interesting guys to look up, if I feel like it. And they're all eager to explain whatever you want to know.

"Or if you don't want to know anything, that's cool too. Except maybe then you wouldn't enjoy the guys."

## THE FIRST HACKERS' CONFERENCE

1984 was the tenth anniversary of Computer Lib, the Altair, and Personal Computing. It was also the publication year of Steven Levy's *Hackers*, an excellent interwoven account of great counter-cultural innovations of computerdom—the events and the people that created PDPs and Apples and the yeasty-feisty atmosphere of hackerdom.

In honor of all this, Stewart Brand and the gang from Whole Earth Catalog held an invitational conference, basically for the people who were mentioned in Levy's book.

The conference was held in spartan old army halls in a park on the California coast, and everyone had to pay to come. The invitation enticed us by listing all the other people who were invited, and stressed that you couldn't get in without being invited. Result: over a hundred computer crazies and geniuses converged on the barracks for two nights and days of banter, formal discussion, and hanging out.

The crowd was young; those who built the great old mainframes and languages had been passed over for famous crackers, Lispians and builders of popular market programs. Folk heroes were there, like Captain Crunch, Woz, Bill Budge of pinball-program fame, and Jerry Pournelle, who could frequently be heard a long way off. The Xanadu group did its first public demo of links (see p. DM 141).

Dishwashing and sweeping were shared by volunteers. People stayed up late, got up early, and argued with warmth instead of heat. A religious group awoke a number of hackers with hymns. Raccoons entered the barracks to break into my luggage and eat cookies. And a good time was had by all.

What made it great was the atmosphere. Having experienced fifty con-

ferences, I rather expected a high-voltage ambiance of competition and putdown. Instead, almost everyone was happy and convivial. It was the Woodstock Effect: everyone felt so good about the privilege of being Definitely Where It Was At that all the usual bad vibes and competitiveness of such a setting simply dematerialized.

THE POINT OF IT ALL. With free-form discussion groups around such questions as "Is the Hacker Ethic dead?", the Hackers' Conference came to the fundamental conclusion—I think I state the sense of the meeting—that HACKERS ARE EXPLORERS OF IDEAS, not law-breakers; and Hackerdom will live on wherever there's a kid facing a universe.

Unfortunately the press thinks a Hacker is someone who breaks into computers illegally. This is not what it means.

A layman is someone who snickers at the names UNIX and LISP, not knowing that despite the effeteness they suggest, they are two of the most powerful entities in computerdom.

Someone who says, "Buffer Management Time," instead of "I have to go to the bathroom," is probably a computer person.

"Only Stewart Brand could have made me buy a portable soap-dish."

—A Hackers' Conference Attendee

## Know Any Technicians?

Non-technical people seem to be largely unaware that THE WORD "TECHNICIAN" IS AN INSULT.

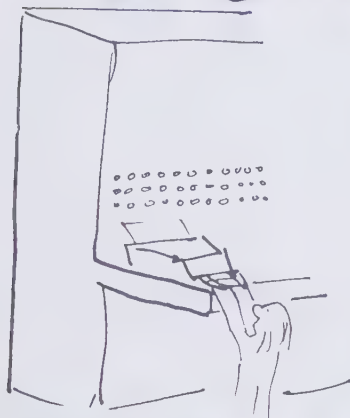
Like the term "lower class," it generally refers to someone else. What "technician" usually means is somebody whose work is less creative than your own.

So probably you do know some technicians. But to say so is neither good etiquette nor politics.



# THOSE ADORABLE INFURIATING RESISTORS.

PDP-8

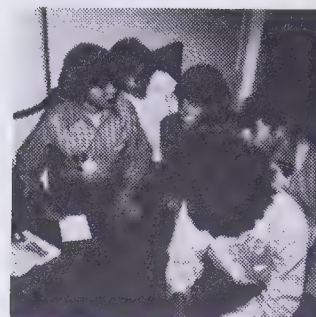


R.E.S.I.S.T.O.R.S.  
after infamous  
Omega ceremony.



Steve at the old straight 8.

PDP-8  
(continued).  
Open for  
fiddling  
inside.



Now here's my  
plan...

A coven of  
R.E.S.I.S.T.O.R.S.  
in executive  
session,  
Atlantic City.



Triple redundancy.



What's a group like you  
doing at a Joint like this?

Their name makes people think they're a war protest group, but actually the R.E.S.I.S.T.O.R.S. of Princeton, NJ, are a bunch of kids who play with computers. They're all young; members are purged when they finish high school. Their clubroom is at Princeton University, but the initiative is strictly theirs.

The name stands for "Radically Emphatic Students Interested in Science, Technology and Other Research Subjects." Computers are not all they do—they've also gotten into slot racing and the game of Diplomacy—but computers are what they're known for. The Resistors (let's spell it the short way) exhibit regularly at the computer conferences, and have startled numerous people with the high quality of their work. They've been invited to various conferences abroad. They have built various language processors and done graphics; lately their fad is working the LDS-1 in Princeton's Chemistry Department.

Where do they learn it all? They teach each other, of course. Newcomers hang around, learn computer talk, work on projects, and tease each other. They also use the informal trade channels, subscribing to magazines and filling out information request cards under such company names as Plebney International Signal Division and Excalibur Wax Fruit.

The great thing about these kids is their zany flippancy. They've never failed, they've never been afraid for their jobs, and so they combine the zest of the young with their expertise. Their forms of expression are as startling to professionals as they are to outsiders: don't say anything ponderously if it can be said playfully. Don't say "bit field" if you can say "funny bits"; don't say "alphanumeric buffer" if you can say "quick brown fox box"; don't say "interrupt signal" if you can call it a "Hey Charlie"; don't say "readdressing logic" if you can say "whoopee box."



They have varied backgrounds. The father of one is a butcher, the father of another is one of the country's foremost intellectuals. (None of that matters to the kids.) I have dined in a number of their homes, and find this in common: their parents show them great respect, love and trust. Indeed, Resistor parents have expressed some surprise to learn that their children's work is at the full-fledged professional level. The important thing, to the parents, is that the kids are working on constructing things they enjoy.

The trade press is ambivalent toward the Resistors. On the one hand they make good copy. (At one Spring Joint they had the only working time-sharing demo—on a carpet next to a phone booth.) On the other hand, they sometimes seem bratty and publicity-hungry, like many celebrities. (At another Spring Joint they dug up an IBM Songbook and serenaded the guys at the IBM pavilion, who had to act nice about it.) So they don't get written up in computer magazines so much anymore.

I first met the Resistors in 1970, and started hanging around with them for two reasons. First, they are perfectly delightful: enthusiastic in the way that most adults forego, and very witty. To them computer talk was not a thing apart, as it is for both outsiders and many professionals.

Secondly, and this was the self-seeking aspect, I noted that these kids were quite expert, and interested in giving me advice where computer professionals would not. They got interested in helping me with my (perhaps quixotic) Xanadu™ project (see flip side). This was enough to keep me visiting for a couple of years. Now, some people are too proud to ask children for information. This is dumb. Information is where you find it.

The last I heard, the Resistors were at work in a COBOL compiler for the PDP-11, hoping it would save the local high school from the disastrous (to them) purchase of an IBM 1130. (Since the school's intent was to teach business programming, they hoped that the availability of COBOL would encourage the school to buy the more powerful and less expensive PDP-11.)

*The COBOL didn't get finished and the 1130 got bought, and as individuals the Resistors have moved on to many destinies.*

The Resistors are few, but I think they are very important in principle, an existence proof. They show how silly and artificial is our edifice of pedagogy, with all its sequences and sterilizations, and how anybody can learn anything in the right atmosphere, stripped of its pomposities. The Resistors are not obsessed with computers: their love of computers is part of their love of everything, and everything is what computers are for.

### **R.E.S.I.S.T.O.R.S. ANECDOTES.**

Lauren, 14, was talking to another girl at the ACM 70 conference. A passerby heard her explaining the differences among the languages BASIC, FORTRAN, COBOL and TRAC. "How long have you been programming?" he asked in surprise. "Oh, almost a month," she said.

I was driving some Resistors around Princeton; they were yelling contradictory driving instructions. "I demand triple redundancy in the directions," I said. "Right up ahead you turn right right away," said a spokesman.

Since there was a lot of excess capacity, the Resistors got a free account on a national time-sharing system. Though they didn't have to pay, the system kept them informed on what they would have owed. In a year or so they ran up funny-money bills of several hundred thousand dollars.

Did they rate free subscriptions to computer magazines? I asked. Could they claim they really "make decisions affecting the purchase of computers?"

"Of course we do!" was the reply. "All together: shall we buy a computer?"

Resistors (in unison) "NO!"

Their original advisor, whom we shall call Gaston, is mischievous in his own right. It was meeting-time at Gaston's place on a bright Saturday, and I was on the lawn working on Xanadu

with Nat and Elliott when Gaston interrupted to say that an unwelcome salesman of burglar alarms was about to arrive. "Let's have a little fun with him," said Gaston. The kids were to be introduced as Gaston's children, I was an uncle. We took our stations.

The salesman may have realized he was walking into a trap from all the strangely beaming adolescents that stood in the living room. He got out his wares and started to demonstrate the burglar alarm, but it didn't go right. Peter, standing in front of the equipment with a demonically vacuous grin, had reversed a diode behind his back so that the alarm rang continuously *unless* you broke the light beam.

"Humpf," said Gaston, "you want to see a *real* security system?" We trooped into the kitchen, where Gaston kept a Teletype running.

ANY NEWS? typed Gaston.

CREAM YELLOW BUICK PULLED INTO DRIVEWAY, replied the Teletype. JERSEY LICENSE PLATE... (and the salesman's license number), and finally, OWNER OF RECORD NOT KNOWN. John was typing this from the other Teletype in the barn.

The salesman stared at the Teletype. He looked around at our cherubic smiling faces. He looked at the Teletype. "That's all right," said the salesman. "But now I'd like to show you a *real* security system..." And it was back to the old burglar alarm.

### **Where are the R.E.S.I.S.T.O.R.S. Now?**

*Well, grown up. And mostly in computers.*

*Jordan, with a BA from Dartmouth in Political Science, a Master's in International Affairs and an MBA from Columbia, originally went into international marketing and consulting on computer development policies in Latin America for IBM. He married Margo, and is now traveling the world in search of enlightenment.*

*John, with a BA and PhD in Computer Science from Yale, is one of the principal authors of PCIX (UNIX for the PC) and Javelin, winner of the PC World award for technical excellence*

*in software. He married his high school sweetheart.*

*Margo also got her BA in Computer Science from Yale, and has been one of the world's first personal computer managers, having been part of the corporate personal computer revolution since its inception. As noted above, she and Jordan are now fellow-travellers.*

*Nat, after obtaining his BS from Harvard and PhD from Princeton in Math, embarked on a highly successful career at MIT. This year, to everyone's surprise, he chucked it all in order to enter Stanford's medical school.*

*Lewis got his undergraduate degree from Princeton and his PhD from Yale (Linguistics and AI), got married, and at last report, was working at a research institution and living in Venice, California, "soaking up the California lifestyle."*

*Steve E. (BA Computer Science, Yale), married a lawyer, and is currently an applications specialist for Alliant, the fastest-growing mini-supercomputer company.*

*Jean (MIT, Engineering) worked for General Foods for a while making frozen french fries and instant coffee taste better; got married; and was last seen at Columbia getting her PhD in Chemical Engineering.*

*Peter got his PhD and worked for Amdahl for a while. He now makes an excellent California wine as a hobby.*

*Andy got his PhD, too. (Is this beginning to get boring?) Igor got his MD/PhD and is currently in his residency in Washington state. Steve L. programs for United Airlines.*

*Note that three got BAs in Computer Science from Yale? That's because the difference between a Bachelor of Arts and a Bachelor of Science at Yale is two additional Math courses. None of them was particularly inclined to take those courses. And they've never missed them.*

*Lauren claims to have left the computer field. She is the author's personal assistant.*



## YOU CAN HACK ANYTHING

A hacker, as the term is used among computer people, is someone who enthusiastically probes and explores and plays with anything, out of love and intellectual excitement, with strong initiative and persistence. (This was the sense of the meeting of the Hacker Conference, where the matter was discussed at length.)

You can hack anything. You can hack cars or stamp collecting. But to hack computers is the highest art (think the computer hackers), because COMPUTERS GO ON FOREVER.

You can Hack LISP or you can Hack BASIC. You can Hack the Mac or the Sinclair. You don't need the fancy equipment, it's the attitude.

Every beginning programmer writes calendar programs. But when a real hacker does a calendar program, it'll have options like Gregorian, French Revolutionary and Aztec. And printouts by week, month, year or pocket style.

## HACKERS VS. CRACKERS

Laymen don't realize how absurdly simple it is to break into on-line computers if they're not strongly defended. All you need is a computer with a modem—an over-the-phone communication box—and the right phone numbers and passwords. But this has been going on a long time (see "A Previously Unpublished Story," on the right).

(These are certainly not things that everybody does. But a few are understandably seduced by the Dark Side of the Force. So strong security measures are needed. And they don't come cheap.)

So we see it in every newspaper—COMPUTER BREAK-IN—and unfortunately they call the culprits "hackers."

The culprit for this terminological foul-up is Computer Security Guru Donn Parker, who regrettably used the term "hacker" for Cracker in contexts the press would pick up. Parker officially apologized for twisting the word to the Hackers.

Speaking of Crackers, here's one of the best yarns of the genre, I think. And it's true.

## A PREVIOUSLY UNPUBLISHED STORY

Not all kids who play with computers are quite as law-abiding as the R.E.S.I.S.T.O.R.S. (see p. 110). And the temptations are very strong.

One such youngster went on a high school field trip to a suburban Philadelphia police station, and saw a demonstration of the police remote information system.

The police who were demonstrating it, not being computer freaks, didn't realize how simple it was to observe the dial-in numbers, passwords and protocol.

When this lad got home, he merrily went to his computer terminal in the basement and proceeded to enter into Philadelphia's list of most-wanted criminals the names of all his teachers.

A few days later a man came to his house from the FBI. He was evidently not a regular operative but a technical type. He asked very nicely if the boy had a terminal. Then the FBI man asked very nicely if he had put in these names. The boy admitted, grinning, that he had. (Everyone in the school knew it had to be he.)

The FBI man asked him, very, very nicely not to do it again.

"Of course it didn't do any harm," says the culprit. "I had them down for crimes like 'intellectual murder'. What could happen to them for that?"

Does that make you feel better?  
PHILADELPHIANS AND CROOKS  
PLEASE NOTE:

This happened five or six years ago, and without a doubt the system is by now totally secure and impenetrable. Let's hope.

## BIBLIOGRAPHY

Guy L. Steele, Jr., Donald R. Woods, Raphael A. Finkel, Mark Crispin, Richard M. Stallman, Geoffrey S. Goodfellow, The Hacker's Dictionary. Harper and Row, 1983.

This is the dictionary for a new age, compiled mostly at AI and SAIL (MIT Artificial Intelligence Laboratory and Stanford Artificial Intelligence Laboratory).

## Etymology Of 'HACKING,' maybe

I am under the impression that "hacking" in the earlier part of the century meant the rustic outdoor activities of the rich—shooting pheasant and beating about in the gorse, as in the movie classic, "Rules of the Game." You wore a "hacking jacket" from Brooks Brothers or L.L. Bean.

This may have evolved into prep-school slang of the early fifties: "hacking around" or "hacking off" meant fooling around or wasting time.

Which in turn presumably migrated into computerdom as follows: "Are you working on the project, or just hacking around?"

"I'm hacking."

## Dreams Shared By Hackerdom And Crackerdom

The great backer's science-fiction book is the novella True Names by Vernor Vinge (Bluejay Books, New York, 1984) with an afterword, hey, by Marvin Minsky, which adds great weight to it.

The book takes place on two planes: in a fantasy, adventure-game world, populated by superheroes, and in the supposedly-real dark world of a near-future America.

The superheroes of this fantasy world are the mental projections of certain elite arch-hackers, who reach it deviously by the phone and other means. They construct this fantasy world with care, building on and through existing computer networks; but to get there requires the greatest computer programming skill, as well as the adventure-game skills of warlock and warrior.

These Great Hackers are at once a secret power elite and a force, more or less, for good. But they must hide their True Names, and maintain their spirit projections as seamless disguises, for else they become vulnerable to reprisals by the evil government. What happens? Ha. A great read.

In many ways, then, this book is the apotheosis of the hacker ethos that Levy tries to capture in the book Hackers. What have we got? Computer skills, Libertarian hatred for government, adventure gaming as paradoxically the real world—a summative allegory of the intellectual adventures of computerdom. Plus the book glamorizes crackerdom as well.

"IT'S A HACK"  
A Hack means either a crude solution to an unimportant problem—the software equivalent to holding a door open with a chair—or an interesting and chewy challenge to work on.

Hackers don't realize they're finite until they're 22. —David Leavitt

# COMPUTER FUN AND MISCHIEF

All kinds of dumb jokes and cartoons circulate among the public about computers. Then our friends regale us computerfolk with these jokes and cartoons, and because we don't laugh they say we have no sense of humor.

Oh we do, we do. But what we laugh at is rather more complicated, and relates to what we think of as the real structure of things.

Some of the best humor in the field is run in *Datamation*; an anthology called *Faith, Hope and Parity* reran a lot of their best pieces from the early sixties. Classic was the Kludge series, a romp describing various activities and products of the Kludge Komputer Korporation, whose foibles distilled many of the more idiotic things that have been done in the field. ("Kludge," pronounced "klooj," is a computerman's term for a ridiculous machine.) *Datamation's* humorous tradition has continued in a ponderous but extremely funny serial that ran in '72 called *Also Sprach von Neumann*, which in mellifluous and elliptical euphemisms described the author's adventures at the "airship foundry" and other confused companies that had him doing one preposterous thing with computers after another.

## COMPUTER PRANKS

*Pranks* are an important branch of humor in the field. Here are some that will give you a sense of it.

### ZAP THE 94

One of the meaner pranks was a program that ran on the old 7094. It could fit on one card (in binary), and put the computer in an inescapable loop. Unfortunately the usual "STOP" button was disabled by this program, so to stop the program one would eventually have to pull the big emergency button. This burnt out all the main registers.

### TIMES SQUARE LIGHTS

One of the weirder programs was the operator-waker-upper somebody wrote for the 7094. It was a big program, and what it did was DISPLAY ALPHABETICAL MESSAGES ON THE CONSOLE LIGHTS, sliding past like the news in

Times Square. You put in this program and followed it with the message; the computer's console board would light up and the news would go by. Since the lights usually blink in uninteresting patterns, this was very startling.

This program was extremely complex. Since the 94 displayed the contents of all main registers and trap, arithmetic and overflow lights, it was necessary to do very weird things in the program to turn these lights on and off at the right times.

### THE TIME-WASTER

In one company, for some reason, it was arranged that large and long-running programs had priority over short quick ones. Very well: someone wrote a counterattack program occupying several boxes of punch cards, to which you added the short program you really wanted to run, and a card specifying how long you wanted the first part of the program to grind before your real one actually started.

This would blink lights and spin tapes impressively and lengthen the run of your program to whatever you wanted.

### BOMBING THE TIME-SHARE

One of the classic bad-boy pranks is to bomb time-sharing systems—that is, foul them up and bring them to a halt. Many programmers have done this; one has told me it's a wonderful way to get rid of your aggressions.

Of course, it can damage other people's work (especially if disks are bombed); and it always gets the system programmers hopping mad, because it means you've defied their authority and maybe found a hole they don't know about. Here are a couple of examples.

#### I. THE PHANTOM STRIKES

The way this story is told, one of the time-sharing systems at MIT would go down at completely mysterious times, with all of core and disk being wiped out, and the lineprinter printing out THE PHANTOM STRIKES.

For a long time the guilty program could not be found. Finally it was discovered that the bomb was hidden in an

Twitting a program within its own premises is a jolly aspect of computer fun.

*The Chicago Museum of Science and Industry had an exceptionally stupid exhibit on nutrition. It consisted of a lot of wax fruit with placards and some interactive terminals ostensibly set up to counsel you on your diet.*

*The program, too, was ridiculous: I managed to convince it that I weighed 1999 pounds and was one foot tall. (Those were the limits the program allowed.) I did not wait around for its dietary recommendations.*

This game of three-dimensional tic-tac-toe was played with a program running on a minicomputer at the Spring Joint, 1969. CAUTION—ADULTS ONLY. While this example may offend some people, it vividly shows how programs may be toyed with—in this case, by the mischievous sign-on—to make them behave humorously.

```
NAME?MOTHER FUCKER
PLEASE PRINT YOUR NAME IN THREE PARTS SEPARATED BY BLANKS
EXAMPLE: MR JOHN JACKSON
NAME?M OTHER FUCKER
PLEASE PRINT YOUR NAME IN THREE PARTS SEPARATED BY BLANKS
EXAMPLE: MR JOHN JACKSON
NAME?MR MOTHER FUCKER
THANK YOU MOTHER YOU MAY TERMINATE THE FOLLOWING PRINTOUT AT ANY TIME BY PRESSING THE "BREAK" KEY
#
#
DO YOU WANT INSTRUCTIONS?NO
PLEASE TYPE "YES" OR "NO"?NO
DO YOU WANT TO MOVE FIRST?YES
YOUR MOVE?141
MACHINE MOVES TO 111
X... ..
... ..
... ..
... ..
YOUR MOVE?111
MACHINE MOVES TO 414
X... ..
... ..
... ..
... ..
YOUR MOVE?414
MACHINE MOVES TO 114
X... ..
... ..
... ..
... ..
YOUR MOVE?114
MACHINE MOVES TO 444
X..X... ..
... ..
... ..
... ..
YOUR MOVE?444
MACHINE MOVES TO 222
X... ..
... ..
... ..
... ..
YOUR MOVE?
333
MACHINE MOVES TO 323
X... ..
... ..
... ..
... ..
YOUR MOVE?
233
NICE TRY MOTHER—MACHINE MOVES TO 322
X... ..
... ..
... ..
... ..
YOUR MOVE?
433
NICE TRY MOTHER—MACHINE MOVES TO 133
X... ..
... ..
... ..
... ..
YOUR MOVE?
431
NICE TRY MOTHER—MACHINE MOVES TO 421
X... ..
... ..
... ..
... ..
YOUR MOVE?
432
CONGRATULATIONS MOTHER
YOU WIN AS FOLLOWS: 431 432 433 434
FINAL POSITION
X... ..
... ..
... ..
... ..
YOU CAN TRY SOME OTHER PROGRAMS OR PLAY ANOTHER GAME. WOULD YOU LIKE ANOTHER GAME MR. FUCKER?
```

Now It Can Be Told

When he was at the MIT AI lab, Danny Hillis surreptitiously made connections to the building elevator so that when he hit Control-E on his workstation, it summoned the Elevator.

*The biggest Yok right now is a little program called Talking Moose that sits in your Macintosh as a Desk Accessory.*

*At random intervals, a small picture of Bullwinkle Moose appears and makes tawdry wisecracks, out loud, in a squawky voice.*

*It's a wonderful prank on the jaded Macintosh user, plugging away at this or that project. He or she jumps out of his or her skin the first time The Moose Speaks.*

*The Moose is also a nice companion on long nights of work.*

*Video games, that immense fad of 1978-82, started among computer people. Spacewar (see Bibliography) and Moonlander (see p. DM 15) came first. (See p. DM 19.)*



old and venerable statistics program previously believed to be completely reliable. The reason the phantom didn't *always* strike was that the Bomb part queried the system clock and made a pseudo-random decision whether to bomb the system depending on the instantaneous setting of the clock. This is why it took so long to discover; the program usually bided its time and behaved properly.

Apparently this was the revenge of a disgruntled programmer, long since departed. Not only that, but his revenge was thorough: the Bomb part of the program was totally knitted into the rest of it, it was a very important program that had to be run a lot with different data, and no documentation existed, making it for practical purposes impossible to change.

The final solution, so the story goes, was this: whenever the rowdy program

had to be run, the rest of the machine was cleared or put on protect, so it ran and had its fits in majestic solitude.

## 2. RHBOMB

The time-share at the Labs, never mind which Labs, kept going down. Mischief was suspected. Mischief was verified: a program called RHBOMB, submitted by a certain programmer with the initials R.H., was responsible, and turned out always to be present when the terminals printed TSS HAS GONE DOWN. It was verified by the systems people that the program called RHBOMB was in fact a Bomb program, with no other purpose than to take down the time-sharing system.

R.H. was spoken to sternly and it did not happen again.

However, some months later a snoopy systems programmer noted that a file called RHBOMB had been stored on disk. Rather than have R.H. scalped prematurely, he thought he would check the contents.

He sat down at the terminal and typed in the command, PRINT RHBOMB. But before he could see its contents, the terminal typed instead:

TSS HAS GONE DOWN

But this was incredible! A program so virulent that if you just tried to *read its contents*, without running it, it still bombed the system! The systems man rushed from the room to see what had gone wrong.

He did so prematurely. The contents of the new file RHBOMB were simply

TSS HAS GONE DOWN

followed by thousands of null codes, which were silently being fed to the Teletype, 10 per second, preventing it from signalling that it was ready for the next thing.

## GAMES

Games with computer programs are universally enjoyed in the computer community. Wherever there are graphic displays there is usually a version of the game Spacewar. (See Steward Brand's Spacewar piece in *Rolling Stone*, mentioned elsewhere.) Spacewar, like many other computer-based games, is played between *people*, using the computer as an animated board which can work out the results of complex rules.

Some installations have computer games you can play *against*; you are

effectively "playing against the house," trying to outfox a program. This is rarely easy. A variety of techniques, hidden from you, can be used.

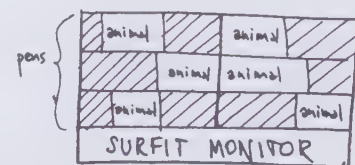
When "a computer" plays a game, actually somebody's program is carrying out a set of rules that the programmer has laid out in advance. The program has a natural edge: it can check a much longer series of possibilities in looking for the best move (according to the criteria in the program).

There is a more complicated approach: the computer can be programmed to *test for the best strategy* in a game. This is much more complicated, and is ordinarily considered an example of "artificial intelligence" (see "The God-Builders," elsewhere in this book).

## SURVIVAL OF THE FITTEST

One of the stranger projects of the sixties was a game played by the most illustrious programmers at a well-known place of research; the place cannot be named here, nor the true name of the project, because funds were obtained through sober channels, and those who approved were unaware of the true nature of the project, a game we shall call SURFIT ("SURvival of the FITtest"). Every day after lunch the guys would solemnly deliver their programs and see who won. It was a sort of analogy to biological evolution. The programs would attack each other, and the survivors would multiply until only one was left.

It worked like this. Core memory was divided up into "pens," one for each programmer, plus an area for the monitor.



Each program, or "animal," could be loaded anywhere in its pen. The other programs knew the size of the pen but not where the animal was in it. Under supervision of the special monitor, the animals could by turns *bite* into the other pens, meaning that the contents of core at several consecutive locations in the other pen was brought back, and changed to zero in its original pen.

When the word got around that this nude was in a public file on the time-sharing system, my office-mates scrambled to get printouts of her. The cleverest, though, had a *deck punched*. As he predicted, she was thrown off by the systems people within an hour or so—leaving the other guys with their printouts, but he had the deck. Now he can put her *back* in the computer any time, but they can't.

**A** lot of people are putting their private materials—correspondence and diaries—on office computers, on the assumption (usually correct) that no one will read them.

But it always amuses people to violate others' privacy, and you better watch yourself.

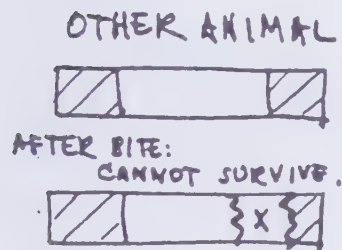
Apparently a number of employees at a weighty New York newspaper got a lot of laughs out of reading the love letters that went back and forth between the keyscopes of two employees. This is mischievous, but it is not funny; it is a serious issue of our time. (See "Hot Screens," p. 160.)

Your animal could then “digest”—that is, analyze—the contents bitten. Then the *other* animal got his turn. If he was still alive—that is, if the program could still function—it could stay in play; otherwise the animal who had bitten it to death could multiply itself into the other pen.

The winner was the guy whose animal occupied all pens at the end of the run. If he won several times in a row he had to reveal how his program worked.

As the game went on, more and more sophistication was poured into the analytic routines, whereby the animal analyzed the program that was its victim; so the programmer could attack better next time. The programs got bigger and bigger.

Finally the game came to a close. A creature emerged who could not be beaten. The programmer had reinvented the germ. His winning creature was *all teeth*, with no diagnostic routines; and the first thing it did was multiply itself through the entirety of its own pen, assuring that no matter where it might just have been bitten, it would survive.



#### CONWAY'S GAME OF LIFE

A Grand Fad among computerfolk in the last couple of years has been the game of “Life,” invented by John Horton Conway.

The rules appeared in the *Scientific American* in October 1970, in Martin Gardner's games column, and the whole country went wild. Gardner was swamped with results (many published in Feb. '71); after a couple more issues Gardner washed his hands of it, and it goes on in its own magazine.

The game is a strange model of evolution, natural selection, quantum mechanics or pretty much whatever else you want to see in it. Part of its initial fascina-

tion was that Conway didn't know its long-term outcomes, and held a contest (eventually won by a group from MIT).

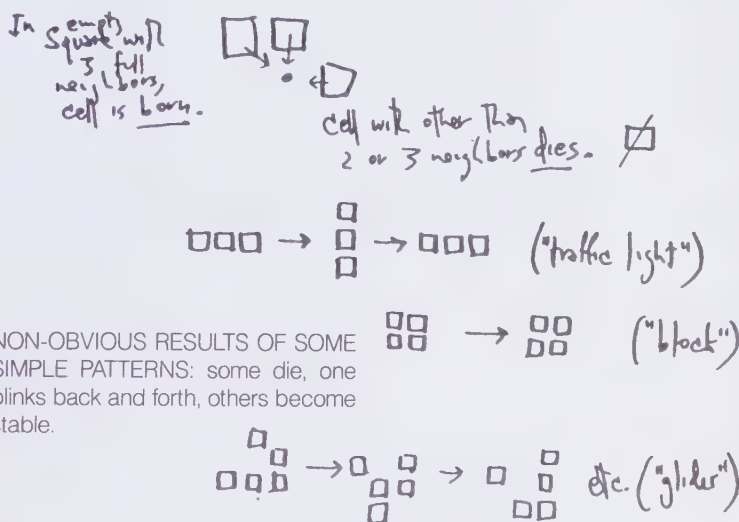
The rules are deceptively simple: suppose you have a big checkerboard. Each cell has eight neighbors: the cells next to it up, down and diagonally.

Time flows in the game by “generations.” The pattern on the board in each generation determines the pattern on the board in the next generation. The *game* part simply consists of trying out new patterns and seeing what things result in the generations after it. Each cell is either OCCUPIED or EMPTY. A cell becomes occupied (or “is born”) if exactly three of its neighbors were full in the previous generation. A cell *stays* occupied if either

two or three of its neighbors were occupied in the previous generation. All other cells become empty (“die”).

These rules have the following general effect: patterns you make will change, repeat, grow, disappear in wild combinations. Some patterns move across the screen in succeeding generations (“gliders”). Other patterns pulsate strangely and *eject* gliders repetitively (glider guns). Some patterns crash together in ways that produce moving glider guns. Weird.

While the Game of Life, as you can see from the rules, has nothing to do with computers intrinsically, obviously computers are the only way to try out complex patterns in a reasonable length of time.



NON-OBVIOUS RESULTS OF SOME SIMPLE PATTERNS: some die, one blinks back and forth, others become stable.

## ZibS.....

Exhibit floors at conferences are Disneylands to us! So much to see! There's a Cray! There's a Pixar! What the hell is that?

Only after decades of conference-going did I come to understand why the fun and excitement of the exhibits were so intense.

It's not just the toys you get to see and dream about. It's the challenge.

#### FUN WITH THE BOOTH PEOPLE .....

1. See if you can understand it faster than the guy can explain it. Ask smart questions that he's never heard before.

2. If you get a dodo, try nevertheless to find out anything you can.

Unfortunately, as this game is usually played by aggressive young men, it can involve a lot of putdowns, leaving the booth people feeling bad. Watch it. You may be a booth person yourself some day.

#### BIBLIOGRAPHY

Donald D. Spencer, *Game Playing with Computers*. (Sparta/Hayden, \$13.)

This includes flowcharts, programs, and what-have-you for some 25 games, and suggestions for more.

Stewart Brand's marvelous *Spacewar* piece is highly recommended. In *Stewart Brand*, Two Cybernetic Frontiers.

Robert C. Gammill, “An Examination of Tic-Tac-Toe-like Games”. *Proc. NCC* 74, 349-355. Examines structure of simple games (esp. 3D tic-tac-toe or QUBIC) where forced wins are possible; and program structures to play them.

“The Game of Life,” *Time*, 21 Jan '74, pp. 66-67.

#### NOW IT CAN ALSO BE TOLD

Yes, A.K. Dewdney in the *Scientific American* correctly guessed that the program here called SURFIT was the infamous DARWIN at Bell Laboratories.

Dewdney has continued the DARWIN tradition by hosting COREWARS, the same thing on a public-access basis, in which readers can submit monsters of their own devising—in a simplified computer language called Redcode—to battle it out in chambers unseen.

This is just the recreational side of genetic and population simulation. Serious evolutionary research has been using these techniques for years (see p. 151).



# HOW SOME COMPUTER COMPANIES ARE FINANCED- A PERSPECTIVE

Those of us who were around will never forget the Days of Madness (1968-9). Computer stocks were booming, and their buyers didn't know what it was about; but everywhere there were financial people trying to back new computer companies, and everywhere the smart computer people who'd missed out on Getting Theirs were looking for a deal.

*Datamation* for November 1969 was an inch thick, there were that many ads for computers and accessories.

At the Fall Joint Computer Conference that year in Las Vegas, I had to cover the highlights of the exhibits in a hurry, and it took me all afternoon, much of it practically at a trot. Then, after closing time, I found out there had been a *whole other building*.

It is important to look at how a lot of these companies were backed, the better to understand how irrationality bloomed in the system, and made the collapse of the speculative stocks in 1970 quite inevitable.

A number of companies were started at the initiative of people who knew what they were doing and had a clear idea, a new technique or a good marketing slant. These were in the minority, I fear.

More common were companies started at the initiative of somebody who wanted to start "another X"—another minicomputer company, another terminal company, expecting the product somehow to be satisfactory when thrown together by hired help. Perhaps these people saw computer companies as some-

thing like gold mines, putting out a common product with interchangeable commodity value.

The deal, as some of these Wall St. hangers-on would explain it, was most intriguing. Their idea was to create a computer company on low capital, "bring it public" (get clearance from the SEC to sell stock publicly), and then make a killing as the sheep bought it and the price went up. Then, if you could get a "track record" based on a few fast sales, the increasing price of your stock (these are the days of madness, remember) makes it possible to buy up other companies and become a conglomerate.

It was very difficult to talk to these people, particularly if you were trying to get support for a legitimate enterprise

built around unusual ideas. (Everybody wants to be *second*.) And what's worse, they tended to have that most reprehensible quality: they *wouldn't listen*. Did they want to hear what your idea actually *was*? "I'll get my technical people to evaluate it"—and they send over Joe who once took COBOL. I finally figured out that such people are impossible to talk to if you're sincere—it's a quality they find unfamiliar and threatening. I don't think there's any way a person with a genuine idea can communicate with such Wheeler-Dealers; they just fix you with a piercing glance and say "Yeah, but are we talking about hardware or software?" (the two words they know in the field).

The joker is that if you missed out on all this you were much better off. Anyone

To get venture capital, you need a Concept, but it has to be expressible in two or three words, without exact meaning that would get in the way of enthusiasm.

What people have wanted to back, and what things have been significant, are totally different.

It is tomorrow's ideas that are important, but backers want to talk about Yesterday's Hard Facts.

## MARKET RESEARCH

'And the wheel,' said the Captain. 'What about this wheel thingy? It sounds a terribly interesting project.'

'Ah,' said the marketing girl, 'Well, we're having a little difficulty there.' 'Difficulty?' exclaimed Ford. 'Difficulty? What do you mean, difficulty? It's the single simplest machine in the entire universe!'

The marketing girl soured him with a look. 'Alright, Mr Wiseguy,' she said, 'you're so clever, you tell us what colour it should be.'

—Douglas Adams, *The Restaurant at the End of the Universe*, Pan Books, 1980, p. 172.

The Business Plan is a structured fiction very like the fairy tale, from *Once Upon A Time to Happily Ever After*.

But it must be presented more solemnly than a Requiem Mass. Anyone who giggles is disqualified.

"To get backing, it doesn't matter if your program works, as long as you give great demo." (See "Negentropy," p. DM 66.)

According to one legend, a certain entrepreneur hired an office and a collection of actors to look busy when he brought in investors. He then got his financing, went to Asia and actually built a company, which was very successful.

"Product planners" in the computer field are a joke: the not-very-knowing, who do not love computers, trying to guess what will be wanted by somebody else, at a time in history which will probably be quite different.

EVERYTHING GOOD IN THE COMPUTER FIELD HAS COME ABOUT BECAUSE SOMEONE SAID "I WANT THIS, AND I WILL BUILD IT FOR MYSELF."

The horrors and fiascoes came from designs dictated by management or market researchers.

with a genuine idea is being set up for two fleecings: the first big one, when they tell you your ideas, skills and long-term indenture are worth 2½ percent (if you're lucky) compared to their immense contributions of "business knowhow," and the second, when you go public and the underwriter gets vast rakeoffs for *his* incomparable services. What is most likely to get lost in all this is any original or structured contribution to the world that the company was intended, in your mind, to achieve.

In part this is because anyone with technical knowledge is apparently labeled Silly Technician in the financial community, or Impossible Dreamer; it is entrenched doctrine among many people there that the man with the original idea cannot be allowed to control the direction of the resulting company. In one case known to me, a man had a beautiful invention (not electronic) that could have deeply improved American industry. It was inexpensive, simple to manufacture, profoundly effective. He made his deal and the company was started, under his direction. But it was a trick. When the second installment of financing came due, the backers called for a new deal, and he was skewered. Result: no sales, no effect on the world, no nothing to speak of.

This is all the sadder because the companies that achieve important things in

this field, as far as I can see, are those with a unifying idea, carried out unstintingly by the man or men who believe in it. I think of Olsen's Digital Equipment Corporation, Data General, Evans and Sutherland Computer Corporation, Vector General. This is not to say that a good idea succeeds without good management or good breaks: for instance, Viatron, a firm which was the darling of the computer high-flying stocks, had a perfectly sound idea, if not a deep one: to produce a video terminal that could be sold for as little as \$100 a month. But they got overextended, and had manufacturing troubles, and that was that. (You can now get a video terminal for \$49 a month, the Hazeltine.) Of course, a lot of ideas are hard to evaluate. A man named Ovshinsky, for instance, named a whole new branch of electronics after himself ("ovonics"), and claimed it would make integrated circuits cheaper or better than anybody else's. Scoff, scoff. Now Ovshinsky has had the last laugh: what he discovered some now call "amorphous semiconductor technology," and his circuits are being used by manufacturers of computer equipment. Another example is one Frank Marchuk, whose "laser computer" was announced several years ago but hasn't been seen yet. Many computer people are understandably skeptical.

Market research is like finding out about sex from questionnaires.  
The real question is, what is it like to those who care?  
The nurturance of products should be by those who care most about the product.  
King Solomon knew that.

*"Any curve is total crap. If you believe it you're being fooled."*

—Steve Jobs,  
quoted in Moritz,  
The Little Kingdom,  
p. 84 (Morrow, 1984)

**"Bad software is not a design problem. Bad software is created by masturbatory programmers."**  
—A publishing executive

**T**he difference between good software design and bad software design is, to quote Twain, the difference between the lightning and the lightning bug. The wheel is a case in point: making it hexagonal is a definite downgrade, and the color is largely irrelevant. Yet all kinds of marketing and MBA twonks think they can monkey with designs for business reasons. This is rarely possible. The good design is a peak; in all directions it's downhill. And the good software design is simple, clear, powerful, elegant, self-revealing, clean, and works well when people use it. It is very rare that you can change part of a good software design.



This is still a field where individuals can have a profound influence. But the wrong way to try it is through conventional corporate financing. Get your own computer, do it in a garret, and *then* talk about ways of getting it out to the world.

#### BIBLIOGRAPHY

John Brooks, *The Go-Go Years*. Weybright & Talley. \$10.

Well, the *Go-Go Years* keep repeating themselves. In years where computers are hot, everybody wants to back the Second Whatever.

But the psychology, the *Scorpion Dance* with backers, remains the same. One friend of mine questions whether venture capitalists really want to make money; "It's making the smart technical guys feel two inches tall—that's what venture capital is about," he says. "They only invest now and then to keep the victims coming."

And still the hopefuls build, in garage (hardware) and cottage (software).

Remember, **MANY MAKE A GOOD LIVING SELLING SOFTWARE WITHOUT BACKERS**. Make sure you know what you're giving up, if you *do* find backing.

The Editor, *TIME*  
Rockefeller Center  
New York, NY 10020

Sir:

Even if you missed out on all the skyrocketing new stock issues, such as Educational Computer Corp., Frigitronics, Nathan's Famous Inc., and Minnie Pearl's Chicken System (June 14), there's still hope. You can buy stock in my new company. The price has already tripled (from 5 cents a share to 15 cents) and nothing stands in the way of further rises: the company has no assets, no product, no employees, and no plans. Thus it is a much safer investment than other hot issues—the company can't possibly lose money. The one thing we have going for us is a sure-fire name.

Sincerely yours,  
Charles S. Harris, President  
Nathan's Chickentronic Computer Associates

"Good ideas are a dime a dozen."  
No, good ideas are rare. But people who can't tell the difference are a dime a dozen.

## THE QUARTERLY BALANCE SHEET

Some people think things are going to hell because today's management is motivated only short-term. The important thing, in this reasoning, is **THE CURRENT QUARTER'S BOTTOM LINE**, since if that's good the managers get to play another round and if it isn't they're gone anyway. In this atmosphere, long-term rationality is virtually impossible. (De Tocqueville commented on this American trait two hundred years ago.)

This outlook is very much present in the software field. Software companies want products they can get out the door in three months, six months at the most. This makes the creation of large, robust and innovative programs exceedingly difficult.



Yes, it's real.  
Life imitates art  
on Route 46, N.J.

A huge new group has come into computerdom: the ignorant carpetbagger entrepreneurs from all over, full of this month's clichés about what's needed in software. And they all want something they can market in six months. It would be interesting if people went into agriculture with that attitude.

# HOW COMPUTERS AND SOFTWARE ARE BOUGHT AND SOLD

The public still thinks there is something rational about computers. Such a belief can be seriously questioned. Computers are bought and sold with total craziness. Perhaps half of all computers are bought rationally, the other half arrive through misunderstanding, caprice or politicking. (But these are very hard lines to draw.)

## THE SELLING

Computers are not purchased, or offered for sale, on much of a rational basis.

In the old days it was company to company; with internal politics, intricate collaborative and evasive maneuvers, corporate knife-in-the-back, and so on.

This all still goes on. Bejewelled salesmen with impeccable clothing and cunning heartiness still go about on their corporate ministrations and vie on the Machiavellian plane. The salesman works tricky political waters.

The computer users in a company generally know what they need, although

there may be different factions. But real decisions are often still made at another level, where the chairman of the board plays golf with an IBM executive, or a committee of beginners is impanelled to evaluate incomprehensible presentations.

The same confusions persist in these days of personal computing. But now they have new outlets.

## NEW DAYS: STORES AND INDIVIDUALS

Now there is a Consumer Market. But the same confusion and mystification persist, the same inane logic of slogans instead of understanding.

I find it painful to visit computer stores and hear a rank beginner—sometimes a businessman, sometimes a secretary delegated to take care of the task during lunch hour—ask, “What’s the best word processor?”—and get stuck with something they will find hard indeed to use.

It’s like cameras: people buy “what I’ve heard is the best,” rather than some-

thing simple and usable (at least till recently).

Computer stores went through many phases since the first one in 1975; now they’re more or less like other stores. Many stores are suave, Concerned for Your Problems, and, a great benefit arising from this, they support their stuff much better.

But the problem of KNOWING WHAT TO BUY is still the heart of the matter. And how you make that decision—whether as an individual or as a company—has more to do with your soul than scientific erudition. (See p. 31.)

## HOW COMPUTER STUFF IS BOUGHT AND SOLD

For the most part, big computers have always been rented or leased, rather than bought outright. There are various reasons for this. From the customer’s point of view, it makes the whole thing tax-deductible without amortization prob-

lems, and means that it’s possible to change part of the package—the model of computer or the accessories—more easily. And big amounts of money don’t have to be shelled out at once.

From the manufacturer’s point of view (and of course we are speaking mostly of IBM), it is advantageous to work the leasing game for several reasons. Cash inflow is steady, the manufacturer is in continuous communication with the customer, and has his ear for changes and improvements costing more. Competitors are at a disadvantage because the immense capital base needed to get into the selling-and-leasing game makes competition impossible.

Basically, leasing really may be thought of as having two parts: the sale of the computer, and banking a loan on it; essentially the lease payments are installment payments, and the real profits come after the customer has effectively paid the real purchase price and is still forking over.

EVERYTHING IS  
DEEPLY INTERTWINGLED®

**“Visual readoffs are all centralized, miniaturized, and set on schematic panels now. And the data compiled is all automatically computed. And there’s an automatic typewritten panel on it.”**

—Spencer Tracy’s presentation in *The Desk Set*, 1950s

It is said that the provincial government of Ontario promised to provide the furniture for the new University of Waterloo.

So, according to the legend, they ordered a raft of mainframes and called it all furniture.

The provincial government of Ontario gamely coughed up millions and millions of dollars.

University of Waterloo now has thousands of computer science majors, and is the principal university for computer science in Canada.



A myth about how to buy computers (the sort of thing Arthur D. Little tells companies to do).

YOU WRITE A DOCUMENT STATING YOUR NEEDS, THEN PUT OUT FOR BIDS.

The only problem: How do you know what your needs are?

If you are a beginning individual or company, you have only the vaguest idea of what’s available and how to map it to what you want.

A “consultant” or delegated employee will tell you his theory of your needs.

And, of course, so will every sales organization.

BUT WHAT IT WILL BE LIKE AFTER YOU GET YOUR COMPUTER SYSTEM, WHAT WILL “CATCH ON”—WHAT WILL WORK IN YOUR LIFE, OR YOUR COMPANY—REMAINS TO BE SEEN.

BUSINESSLAND!  
What A Concept.



Many firms other than IBM prefer to sell their computers outright. Minicomputers are almost always sold rather than rented. However, for those who believe in renting or leasing, the so-called "leasing firms" have appeared, effectively performing a banking function. They buy the computer, you rent or lease it from them, and they make the money you would've saved if you'd bought.

IBM, now required to sell its computers as well as lease them, keeps making changes in its systems which cynics think are done partly to scare companies away from buying, since if you've bought the computer you can't catch up. (Large computers bought from companies that like to sell them, such as DEC and CDC, do not seem to have this problem.)

## ANNOUNCEMENTS

### *Now Called VAPORWARE*

An eccentric aspect of the computer field is the Announcement, the statement by a company (or even individual) that he is planning to make or sell a certain computer or program. Some very odd things happen with announcements in this field. (None of this is unique to computerdom, but it goes to unusual extremes here.)

Under our system it is permissible for any person or firm to announce that he will make or sell any particular thing, and even if he's lying through his teeth, it's not ordinarily considered fraud unless money changes hands. Talk is cheap. Thus it is common practice in American industry for people to say that they will soon be selling hundred-mile-an-hour automobiles, tapioca-powered rocketships, antigravity belts.

Okay. In the computer world the same thing happens. The strategy depends on the announcer's market position. The little guys are often bluffing wistfully, hoping someone will get interested enough to put up the money to finish the project, or the like; the big companies are often "testing the water," looking to see whether there are potential customers for what they haven't even attempted to develop. Announcements by big companies also have strategic value: if they announce something a smaller guy has already announced, they may cut him off at the pass, even though they have no intention of delivering. That's just one example. The analysis of IBM's

announcements is a parlor game in the field. It has been alleged, for instance, that IBM announced its 360 computer long before it was ready to cut off incursions on its customers by other firms; Control Data, in a recent suit, alleged that the Model 90 numbers of the 360 were announced, and then developed, simply to destroy Control Data and its own big fast machines. These are just examples.

In other words, *caveat auditor*.

*Datamation* ran several good articles on buying computer stuff in its September 15, 1970 issue.

"Software Buying" by Howard Bromberg (35-40) and "Contract Caveats" by Robert P. Bigelow (41-44) are very helpful warnings about not getting burned.

Another, "Project Management Games," by Werner W. Leutert (24-34) is an absolutely brilliant, blood-curdling strategic analysis of the ploys and dangers involved in buying and selling very expensive things, such as computers and software. ANYONE INVOLVED IN COMPUTER MANAGEMENT SHOULD READ THIS MACHIAVELLIAN PIECE WITH THE GREATEST CARE. Anyone interested in the theory of showdown and negotiation can read it with a different slant.

*Q: What's the difference between a used-car salesman and a computer salesman?*

*A: A used-car salesman knows when he's lying.*

—Anon.

## THE THREE AGES OF A PRODUCT

**Alpha test is when you wouldn't even show it to your Mom.**

**Beta test is where you strap your Mom down and say, Here, Mom.**

**Gamma test is the rest of the life of the product.**

—Mike Butler

# COMPUTER COMPANIES

*There are tens of thousands, maybe hundreds of thousands, of computer companies. Anyone can start one; all it takes is a letterhead.*

*For the purposes of this book there are only a few companies to discuss.*

## THE SEVEN DWARVES AND THEIR FRIENDS

The computer companies are often called "Snow White and the Seven Dwarves," even though the seven keep changing. Here are some main ones beside IBM. I hope I haven't left anyone out.

Sperry Rand Univac  
Honeywell  
Burroughs  
Control Data Corporation (CDC)  
National Cash Register (NCR)  
Digital Equipment Corporation (DEC)  
Xerox Data Systems (XDS: formerly Scientific Data Systems (SDS))  
Hewlett-Packard (HP)  
Data General  
Interdata, Inc.  
Varian Data Machines  
Lockheed

### Requiescant in Pace:

General Electric (sold out to Honeywell)  
RCA (sold out to Univac)  
Philco  
General Foods  
and others beyond recollection

*Companies that market to corporations come and go without doing anything interesting technically. The innovations now are in the "workstation" computer companies, such as Sun Microsystems and Silicon Graphics (see p. DM 113).*

## BIBLIOGRAPHY

Kate Fishman, *The Computer Establishment*.

NO COMPUTER MANAGEMENT—NOT IBM, NOT AT&T, NOT COMMODORE OR DEC OR ANYBODY—COMPREHENDS THE PROBLEMS OF COMPATIBILITY, UNITY AND GOOD DESIGN.

Everybody is still just jamming ill-matched product lines together and mistaking software for "functionality."

Computers and software don't mix'n'match very well. Whoever comes out with really unified stuff is going to make a bundle.

## WHAT CAN WE GET OUT THE DOOR IN A HURRY TO GRAB MARKET SHARE?

*The dumb question that drives company after company over the cliffs.*

Good things for hackers—cleanly-programmable computers, useful software components:

DEC'S GREAT ACHIEVEMENTS, wonderful for their day, now part of the personal computer world

1. THE COMMERCIAL MINICOMPUTER (originally with 18 bits, about 16K, and paper tape)—

a tradition now carried on by today's desktop computers and "microcomputers" (yecch).

2. THE FREE-COUPING, YOU-INTERFACE-IT COMPONENT COMPUTER—

with mix-and-match boards (except they were bigger then) and the chance even to monkey around at the bit level—  
a tradition carried on by the Apple II, PC, and Mac II.

3. OPERATING SYSTEMS ON THE FILE-TRANSFER MODEL—

a tradition carried on by AppleDOS, CP/M and MS-DOS, and (by far the best) UNIX.

4. GOOD TIME-SHARING.

(A tradition arguably available through The Source and CompuServe.)

The Computer Fan's Computer Company

# DEC - THE PDPEOPLE

The computer companies are often referred to in the field as "Snow White and the Seven Dwarfs"—a phrase that stays the same even as the lesser ones (like RCA and General Electric) get out of the business one by one. The phrase suggests that they're all alike. To an extent; but there is one company sufficiently different, and important enough both in its history and its continuing eminence, to require exposition here. This is Digital Equipment Corporation, usually pronounced "Deck," the people who first brought out the minicomputer and continue to make fine stuff for people who know what they are doing.

Other computer companies have mimicked IBM. They have built big computers and tried to sell them to big corporations for their business data processing, or big "scientific" machines and tried to sell them to scientists.

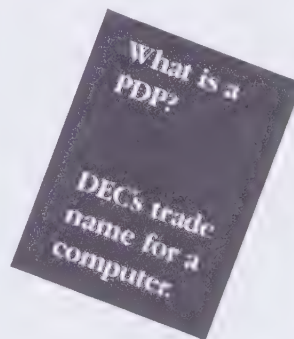
DEC went about it differently, always designing for the people who knew what they were doing, and always going to great lengths to tell you *exactly* what their equipment did.

First they made circuits for people who wanted to tie digital equipment together. Then, since they had the circuits anyway, they manufactured a computer

(the PDP-1). Then more computers, increasing the line slowly, but always telling potential users as much as they could possibly want to know.

The same for its manuals. People who wrote for information from Digital would often get, not a summary sheet referring you to a local sales office, but a complete manual (say, for the PDP-8), including chapters on programming, how to build interfaces to it, and the *exact timing and distribution of the main internal pulses*. The effect of this was that sophisticated users—especially in universities and research establishments—started building their own. Their own interfaces, their own modifications to DEC computers, their own original systems around DEC computers.

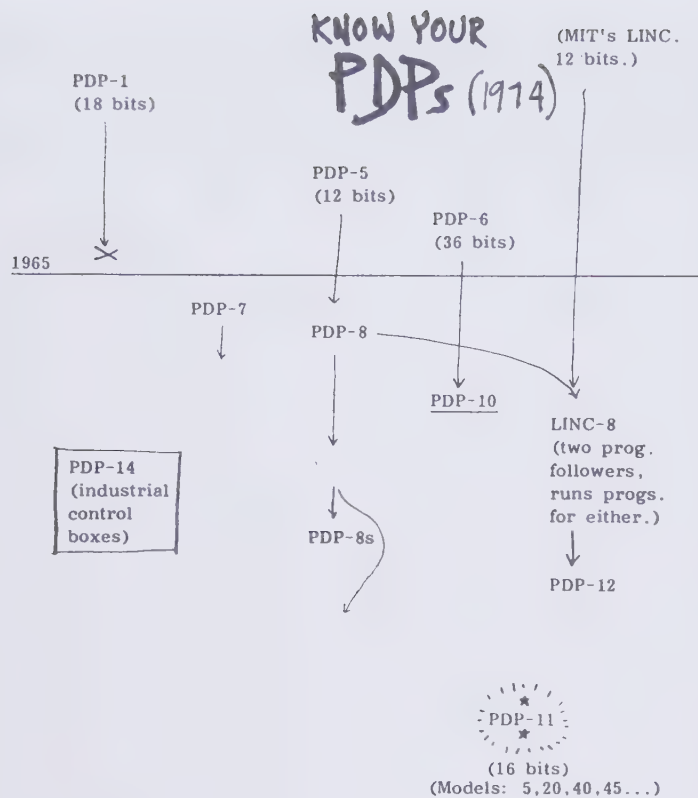
This policy has made for slow but steady growth. In effect, Digital built a national customer base among the most sophisticated clients. The kids who as undergraduates and hangers-on built interfaces and kludgy arrangements, now as project heads build big fancy systems around DEC equipment. The places that know computers usually have a variety of DEC equipment around, usually drastically modified. ➡



### NOW-CHASTENED COMPANIES THAT THOUGHT THEY HAD PERSONAL COMPUTING KNOCKED

- Bally
- Coleco
- Texas Instruments
- Atari
- Commodore
- IBM





Because of the great success of its small computers, especially the PDP-8, even many computer people think they only make small computers. In fact their big computer, the PDP-10, is one of the most successful time-sharing computers. An example of its general esteem in the field: it is the host computer of ARPANET, the national computer network among scientific installations funded by the Department of Defense; basically this means ARPANET is a network of PDP-10s.

DEC's computers have always been designed by programmers, for programmers. This made for considerable suspense when the PDP-11 did not appear, even though the higher numbers did, and the grapevine had it that the 11 would be a sixteen-bit machine. It proved to be well worth waiting for (see p. 90), and has since become the standard sophisticated 16-bit machine in the industry.

An area DEC has emphasized from the first has been computer display (discussed at length on the flip side). (And the University of Utah, currently the mother church of computer display, runs its graphic systems from PDP-10s.)

In this plucky, homespun company, where even president Olsen is known by his first name (Ken), it is understandable that marketing pizzazz takes a back seat. This apparently was the view of a group of rebels, led by vice president Ed deCastro, who broke off in the late sixties to start a new computer company around a 16-bit computer design called the Nova—rumored to have been a rejected design for the PDP-11. The company they started, Data General, has not been afraid to use the hard sell, and between their hard sell and sound machine line they've seriously challenged the parent company.

But Digital marches on, the Computer Fan's computer company. If IBM is computerdom's Kodak, whose overpriced but quite reliable goods have various drawbacks, DEC is Nikon, with a mix-and-match assortment of what the hotshots want. That's pluralism for you.

Originally DEC sold clockwork: the digital circuits, then the whole computers, then the computers on boards to be put inside equipment. And the greatest thing was the completeness and accuracy of the manuals.

By selling reliable, well-designed small computers to fit into other people's

equipment, Digital inveigled its digitalia into the laboratories and factories of the world, calibration and graphics machines, communication and networking processors.

Meanwhile, DEC gave their brilliant hacker-designers free reign in building the machines and software they wanted for themselves, and their customers free reign in configuring what they wanted down to the bit level—rather than having the design dictated (or selected) by churls out of the marketing department. How unlike IBM! (And many other companies.)

And so Digital made the best equipment in the world: notably the PDP-10 time-sharing computer and the PDP-11 small machine. And then the VAX, the ultimate bells-and-whistles computer (see p. 90).

But gradually, with growth, DEC drifted into systems: whole laboratory packages, information retrieval setups, and software packages for specific lines of work. This was not wrong. But they also drifted into marketing-department type direction of the product line, which was.

The company has grown great and fat and (some think) lost its roots. (DEC's president-for-life Ken Olsen gets a bigger salary than IBM's succession of helmsmen.)

Here's how things stood in 1975:

Despite its steadfastly insipid marketing, DEC has consolidated its position at the center of the small-computer maelstrom, and the PDP-11 has been consolidated as the small and medium-sized computer of choice among sophisticates. (The PDP-11 is also attracting considerable interest as a network computer.) In one curious instance, First National City Bank of New York is creating a network of 11/45s.

## COMMUNICATIONS AND UNIFICATION

The networking has been the key to DEC's immense growth since then.

DEC's immense present position is based on offering complete packages—with corporate communications among remote machines—all packaged and ready to use. (IBM, making its systems too complex to prevent others' interconnection, has done itself in in this department.)

I'm not getting any favors from DEC, I'm just saying about them what people ought to know.

However, I do have grateful recollections of the warmth and courtesy with which people from Digital Equipment Corporation have taken pains to explain things to me, hour after hour, conference after conference.

In the early sixties they had one man in one small office to service and sell all of New Jersey and New York City. But that one guy, Dave Denniston, spent considerable time responding to my questions and requests over a period of a couple of years, and in the nicest possible way, even though there was no way I could buy anything. You don't forget treatment like that.

## THE BAD NEWS

Though DEC invented the little computer, they have abandoned that market; the generic "PDP" is now largely a memory; component computers and desktop machines suited for lab and office are made by everybody's uncle in the Far East, out of cheaper chips, made elsewhere. Most ignominiously, DEC's stab at the personal-computer market—the Rainbow, a Z-80 in a very expensive package—was a dismal flop.

Though the VAX has been admired and successful, its doom awaits. A baroque machine (see p. 90), it is being passed by cheaper powerful machines (e.g., those with 68000s) and in the long term by the sleek eel-like RISC machines (see p. 91).

## BELL, THE BELLWETHER

Time was, DEC was a center of gravity for geniuses. The tilting old factory in Maynard creaked under midnight maniacs playing their adventures of the mind.

For such people DEC has been permissive but not lucrative. Like moths, many of the smartest have gone to the brighter lights—more glamorous places, some offering equity. None can fail to note the departure of Gordon Bell, designer of the 3, 7, 9, 5, 8, 11 and VAX. Without Gordon Bell, what can DEC have left?

## HOW STACKS THE DEC?

The company is overstuffed, cut off from its roots and (some think) its virtue.

DEC's strong points were frankness, clarity, ruggedness and good design. Frankness and clarity faded away when they started marketing to the naive; ruggedness is now a game that many play; and design now is a matter of chipmanship, a different political and R&D ballgame. What will be left in the long run for this dear old company that once gave us so much is unclear.

Digital Equipment Corporation, in response to the "Energy Crisis" of 1973, didn't turn out their Christmas tree. Instead they hooked it up to a water wheel they happened to have. Typical.

I attended a tragic sales meeting at a small college. DEC was making its VAX presentation. The college really needed a VAX, but the salesmen were appalling. They spoke incomprehensibly and made the college people—who were all very intelligent—feel small.

I tried to ask helpful questions and aid the presentation, but it was no use. They were determined to blow the sale, and did.

Afterward I told my friend Roger Gregory about this idiot DEC salesman (deeply suntanned) who couldn't speak English to a non-technical group, and kept using gibberish like "functionality."

Roger: "He thought 'functionality' was English, Ted."

.....

## DEC, The Halls

A wickedly funny description of DEC's home factory, fairly accurate, can be found in a neatsy belletristic book called *Travels in Computerland* by Ben Ross Schneider, Jr. (Addison-Wesley, paper, \$6), esp. pp. 73-75.

Even though they had the first commercial desktop computer, DEC never understood that there would be a Personal Market.

Except for engineers. DEC understood engineers, and the PDPs were sort of personal computers for them.

But reputedly, President Ken once stood up at a conference, when computer games were mentioned, and decreed heatedly, "DEC computers will never be used for playing games!" Which was already historically false.

And after the IBM PC they brought out their Rainbow personal computer—too expensive, and scarcely better. It was a floppola.



# WHO SAYS *Elegance* DOESN'T SELL?

Wozniak would have us believe he just did it for love and enthusiasm (and to impress the guys at the club). Hmm: doubts arise easily. But his

## APPLE II WAS A GREAT PACKAGE.

The original Apple II, as put together by Steve Wozniak, contained: built-in text display (which in those days was not yet standard; most computer fans still attached "terminals"); built-in low-res graphics; built-in high-res graphics; a cassette interface that would use a regular audio cassette recorder for storage; an interpreter which more or less mimicked the hardware instructions of the PDP-11 (called by Woz "Sweet 16"); built-in integer BASIC, written in Sweet 16; and a resident debugger for machine-language programs. Early purchasers also got a cassette of a Breakout game, written in integer BASIC.

This was just a year or so after the early Altairs and Im-sais, when every one of these features had to be purchased separately (if they could be obtained at all), and were not guaranteed to work together—just like today's boards and software for the PC.

Oh yeah. You also got the computer chip, power supply and memory, all packaged complete in a typewriter-like case with keyboard right on it. And a reliable bus (unlike the very flaky S-100 bus of that time) with six accessory slots.

Considering what an excellent collection of features the Apple II contained—it was a panoramic offering, good for beginners up to experts—it's amusing how the chip-monks would bad-mouth it. Sure, somebody who wanted Custom This and Special That—or lower case—could go mix and match. But the Apple II was a single unit, the price was okay, all the elements worked together, and it was reliable. Apple's first success was richly deserved.

## AND DESIGNED THE RIGHT WAY.

Given all that, you can tell who Wozniak designed it for. He designed the right way, the way everything good is designed, for himself. Wozniak's instinctive understanding was worth far more than any amount of market research. (The Apple III was designed by a committee, and you can tell.)

An evil motto that has appeared in Apple advertising:

**"WE INVENTED THE PERSONAL COMPUTER."**

This is a bald-faced lie. They don't need to say that.

# APPLE

Great legends built America, and Apple Computer carries on the tradition, mythologizing us into the sunset. Jobs'n'Woz, just two apres-college kids, blundered (they say) into incredible success: they just happened, casually and without premeditation (goes the myth), to throw together a superb personal computer almost completely by accident, and build a billion-dollar company by being the nicest guys in the world.

Apple is Wall Street's idea of a counterculture company: like they wear jeans and play hi-fi at work, isn't that a trip? The world sees a polished yuppie-craftsy image, carefully maintained by Regis McKenna Associates. Actually Apple Computer is about as counterculture as Pepsi-Cola (whence its current president): a computer company that sells a fancy sideline of beach-towels and glassware etched with the Logo of the Great Striped Fruit.

But Apple Computer certainly does attract counterculture customers, which today means academic and upscale. Folks who wear jeans and earn over \$75,000 a year just love the Macintosh. The problem is with more conventional people who already think they know what a computer is, and are affronted by the Mac's simplicity and funkiness.

## APPLE'S HISTORY

Steve Jobs and Steve Wozniak (called Jobs'n'Woz) were an unusual and powerful pair. Woz built the computer, Jobs built the company.

From the media portrayals, many people get the impression of a good guy-bad guy pair, with Woz the cuddly huggypair and Jobs the meanie.

The reality is much more complex, as are both men. Both are design geniuses with a strong drive toward elegance, simplicity and achievement.

Woz was a hardware hacker who, when motivated—and especially when pushed by Jobs—saw things to elegant completion. His greatest power was in the creation of dense electronic designs with a simple exterior.

Jobs was a Seeker After Truth in the classical Goethe style—wandering to India and Oregon to try various gurus and primal screaming, experimenting with eccentric diet. (His concern with health and the eating of fruit led to the company name.) He did odd electronics work, including game design at Atari (where he was not well liked).

Jobs'n'Woz grew up in Northern California, and knew each other from high school. Their first business together (according to Moritz' The Little Kingdom) was selling high-class blue boxes (see p. 159), enabling users to steal long-distance service from the telephone company. They have since made no secret of this, but unlike their mentor John (Cap'n Crunch) Draper, have never been prosecuted for it. (Even his Blue Box showed Woz' extraordinary design elegance, and it is a Collectible.) The two of them also designed the internals of the game "Breakout" for Atari, but Atari engineers couldn't understand their circuitry and rewired it.

Then in '75 came the Altair and the Homebrew Club (see p. 18), which attracted Wozniak. Woz (who had already built a computer of his own) wanted to do a little computer with one of the new processor chips. He chose unusual chips for both the processor and memory, and configured them in an elegant, non-obvious way. It was original, compact and clean. A true hacker (not just a hardware guy), he also programmed a PDP-11 simulator for it ("Sweet 16") and Integer Basic.

In April 1976, Jobs'n'Woz showed the Apple board at the Homebrew Computer Club. Woz explained his board and what his integer BASIC would do. Jobs asked if anybody wanted to buy it. Nope, not in that highly competitive group.

But real impetus was around the corner. It was Paul Terrell, hearty and dynamic supersalesman, creator of the Byte Shops and perhaps the first hustler of personal computers, who kicked Apple down the road. In the summer of '76 he astonished Jobs with an order for 50 assembled Apples.

This galvanized Jobs into action. He slapped together a team to make the boards and fill them with parts. He also found a third key tekkie: Frederick Rodney Holt, Marxist motorcycle racer and generalist, who created a radical, lightweight power supply that made the Apple fanless and lightly portable.



*But Jobs also found some more conventional individuals who saw the promise of the Apple and built a hard-driving company to put it across.*

For the first year they stayed a small team, working round the clock. Borrowing money, they took the Apple II to successive personal-computer fairs, made color logos, fancy conference booths, plastic molded cases. In a few months they sold hundreds. A year later they had sold thousands.

By now the fruitily-named company was fruitful indeed.

Venture capitalists had started kicking in in 1977; in 1979 they put in millions. Very soon Apple was ripe for plucking. In December 1980 Apple went public, and by New Years' Eve its shares were worth almost **two billion dollars**.

## HUBRIS

Apple computer's rocketing growth brought with it too many people, called "the bozo explosion"—bureaucracy, committees. And worst of all, it brought smugness:

"...the company also became intrigued with the notion of empire, and an aggressive conceit threatened to unravel much of the earlier success." (Moritz, p. 290.)

They also expected (like many others) that IBM could not operate in the personal computer field:

"Markkula [the marketer] could barely conceal his irritation when asked how Apple planned to respond to IBM. 'We've been planning and waiting for IBM to get into the marketplace for four years. We're the guys in the driver's seat. ... We're going to outmarket IBM.'" (Moritz, p. 310.)

## LISA AND MAC

In the late seventies, from its pinnacle of success, Apple was threatened only by one competitor. Apple could guess that IBM was readying a personal computer, likely to be a conventional one, with a conventional operating system. Perhaps Apple knew they could not compete with such a machine unless they merely copied it (as many would). Instead they chose to leapfrog to the kind of graphic-style machines that hackers like.

Sounds rationally planned, right? But actually Steve Jobs was wandering through Xerox PARC (see p. 128), saw the

Alto, and being perceptive and discerning, said "I want to build that!" If only the rest of American business (including Xerox) could have such discernment.

The general idea was good: Apple would build swell desktop machines using the bit-map graphic and interactive styles developed at PARC, picking up the cookies that Xerox dropped (and the mouse that Engelbart invented; see p. DM 16). But then they went and built **two different incompatible machines**, the Lisa and the Macintosh. It was crazy to have two teams. It was crazy not to make the machines software-compatible. (That's the sort of thing IBM does, assigning the same project to two groups, and picking a winner; but only IBM can afford to kill the losing version.)

The Lisa, a lovely single-user multi-tasking machine, was a market disaster. It came out in late 1982 and cost ten thousand dollars, bundled with word processing, spreadsheet, line-drawing graphics and scheduling programs—an executive decision and study box with wonderful screen graphics. It was too strange and new for corporate America. Like the Xerox Star, another attempt to commercialize that style, it did not sell.

## MASINTOSH

"C" before "I" is supposed to be soft in English, so the correct pronunciation is "Masintosh." (If you want to sell to universities you should think of these things.) But the fundamental misspelling aside, it

really is a very nice thing, Macintosh, your basic yuppie-craftsy computer.

It was separately developed under Steve Jobs' personal supervision—much the same thing as the Lisa but smaller, and cheaper and **INCOMPATIBLE WITH IT**. (And it couldn't do several things at once, like the Lisa.)

The Mac came out in 1984 with just a word processor and a graphics program—but the graphics program had beautiful fonts and a whole art department all in one package, Macintosh's strong point.

But after that, more software took a long time coming out, because of the Mac's immense technical difficulty for developers. That stalled the machine a lot.

## The Mac Is Tough To Program, Like The Atari



*Apple made a serious miscalculation as to how difficult they could make the Macintosh for programmers. Inadvertently they followed in Atari's footsteps.*

*Atari erred supremely in the design of their 400-800 computer series. They wanted great graphics but by bit-fiddling choices made it very difficult and complex to program, which was not necessary and sank the machine. (A good one might have saved the company.)*

*Apple learned half the lesson. The great graphics of the Macintosh are straight bit-mapping, which is clean and straightforward.*

*In addition they wanted extreme simplicity for the user. This too was wonderful in principle. Unfortunately, the way they did it inside—the messy internals of the "Macintosh toolkit"—made monstrous problems for the program developer.*

*The Macintosh has been set up to make all user programs easy under a consistent set of rules—with pull-down menus and safety nets of various kinds. But it turns out to be very hard to make programs that work within this system. (By the time the Mac was publicly announced, some advance developers had given up and discarded their early Macs. Even Lotus, who hadn't missed product deadlines before, was seriously delayed in the release of their Jazz program for the Mac.)*

*External simplicity is all very well, but they should have given attention to the inner simplicity as well.*

Jobs appears to be a hard-driving, demanding and tireless visionary—often discourteous, perhaps cruel, but greatly inspiring to those who work with him. ("He's extraordinarily seductive. He would have made an excellent king of France.")

—Jef Raskin, quoted in Moritz, The Little Kingdom, p. 260.)

The parallel to Thomas J. Watson is instructive. The drive, demandingness, leadership and self-certitude are the same.

Cynics say the Mac came about because **JOBS WANTED TO SHOW THAT HE COULD DO WHAT WOZ HAD DONE**.

A number of decisions he made—incompatibility with the Lisa, limiting programs to 64K—were disastrous. (Fortunately, though, all the software was designed so it could go directly to big screens and big memories, and we can credit Jobs with that too.)

**The Great Truth that used to be Apple's Motto:**

**SIMPLICITY IS THE ULTIMATE SOPHISTICATION.**

**But for some reason they don't say that any more. They still have a right to. Even if Corporate America is too stupid to see the truth of it.**



## THE SLIDE

Meanwhile had come IBM's PC, which copied the Apple II's open-architecture strategy with more up-to-date chips, greater expandability, and a more powerful (though awful) operating system. Though there was great loyalty to the Apple II in schools, among IBM-haters, and in certain kinds of aspiring young families, Apple quickly lost its leadership of the personal-computer industry. And started to roll downhill.

Layoffs began. Nearly a thousand people left or were fired in '83 and '84—including the older top brass, leaving Jobs in absolute control.

## NUMBER TWO

Jobs picked John Sculley to be President of Apple, wooing him from Pepsi-Cola. He was chosen, some say, because "he knew how to be Number Two"—referring to his experience heading up Pepsi-Cola. *Fortune* magazine depicted Jobs and Sculley taking long walks together, discussing philosophy and ballet, becoming As One.

Honeymoons end. The continued souring of sales set the stage. At the board-of-directors meeting on 31 May 1985, Jobs planned to fire Sculley, but it went the other way. Sculley, a smooth politician, had the board on his side, and Jobs found himself on permanent salaried

vacation, poor guy. This put Sculley in position to try to unite and salvage a company in decline and disarray, topheavy and in trouble, with a skyrocket past and an unknown future.

## DOWN TO THE CORE

Apple computer had shrunk. Wozniak had left and sold his stock. The Lisa failed (like the Apple III), leaving Apple with only **two** wholly unrelated and incompatible product lines: the weary and overpriced, if not exactly obsolete, Apple II line, and the quite nice Macintosh, a good personal and home machine.

No longer Number Two, Sculley, a marketing man, has brilliantly reintegrated the company, brought the two product lines "closer together"—despite their continuing mutual incompatibility.

Best of all, he kept the Mac alive with the fad of "desktop publishing"—see p. DM 27—until he could bring out the Mac II, a ferocious and superb workstation computer, with OPEN ARCHITECTURE again, color, big screen, stereo sound, and much much more. Though few have been delivered at this writing, the Mac II is now the *de facto* workstation standard.

## READ ABOUT IT

Michael Moritz, *The Little Kingdom* (William Morrow & Co., 1984).

## Can Apple 'Capture The Desktop'?

*Apple's duel with IBM for the corporate desktop—PC versus Macintosh—is portrayed by the media as David vs Goliath, Han and Luke versus the Empire. But it's much stranger than that.*

*IBM's PC imitates and expands on the Apple II—open architecture, a package pretty much copied from the Apple II (at least till the IIc).*

*While the architecture of the Apple Macintosh (when it first came out) was a closed system, of the kind IBM used to stick you with. Note, however, that*

## BOTH MACHINES WERE STEPS OF LIBERATION.

*The PC is standard, modular, powerful, and takes a thousand gadgets; the Mac is more powerful, far easier to use, and much more fun. (If a trifle clumsy and still limited by the design of its "desktop" operating system.) Which makes it very practical for offices. But it has taken corporate purchasers a while to see that.*

# THE COMMODORE (and Jack Tramiel) STORY

From a typewriter repair company in the Bronx to the Niagara of computers.

Millions of computers out there bear the Commodore label, though many have been relegated to closets. Commodore may have made the most computers of any company in the world (though this is an honor we may expect to see pass from hand to hand with increasing frequency, as the number of computers sold goes into the hundreds of millions and beyond).

The Commodore story is really the story of its founder and (till recently) its mogul. Jack Tramiel, cunning and hard-driving, is the survivor's survivor: he lived through six years of Nazi concentration camps, including Auschwitz. That perspective makes fierce business dealings mere candy to him.

Tramiel takes credit for keeping the Japanese away. His famous quote: "The Japanese are coming. So we will become the Japanese." (Note that he referred to volume and price, not quality control.)

Commodore was early into the mass distribution of little calculators. First in calculators, then computers, Commodore maintained a volume lead by bearing down on the price edge.

Shortly after the Altair came out in '75, Commodore announced the PET, a complete-with-screen computer for eight hundred dollars. It became the leading dinky in Europe: Germans made it a factory machine, English marketing concentrated on hobbyists.

Then, with the Vic-20, they introduced a computer (designed by Bob Yannes) for about \$300, the lowest price ever. Soon after that came the Commodore 64, a 64K improvement with good graphics and very good sound for about \$500. And sold millions. (Note that the Apple II also uses Commodore's 6502 chip, so that Commodore profits a little from every Apple II sold.)

Tramiel has now left Commodore to buy Atari, and, like Kubla Khan, is bringing in his sons to run the Atari empire. Whether they will remain moguls is another question.

Commodore has brought out the Amiga computer (see p. 91), which Tramiel commissioned. He could have made it the number one personal computer. Maybe. They could not.

## ONE VERSION OF THE COMMODORE STORY

Michael S. Tomczyk, *The Home Computer Wars*. Compute! Publications, Inc.



# ATARI: DANGEROUS COMPANY, RISKY BUSINESS

"Atari" means you're in danger, at least in ancient Japanese game of Go. When Nolan Bushnell gave that name to his electronic game company, whom was he warning?

Bushnell created the first video game, "Computer Space," which proved too complicated. But he hung in and created "Pong," the second video game, in 1972. "Pong" was such a smash hit that the first unit jammed on Day Two because it was too choked with coins.

Bushnell found he was riding a rocket. Wildly successful, he rode that rocket as the incredible video game craze grew and changed. Atari put out new arcade games every few months.

They also pioneered in putting out a game machine for the home—the VCS, an almost-computer. (Unfortunately the VCS could do almost nothing *but* shoot-em-up games; its 6502 chip is kept occupied most of the time refreshing the video. All computation has to be done during the brief periods at the end of scan-lines. That programmer Warren Robinett was eventually able to put BASIC on it—however crude—was a miracle.)

The public grew fickle and Bushnell's inventory grew large. In 1976, overextended, he sold Atari to Warner Communications for many millions of bucks.

(Bushnell then went on to invent the *Pizza Robot*, singing as you eat at Bushnell's syndicated gastro-gamery, *Chuck E. Cheese's Pizza Time Theater*. This too was a temporary great success.)

## WARNER YEARS

Under Warner, Atari took a new turn. Ray Kassar—a suave "marketing genius" with a background in cloth and clothing (and himself an elegant dresser), was brought in to run Atari as a division of Warner Communications. He demellowed the company; pot-smoking was no longer condoned and business hours were made exact.

Kassar marketed the home video games harder and brought out the Atari computer, impressive in its demo capabilities but a design disaster (see

*Candy-Machine*, below). Yet the money was now coming in in torrents: something like **half a billion** dollars in 1982, from all the VCS cartridges and arcade games.

## A LICENSE TO DREAM

Kassar dismantled all the small R&D groups around the company and instead hired brilliant and sardonic Alan Kay, the highest hacker of them all, as R&D czar. Kay, inventor of the Object paradigm (see p. 62) and the Smalltalk language (see p. 62), dazzled the firm with his wit, erudition and wide acquaintance.

Kay was given an R&D budget of ONE HUNDRED MILLION DOLLARS A YEAR, instructed to "take risks" and "dream." So instructed, and being under no pressure to develop products in a hurry, he proceeded to spend the money in a dreamlike fashion. He supported talent of his choosing throughout the land; he set up labs around the country to back various talented people; he hired MIT's Logo Lab.

There may not have been complete understanding in the Warner community, or even at Apple, of where Kay was coming

from, or what he was supposed to do. "I'm not sure if Alan Kay has the world's best track record for new products," the book *Zap* quotes an analyst as saying. (p. 134.) Considering that his great achievement, the Smalltalk language, had not yet been released, this was an understatement. But this quote suggests that others saw him, not as a pioneer, but as a productizer expected to get new stuff out the door.

What Alan Kay had done at Xerox was start and develop extraordinary and rev-

## The Candy Machine

One of the nicknames of the Kassar-era Atari computers, when they were first under wraps, was "Candy." Indeed, their colorful effects were candy to the eye.

## ... And Why It Jammed

The Atari computer was a chipmonk's dream, but a dog to program. (I use the singular because the 400, 800 and later models are essentially all the same. The post-1984 machines are another matter.)

With an endless variety of special-case switches and tricks, the Atari computer could make big pixels and little pixels (see p. DM 92), even horizontal stripes that each had pixels of different sizes. Its famed "player-missile graphics" were small pictures that could be superimposed, bounced around, and noticed by the program if they collided. So you could do arcade-type effects on it. (This design emphasis came from the assumption that your program involved shooting things at other things. Which is what some guys thought was important.)

Yes, great arcade-type effects could be done on the candy machine—but with great difficulty. No thought had gone into making these effects accessible to beginning programmers, or any ordinary human beings. It was a chipmonk's dream with innumerable modes and special cases for a variety of different graphic effects—but not elegant, not general, not simple, not clean, and very hard to set up. Indeed, you could say it consisted only of special cases.

Oh, wonderful and eye-popping things could be programmed for it, and several were: "Star Raiders," Lucasfilms' "Ballblazer" (action by Dave Levine) and "Rescue on Fractalus" (with fractals by Loren Carpenter—see p. DM 118) are masterful. And all of them make the machine a good purchase. Now that you can get 800s remaindered at a hundred bucks, it may be worth it just for "Star Raiders" (but you have to buy a disk for "Ballblazer" and "Fractalus," and that raises the price a lot).

## WHAT STICKS IN THE TEETH

Also the Atari computer never made money. It just threw away hundreds of millions of Atari's arcade loot, and, thus subsidized, lowered the prices of the industry in general.

The chipmonks lament the failure of the machine, saying that the public didn't "appreciate" it. Indeed not. It was a machine only an engineer, or a maniacal bit-fiddler, could love; not a graphical computer, but a computer whose special graphics engine was suited only to engineers.

## Kay Is Not For Candy

*The Atari computer came out in 1980, and Alan Kay arrived as Atari's chief scientist in 1981. This represented a marvelous irony.*

*Atari's computer line, the Candy machines, was the opposite of everything Kay had worked to achieve: a world of simplicity, clarity, and everyone being able to program easily; a world whose computer graphics were done by uniform planes of bits, rationally operated upon—procedures clean, uniform, profoundly thought out. (See p. DM 92.)*

## WHERE ARE THEY NOW?

- ➡ Restaurobots have gone to rest in unlikely places. (See "Robotics," p. DM 140.)
- ➡
- ➡
- ➡
- ➡



olutionary ideas, not prepare new products for the market. His new Atari title was Chief Scientist, and he is indeed a scientist, but not a product packager. Secure in his brilliance and achievements, he was in no great hurry for Product. His long-term goal remained the Dynabook (see p. DM 51), the powerful portable computer that would be extremely easy to program, and his research projects pointed in that direction—but dreamily, rather than at competitive speed.

## THE COLLAPSE

Suddenly, very suddenly, Atari's money ran out, as if a faucet had been turned off. The public deserted the videogame parlors, then stopped buying video games. It was one of the greatest income collapses in history.

Whatever Kassar did wouldn't have mattered. What really happened was that the videogame craze came and went. Shoot-em-ups and pound-for-points—chewing gum for the mind—had gone away.

So the Warner Years closed with a bang: in a surprise move, Warner sold its Atari division to Jack ("Business is War") Tramiel, who had built Commodore, in 1984. Warner even **lent him the money for the purchase**, clearly under the assumption that only a fierce fighter like Tramiel could rescue anything whatever of their investment.

## THE NEW TRAMIEL ERA

True to his reputation, Tramiel rolled heads: researchers at the Atari Logo Lab, for instance, were simply locked out. (Some have complained to me that they lost their books and personal possessions.)

Tramiel's reborn Atari—the third incarnation of the company—has a line of lean, mean "personal business computers" that use the pseudo-Macintosh GEM software from Digital Research, with a 68000 chip. With neither the momentum of the PC, the hacker veneration of the Mac or the eye-popping graphics of the Amiga, it is still not **special**. But they offer the most crunch per buck in the industry.

## BIBLIOGRAPHY

Scott Cohen, *Zap! The Rise and Fall of Atari*. McGraw-Hill, 1984.

# XEROX

*The Xerox Corporation occupies a unique place in history, and a unique place in the computer field.*

*Originally they were a company called Haloid that in the 1940s made photographic papers and chemicals. But they had the vision to buy the rights to the electrical copying process invented by Chester Carlson, which he had first demonstrated crudely in Long Island City in the thirties.*

*How long the process would take to develop no one realized. Haloid poured money into it for years, and many sold their Haloid stock. But at last they brought out their first copier, the 914, in 1962; changed their name to Xerox, and took the world by storm, creating the Copier Way of Life.*

*So they became a Big Fat Company with a great reputation for innovation.*

*And they said they were going to challenge IBM across the board in computers, and they did (down to copying the names of all IBM's departments at Xerox Data Systems), and failed miserably, in 1975.*

## X MARKS THE SPOT

When Xerox Corporation entered the computer business a few years ago, it announced that it was going to challenge IBM head-to-head for domination of the whole broad field of Information, whatever that is. Xerox made copiers, but saw the handwriting on the drum: eventually the handling of written materials would cross over into the computing realm, few are sure in what way. (For three future directions that have been proposed, see Engelbart, p. DM 16, PLATO p. DM 93, and Xanadu™, p. DM 141.)

*They bought Scientific Data Systems and ran it into the ground completely, closing down at an immense loss.*

The news was presented in the framework of grand tragedy, the Promethean collapse of overextended ambitions. Evidently, Xerox management pushed too hard in two incompatible directions—

building slowly for the eventual challenge of IBM, vs. showing profits quickly. The firm fell between the boat and the dock, joining RCA and General Electric and the other big companies that found they couldn't make it selling computers.

But Xerox is not as far out of the field as some might think. Now, the Xerox Corporation has said that they intend to replace paper (or, the way I heard it, "Somebody is going to replace paper with screens, and it will either be IBM or us, so let's have it be us.")

Well and good. Save the trees and stem the grey menace.

*And so they built the remarkable Xerox Palo Alto Research Center, known to everyone as PARC, a strange university-like institution in the hills.*

In this secret mountain hideaway—well, not too secret—Xerox still has perhaps the sharpest bunch of computer rascals in the world. And they are planning way way ahead, to the time computers are practically free. If Xerox gives them their head, and doesn't cut back, the corporation will have little trouble in triumphantly returning to the field five or ten years from now, conceivably knocking IBM off its feet in the new markets of that day with a karate-like sweep.

PARC's atmosphere of California Mellow can mislead the unwary.

I spoke there a few years ago and found it an astonishing experience. First, there was a busy volleyball game outside when I arrived, and when I asked for the person I was going to see, the reception-person said to pull up a beanbag and wait till he had finished playing volleyball. Later, when I addressed a group, it was in a room furnished only with a mountain of those beanbag sacks. As people came in, they would pull beanbags off the mountain and sit down on them.

So far so good: California Mellow. So I went into my rap, and everybody sat listening. I had no idea if I was getting through. Since what I try to tell people begins where technology stops—moral precepts, as it were, for organizing ideas and systems in the world of the future (see the whole DM side)—I'm used to people looking confused, or worried, or angry, or even walking out. There was none of that. Was I getting through? Or were they all just stoned?

I think I just sort of stopped and said, "Is everybody following this?"

There were smiles and I think someone said, "We're with you, Ted."

And they were. It was the only place I've ever spoken where the audience was on the same wavelength, going straight on into Systems Design for Future Man. Very moving.

What's gone on at PARC has a lot to do with the replacement of paper (see p. DM 27). Xerox has poured hundreds of millions, perhaps billions, into this institution, and its general contributions during the 1970s were extraordinary.

Best known is the Smalltalk language work (see p. 62)—initiated by Alan Kay, carried on by the stalwart Learning Research Group under Adele Goldberg; and finally released to the world in the mid-eighties.

But less flamboyant has been the work at PARC on artificial intelligence, pattern recognition, graphics, interactive systems and computer languages. Effectively Xerox PARC has been the equivalent of half-a-dozen top university departments of computer science, issuing numerous pamphlets and articles. (Presumably the request postcards are counted, like those for radio shows, as one metric for rating the authors.)

Xerox PARC has a sort of idyllic reputation, but it is in many ways a tough place to work. Because so much work goes on there in immensely important fields scarcely known to the outside world, its atmosphere is perhaps more highly competitive than at any university, and its tea-pot-tempest politics are widely known.

However, the relation between PARC and its parent megacorp is rather like the traditional town-and-gown (or perhaps surf-and-turf) conflict between academics and mundanes. The paper wall-lopers at Corporate think of the PARCKies as wild-eyed kooks, whereas at least some PARCKies see the corporate folk as Babbits and Yahoos with no comprehension of the great work they have been doing.

The Corporate side of it was told in Fortune a couple of years ago. To hear this straightfaced account from a very non-intellectual perspective, Xerox Corporation nicely asked a bunch of academic crazies to tell them what they should build for the world of the future, and the academics *wouldn't get it right*. Instead of designing an office of the future around copying machines (or whatever

Corporate thought they were supposed to do), those intractable loonies kept talking about personal computers and interactive graphics.

Which was of course the *right* thing to do.

Nonetheless, Xerox Corporate took some of PARC's work and Productized it. That is, they had other divisions manufacture the Ethernet (see p. DM 23) and build rejiggered workstations visually similar to the Alto computer. Then Xerox tried to sell a networking office system built around this graphical approach. While they built very nice stuff, to outfit an office with it costs hundreds of thousands, an immensely radical step to take in any customer's office. Sales have not been successful. They have also thrashed around selling Z-80 dinkies and are now involved with PC clones.

Why didn't Xerox just take the Alto and sell that with Smalltalk? The political ins and outs are not now known, although presumably they will make an interesting chapter of history in ten years or so. (Xerox did put Altos out in classy places like the White House and Congress in a sort of seeding operation, and sold them to a few publishers and whatnot, but nothing great came of it. The software they supplied was improfound. (I asked a White House staff member what they used their Alto for, and he said they *just used the word processor for making up schedules*.)

It is conceivable that Xerox may someday get its computer act together and put out products that truly exploit the graphics and object-oriented language work that they've paid so much for.

(Note: the above predictions are based, of course, on the assumptions of Xerox management knowing what it's doing. Assumption of this type in the computer field all too often turn out to be without basis. But we can hope.)

## "WE'VE TRIED IT ALL"

*While the contributions of Xerox PARC scholars have been considerable throughout computer science, other people sometimes find the attitude of PARCKies and/or PARC alumni to be insufferable. "We've tried it all," they say, and don't always bear what you actually said.*

*But it depends on the individuals. Some are up-tight, some are mellow and very nice.*

"Xerox PARC is the most intensely status-conscious and up-tight place I've ever seen."

—One observer

# AT&T

*Let's see what the old Computer Lib said about AT&T coming into the computer business and selling UNIX (which had not yet occurred):*

Unfortunately, this raises certain grave questions, since the telephone company (of which Bell Labs is a branch) is not supposed to be in the computer-programming business; and those who are in the business are dismayed by the idea of such a competitor.

*Well, they need not have worried.*

## AT&T

AT&T has done one thing right, very right, better than anyone in the whole world; and that is make an extraordinary, reliable, telephone network that spans our continent and connects much of the rest of the world, in orbit and under the seas.

Moreover, they had Bell Labs, which has done some of the finest computer research in many areas, from music to the physiology of the ear to the development of UNIX.

Then they were broken up by antitrust, to make room for telephone rivals, and allow AT&T to compete in computers. Many thought AT&T would be a natural rival to IBM. Wall Street thought a tiger was being unleashed.

## MEOW

*So their first big computer offering was a PC clone, and by whom was it made? Olivetti, for Pete's sake.*

Now AT&T is going through shakeup after shakeup, trying to grab market share as if there were something to grab, as if something had "gone wrong," and as if they had had some special knowledge that entitled them to a share of the computer business.

## ONE RINGIE-DINGY

IT'S TRUE that UNIX was developed at Bell Labs.

But not as a product. It was a subversion by brilliant programmers. For years the AT&T product planners never wanted to let UNIX out the door, as it did not fit in with some current "master plan."



## TWO RINGIE-DINGIES

UNIX only got out the door because Programmers broke the door down demanding it. And, of course, product planners came and went, and by and by came to imagine that UNIX was their own doing.

## THREE RINGIE-DINGIES

So now AT&T pretends they created UNIX because it was what the marketplace needed. (In fact, UNIX is what the programmer community needs, but should be hidden from vulnerable users, like guns from children.)

## SORRY, THAT LINE IS OUT OF ORDER

No, the front-office marketers don't understand computers anywhere, certainly not at AT&T, and aside from UNIX license fees they may never get anywhere at all in the computer business.

*"AT&T is the type of company that has such vast financial resources that they will eventually get it right."*

—A computer retailer, quoted by the Wall Street Journal, 15 Mar '85, p. 10.

**At the roulette wheel, reliance on more money to save you tends only to lose more money.**

# DATAPOINT

"They never knew why they succeeded, and they never knew why they failed" —Anon.

**Note: the following expresses the opinions of the author, and is based on information which is easily available outside the company.**

The story of Datapoint will be rehashed again and again in textbooks—a high-flying computer company that grew from making terminals to computers to computer networks, and then came to an abrupt crash, astonishing many.

The MBA candidates will pore over profit-and-loss statements, trying to find out something wrong in administrative or financial technicalities. But they will probably find nothing.

The problems did not lie there. The problem was what Datapoint was making.

For a decade Datapoint made excellent products of clean, simple design; but in a few short years they managed to destroy this simplicity, producing a hideous tangle. Yet they thought their customers would follow like sheep because Datapoint was providing "the functionality"—meaning that you could still use their stuff if you really knew what you were doing and worked very hard to hold it all together.

No, only one company can force hideous complexity down the throats of its customers, and that company is not Datapoint.

## ACHIEVING MORE WITH LESS

Originally, the company did not even know they were making a computer. What Vic and Harry designed was just THE SIMPLEST PROGRAMMABLE TERMINAL—intended to have just enough computer in it to allow people to change its terminal characteristics by software.

Customers were beginning to want so many options on their terminals that the wiring was getting too expensive: instead of having physical switches to change the functions of a terminal, it made sense to Old Vic (his grumpy manner always made him seem older than he was) to fix up a

terminal whose options could be changed by program.

So he hired young engineering student Harry Pyle, who would prove to be the Tesla of Datapoint. Together they designed one of the most important machines in history, though they thought it was just a terminal.

It was a terminal, yes, a rugged and reliable terminal that caught on and made Datapoint a flash success in 1970.

But making it just enough computer to act like different terminals was making it just enough computer to be a computer.

So here they were, roaring along, selling these terminals, and little suspecting their historical importance. Until they noticed their customers were **programming the terminals as if they were computers.**

As indeed they were.

## HAVE IT YOUR WAY, IT'S A COMPUTER

Datapoint was young then, and able to listen; and so they turned around and said Okay, It's a Computer, and started giving out computer software for it.

And this minimal computer was in fact the prototype of the 8008 and grandfather of today's PC. (It was to become most of the important microprocessor chips of our time—the 8008, the 8080, the Z-80; and Intel's awful upgrades, the 8088, the 8086, the 186, the 286 and whatever lies beyond in our future. (See p. 97.) And these chips in turn furnish the architecture of the IBM PC and its clones.)

Anyway, Harry Pyle created an extremely simple and clean beginner's language, Databus, which caught on. Unlike Basic, it had various provisions for management of business records. He then went on to create a time-sharing version of Databus, calling it Datashare.

And Harry went on to create a superb local-area net (see p. DM 23), the ARCNET, whose operation was extraordinarily simple to the user.

The ARCNET was the first local area network to be offered commercially (in 1977), and is probably still the best, no doubt because it was created—and programmed—by Harry Pyle.

Then, in 1977, Datapoint brought out the ARCNET. (In one of the two mistakes that killed the company, they wouldn't tell anybody what it was, except for VERY qualified customers.)

Meanwhile a jolly, loquacious programmer named Gordon Peterson produced an operating system, which they called simply DOS. It was a fairly simple variant of the old DEC-type system—except that Gordon managed to make it work over a network of up to 255 computers. It also had reasonable password security across the whole network without too much complication.

**START SIMPLE, STAY SIMPLE**

Once again clean-and-simple had won the day. The combination of the simple processor, Datashare and the ARCNET was a wonderful package for that time, and a spanking success.

It was a superb package for small and growing businesses. All that you had to buy at the start was one computer with a disk drive. A beginner could learn to use it and produce all the software a small company needed. (I know one who did.) Then the customer firm could add processors along the network with **no difficulty or reprogramming**.

It was just great for the mid-seventies. Though not too well known, it sold well, especially in Europe.

**ACHIEVING LESS WITH MORE**

In that heyday of Datashare and ARCNET, say from 1977 to 1980, Datapoint's strength was **SIMPLICITY AND COMPATIBILITY**. But they then threw those away with a vengeance, going from great simplicity into a thicket of escalating complications.

This was the other decision that killed the company: to build a much grander and more versatile operating system. Thus was born RMS, or Resource Management System.

It all started reasonably enough. They thought they needed a more powerful operating system, one that would make their small processors able to handle huge files and more complexity.

Unfortunately, "more complexity" was the operative concept. Giving users many options while keeping things simple is the fundamental problem of software. Datapoint did not solve it, and indeed forgot all about simplicity.

The basics of RMS were programmed by—yes, Harry Pyle, along with Klavs Landberg. But the design was too hard to use: quite like MS-DOS, with a forest of complex options that had to be typed in by

users. This despite the fact that most Datapoint workstations were used by secretaries and clerks, not programmers.

Worse, after the basics of the operating system were built, it was moved over for "finishing" to other teams. (Harry Pyle, who **could** have made it wonderful, abruptly learned user design and did a brilliant programmer's editor called EASL.)

RMS was an immense effort, taking five years or so (depending on when you think it was finished). It was "MS-DOS done right," if that is not a contradiction in terms. For lovers of MS-DOS it would have been terrific, indeed a considerable improvement, in that it worked reliably over a network and had enhancements lacking in MS-DOS (like a MOVE command that would move a series of files, deleting only those originals which had been successfully copied).

But it was far too complicated. While you could still add a processor to the network just with a simple screw-together connection—one of the beauties of the ARCNET—just to set up software for a new user could take hours. The productizers had little sense of usability or comprehensibility.

There was too much generality in RMS, too many options. Deciding how to set up your files and working setups was vastly complex, involving many different things to keep in mind at once. (There was so much generality just in the password system that most users couldn't figure out how to use it; it was easy to lock yourself out of your own files. Also, anyone who could physically get to the file server—the computer that had the disks—could defeat all security for those disks anyway.)

**WALLPAPER**

As they began to see how bad the situation was, Datapoint tried to wallpaper over these complications with nicer interactive structures. But this was a cosmetic layer, and a thick one at that.

One coverup program, called Vista-View, was a nice multiwindow package. It was brilliantly retro-engineered by Mark Miller (see p. 108), allowing some programs to be used together and transfer data. Another, Vistascript—largely the work of Stuart Greene—allowed the user to select command options in a branching

*Datapoint's original successes are a wonderful example of how creative computer work should be done: by a **SMALL NUMBER OF PEOPLE**, often **ONE**.*

*The original computer was designed by **Vic Poor and Harry Pyle**.*

*The DOS operating system, working over a local-area net, was done by **Gordon Peterson**.*

*The technical problems of tuning and maintaining ARCNET protocols were handled by **John Murphy**.*

*And note that besides his work on the original computer, Harry Pyle did **Databus, Datashare and the ARCNET** virtually single-handed. That tells you something.*

*You could say, "One Harry Pyle is worth a hundred ordinary programmers," but that understates it. Because a hundred ordinary programmers can never produce the really good, well-coordinated technical decisions that can be achieved in one mind, let alone one such as his.*

*This calls into question the principles on which big computer companies are built. As well it should.*

*What one Harry Pyle can design, a team cannot; thus, through some sort of law of administration, the teams become larger and larger and the software worse and worse.*

Trying to lift morale in 1984, Datapoint gave employees bumper stickers that said

Datapoint—WE SPARKED THE REVOLUTION

which of course referred to small desktop computers and ARCNET, the first local area network.

But as 1984 wore on, and Datapoint continued to sink, some disgruntled employees changed their bumper stickers by removing the S.



piece of text. It was a good system for choosing options for a complex command.

But these wallpaper programs came out too late, and, by the time they were productized, were themselves too complicated for most people to install. And the window package, for various reasons, **could not be adapted to their enhanced word processor**—merely due to a slipup in coordination. There was little point in a window package without word processing, but it got shipped anyway.

#### THE END

Meanwhile the price floor fell out. The word processing you could do on an Apple II or IBM PC made Datapoint's look pitiful. And worst of all, Datapoint's computers were very slow compared to what else was available.

The mellow, business-as-usual atmosphere continued, but thousands were laid off in successive waves. Topheavy with bureaucracy, vice-presidents and committees, the company teetered.

The famous takeover of Datapoint by Asher Adelman in the spring of 1985—and his subsequent dismembering of it—should not have surprised anybody who knew what Datapoint was trying to sell.

## COMPAQ

Has done a fabulous job cloning IBM's personal computers and **improving them**. For this they justly deserve, and get, praise and customers.

## SUN MICROSYSTEMS

The Berkeley UNIX gang, especially Bill Joy, have a company that has been pushing the high end of the workstation market—generically "Sun" right now means a very good workstation machine. At this writing, they are presently holding at the top of the line, though challenged by the Jobs-Perot NEXT company.

# THEY CALL IT JAPAN, INC. FOR A REASON

Only a little more than a century ago, Japan didn't want to trade at all: after some bad experiences with Westerners, Japan closed itself to trade until Admiral Peary forced it open and taught them a lesson. They learned it very well.

Trade is now the center of Japanese foreign policy, and so MITI, the Japanese ministry of trade, occupies a unique position in determining what the country will make and sell abroad, coordinating the systematic, intelligent use and improvement of technology. There's nothing like MITI in the United States or Europe: an

industrial forecasting and coordination department. (We have an Office of Technology Assessment, but it does far less.)

#### THE COMPUTERS ARE COMING!

Now we hear all kinds of alarmist talk about how Japan is leapfrogging us to the next kind of computer: MITI has decreed that Japan will undertake to build "Fifth Generation Computers," using various artificial intelligence techniques (and especially the Prolog language, see p. 62) to do far more than has been possible up to now. This is not clear. (See p. 62.)

#### HARDWARE

However, in hardware construction—chip manufacture—the Japanese now have us beat hollow. They sell us chips at bargain rates, and we complain and call it "dumping." The United States has recently penalized the importation of integrated circuits from Japan.

#### But SOFTWARE?

Although there has been some original software out of Japan (most notably the astounding graphics of one man; see p. DM 118), there is so far only one program from Japan that has had lasting worldwide influence. And that is, of course, Pac-Man.

## TOUGH GUYS

Some people are a force wherever they are. These include Seymour Cray, Clive Sinclair, H. Ross Perot, Jack Tramiel, Steve Jobs, Alan Kay.

#### CLIVE SINCLAIR

Sir Clive Sinclair is a short but commanding man, redhaired, polite and slightly Cockney, who has created a worldwide empire of innovative electronic products. Indeed, it was he who assembled and marketed the first four-function calculator, first digital wristwatch, the first palm-of-your-hand television.

Very much the successful businessman he, but no remote executive: very much in evidence, and friendly, at his own show booths. When I thought something was wrong with my early calculator and brought it in to the Sinclair office, he himself checked it. Talk about lean management.

He is unjustly known in American computer circles for the ZX-80, a useless little computer kit that sold for a hundred dollars—but can't blame him for that price breakthrough, now can we? Many people of course hoped it would do for them—well, what people hope for from a computer—and, with only a K or so and

no storage or software, how could it?—so it was thrown in hundreds of thousands of closets, and some of the buyers went on to more useful computers and some didn't.

But Sinclair has done other, much better computers that have sold abroad, and perhaps we may expect great computer things from him in the future. He's got the smarts.

#### EDS and Ross Perot

Electronic Data Systems, or EDS, promised to make your IBM systems do what you needed. Which not even IBM would do. Thus certain big companies (especially the Blue Cross chain) contracted with them.

EDS' founder was H. Ross Perot, a lean, mean former marine and ex-IBMer. He hires other military people, and they are said to treat business projects like military campaigns. ("A bunch of sharks," some say.)

Indeed, the company became involved in real paramilitary operations. When the Shah of Iran was overthrown, members of the new Khomeini government held certain EDS staffers hostage. Calling on the military skill and training of some of his staff, Perot oversaw a guerilla operation to get them out. Read about it in *On the Wings of Eagles* by Ken Follett, if you missed the TV movie with Burt Lancaster.

Perot's fame grew when EDS went public, making him on that day a billionaire. Then General Motors bought

EDS and put Perot on the GM board of directors. However (presumably because of his shake-em-up ideas), in December '86 they asked him to go away, paying him **another 700 million dollars to leave**.

Perot has been shaking up education in Texas and trying to bring New York's American Indian museum to Dallas. He has now joined another crusty fellow, Steve Jobs, at a company called NEXT. ("I give that marriage two weeks," said a friend—several months ago.)

NEXT is making high-power desktop computers. The clout of Jobs and Perot creates a fascinating new situation in the high-power workstation market. Look out, Apple? Look out, IBM?

Ever the one-upman, Seymour Cray (creator of great machines like the 6600 and the Cray; see p. 89) is supposed to have recently said, "I'm glad that Apple is using a Cray to design the next Apple. I'm using an Apple to design my next Cray."

# THE BEHEMOTH



also known affectionately in the field as

- International Big Mother
- Itty-Bitty Machine Company
- International Brotherhood of Magicians
- "I'm Being Moved"
- Institute of Black Magic
- In Bleakest Mordor
- It's Better Manually

as well as

- Mother of Us All
- The Grim Gray Giant
- Big Mamma Crass
- Security Blanket
- Snow White
- Grey Menace and
- Big Brother

Also, since the arrival of the Blue color in the seventies,

- BIG BLUE
- Jolly Blue Giant
- The Blue Bulldozer
- The Empire (since "Star Wars")
- The Evil Empire (since Ronald Reagan's use of the term)



"IBM," as everyone knows, is the trademark of the International Business Machines Corporation, an immense company centered in Armonk, NY, but extending to over a hundred countries and employing well over a quarter of a million people.

IBM dominates two industries, computers and electric typewriters.

In the sixties, they had a firm hold on the "word processing" market (since eroded); in the early eighties, they had a firm hold on the personal-computer market (since eroded).

To many people, IBM is synonymous with computers. Some of the public, indeed, believes them to be the only computer manufacturer.

(Though by now, what with all the TV ads for competing computers, there can't be too many people with this impression any more.)

In cameras and film, there is Kodak. In automobiles, there is General Motors. And in the computer field there is IBM.

IBM sells some 65% to 70% of all the computers that are sold. In this respect, the balanced near-monopoly, they are like Kodak and GM.

These figures, rough at best in the seventies, are now shot, since various companies now sell computers by the million. In mainframes, however, I think the figures are still about right.

But there are important differences. Everybody knows what a camera is, or an automobile. But to many, if not most, people, a computer is what IBM says it is.

The importance of this firm, for good or ill, cannot be overstated: whose legend is so thick, whose stock prices have doubled and re-doubled, ten times over, to its multibillion-dollar mass; whose seeming infallibility—at least, as seen by out-

siders—has been the stuff of legend, whose style has proliferated across the world, a style which has in a way itself become synonymous with "computers;" whose name symbolizes for many people—remarkably, both those who love it and those who hate it—the New Age.

The rigidity associated in the public mind with "the computer" may be related in some deep way to this organization. As a corporation they are used to designing systems that people have to use in their jobs by fiat, and thus there are few external limitations on the complications to our lives that IBM can create.

Many people mistake IBM for "just another big company," and here lies the danger. IBM's position in the world is so extraordinary, so carefully poised (as a result of various antitrust proceedings and precautions) just outside of total monopoly of a vitally important and all-penetrating field, that much of what they do has implications for all of us. Ralph Nader's contention that General Motors is too powerful to function as an independent government surely applies even more to IBM. General Motors is not in a position to persuade the public that every car has to have ten wheels and a snowplow. IBM seems in some ways to have molded computers in its own image, and then persuaded the world that this is the way they have to be.

(Until personal computers, when they finally had to make a machine people could use freely.)

But IBM is deeply sensitive, in its way, to public relations, and has woven an extensive system of political ties and legends (if not mythology) which have kept it almost completely exempt from the critical attention of concerned citizens.

Thus it is necessary here, simply as a matter of covering the field at an introductory level, to raise some questions and criticisms that occur to people who are concerned about IBM. IBM presumably will not mind having these matters raised; their public-spirited concern in so many areas assures that where something so publicly important as the character of their own power is concerned, occasional scrutiny should be welcome.

*"What people give up in creativity and originality when they opt for IBM, they make up for in stability."*  
—David Bunnell, "Does the Mac Mean Business?"  
Macworld, March 85, 9.

## ● THIBMK

Some useful words for discussing the IBM problem. (Thanks to Computer Decisions magazine, in which some of these were first published. (c) 1973, 1975 Theodor H. Nelson.)

- ibmology the study of IBM.
- ibmosophy the wisdom of IBM; ibmosoph, one wise to IBM.
- ibmophily love of IBM.
- ibmolatry worship of IBM.



## A FINE PROGRESSIVE CORPORATE CITIZEN AND A WONDERFUL EMPLOYER

It is important to note first of all that IBM is in many respects the very model of a generous and dutiful corporate citizen. In "community relations," in donations to colleges and universities, in generous release of the time of its employees for charitable and civic undertakings, it is almost certainly the most public-spirited corporation in America, and perhaps on the face of the earth.

They have been generous about many public interest projects, from Braille transcription to donating photographers and facilities for films on child development.

The corporation sponsors worthwhile cultural events. "Don Quixote" with Rex Harrison on TV was terrific, Katherine Hepburn's "Glass Menagerie" was marvelous.

They treat their small suppliers honorably and with great solicitude.

IBM's enlightenment and benevolence toward its employees is perhaps beyond that of any company anywhere. They have rigorously upgraded the position of women and other minority employees; the opportunities for women may be greater there than anywhere else. They have upgraded repair of their systems, at any level, to white-collar status, and tool kits are disguised as briefcases. This innovation, making a repairman into a "field engineer," is one of the cleverest public-relations and employment policies ever instituted.

They are openhanded to employees who want to run for office, evidently regardless of platform. In the sixties there were peace candidates who worked for IBM, and evidently got time off for it. More recently, Fran Youngstein, an IBM marketing instructor, was a 1973 candidate for Mayor of New York on the ticket of the Free Libertarian Party, opposing all laws against victimless crimes (e.g.,

prostitution and odd sex), as well as opposing Day Care and welfare.

They also rarely fire people. Once you're in, and within certain broad outlines, it's extremely safe employment. For those who turn out not to fit in well, they have a tradition of certain gentle pressure-practices like moving you around the country repeatedly at IBM expense. This encourages leaving, but also exposes the less-wanted employee to a variety of opportunities he might not otherwise see, without the trauma and anxiety of dismissal.

*In 1985, IBM ran ads stressing their intention not to fire people despite the company's downturn.*

(It is said that there are IBM firings, but they are rare and formidable. Heywood Gould's description of an IBM firing (*Corporation Freak*, pp. 113-115), for which he does not claim authenticity, is nevertheless bloodcurdling.)

IBM's international manners (in its 115 countries) are likewise praiseworthy. Compared to the perfidious behavior of some of our other multinational corporations, they are sweetness and light and high school civics. Sensitive to the feelings of people abroad, they are said to operate carefully within arrangements made to satisfy each country. They train nationals for real corporate responsibility rather than bringing in only outside people. And they are sensitive to issues: for instance, they refused to set up an Apartheid computer in South Africa.

## THEN WHY SUCH A RANGE OF FEELINGS TOWARD IBM?

Among computer people, feelings toward IBM range from worship to furious hate (depending only in part on whether you work there).

Many, many are of course *employed* by IBM, and the devotion with which they embrace the corporation and its spirit is a wonder of the world.

## Quickie History of IBM

IBM appeared in 1911 as the consolidation of a number of small companies making light equipment, under the name C-T-R Company (Computer-Tabulating-Record). This was prophetic, considering how aptly it described the company's future business, and especially prophetic considering that today's stored-program computer was undreamed of at that time.

According to William Rodgers' definitive company biography *Think*, the company's creator was a shrewd operator named Charles R. Flint, dashing entrepreneur and former gun runner to the South American republics, who in his shrewdness brought in to run the company an incredibly talented, fire-breathing and self-righteous individual named Thomas J. Watson, even though Watson at that time was under prison sentence for his sales practices at another well-known company. The sentence was never served, and Watson went on to preside for many years over a corporation to which he gave his unique stamp.

Watson arises from the pages of *Think* as a sanctimonious tyrant, hard as nails yet reverently principled in his words; the pillar of fervid, aggressive corporate piety.

IBM was totally Watson's creation. The company became what he admired in others, a mechanism totally obedient to his will and implementing his forceful and inspiringly rationalized convictions with alacrity. As the Church is said to be the bride of Christ, IBM might be characterized as the Bride of Watson, molded to the styles of demandingness, pressure, efficiency and pietism which so characterized that man. But the ideas flowed from Watson alone, except for a few confidantes who received his nod. The company is vastly bigger now, and slightly more colorful, in a muted sort of way; but it is still the stiff and deadly earnest battalion of his dream.

Because of Watson's background as salesman, he made Sales the apex of the corporation. The salesmen had the most prestige within the company and

could make the most money; below that was administration, below that, technical staff.

Watson eliminated the meat-slicing machines, and pushed the product line based on punched cards developed by IBM's first chief engineer, Herman Hollerith. According to Rodgers, it was impetus from the Depression, and the new bookkeeping requirements of Roosevelt's remedies, that skyrocketed the firm uniquely during the depths of general economic catastrophe, till Watson came to draw the highest salary of any man in the nation. In 1934 his income was \$364,423 (Will Rogers, not the author of *Think*, was second with \$324,314). Watson had neatly arranged to get 5% of IBM's net profit.

While IBM participated in the creation of certain early computers, it is interesting that Watson dismissed Eckert and Mauchly when they came around after World War II trying to get backing for their ENIAC design, in certain ways the first true electronic computer. Eckert and Mauchly went to Remington Rand, and the resulting Univac was the first commercial computer.

However, IBM bounced back very well. If there was one thing they knew how to do it was sell, and when they brought out their computers it was practically clear sailing. (The Univac I was the first of many computers to be delayed and boggled in the completion of its software, and this considerable setback helped IBM get the lead very quickly; they have never lost it since.)

In the early sixties, the IBM 7090 and 7094 were virtually unchallenged as the leading scientific computers of the country. But IBM in the late sixties almost relinquished the fields of very big computers and time-sharing to other companies, and their computers are not regarded as innovative. Nevertheless, IBM's Systems 360 and 370, despite various criticisms, have been very successful; thousands of them are in operation around the globe, far more than all their rivals' big computers all put together. This despite the fact that some of these systems have failed, including the big Model 91 (an economic failure) and the TSS/360 time-sharing program, a technical catastrophe.

They have from time to time been accused of unfair tactics, and various antitrust and other actions (see "Legal Milestones," p. 136) have required IBM to change its arrangements in various ways. One decree required them to sell the computers that before they had only rented; another decision, to "unbun-



But the spiritual community of IBM extends further. Upper-management types, especially Chairmen of Boards and comptrollers, seem to have a reverence for IBM that is not of this world, some amalgamated vision which entwines images of eternal stock and dividend growth with an idealized notion of management efficiency. Many others use and live with IBM's equipment, and view IBM as anything from "the greatest company in the world" to "a fact of life" or even "a necessary evil." In some places whole colonies of users mold themselves in its image, so that around IBM computers there are many "little IBMs," full of people who imitate the personalities and style of IBM people. (RCA, before its computer operation fell to pieces, imitated not just the design of IBM's 360 computer, but a whole range of titles and departmental names from out of IBM. The sincerest form of flattery.)

**A LITTLE GEM FROM THE IBM SONGBOOK**

(Who says IBM doesn't encourage individualism?

To the tune of "Pack Up Your Troubles in Your Old Kit Bag.")

**"TO THOMAS J. WATSON, President, IBM"**

**Pack up your troubles—Mr. Watson's here!**

**And smile, smile, smile.**

**He is the genius in our IBM**

**He's the man worth while.**

**He's inspiring all the time,**

**And very versatile—oh!**

**He is our strong and able President!**

**His smile's worth while.**

**"Great organizer and a friend so true"**

**Say all we boys.**

**Ever he thinks of things to say and do**

**To increase our joys.**

**He is building every day**

**In his outstanding style—so**

**Pack up your troubles, Mr. Watson's here**

**And Smile—Smile—Smile.**

**SOME DIVISIONS OF IBM  
YOU MAY HEAR ABOUT**

OPD	Office Products Division. Typewriters, copiers.
DPD	Data Processing Division. Computers and accessories.
FSD	Federal Systems Division. Big government contracts: NASA stuff, and who knows what.
ASDD	Advanced Systems Development Division. Very secret.
Components Division	Makes parts for the other guys, including integrated circuits.
SRA	Science Research Associates, Chicago. Publishes textbooks and learning kits.
Watson Lab	T.J. Watson Research Laboratory, Westchester County, north of New York City. Theoretical and lookahead research.

dle," or sell computers separately from their programs (previously "given" away with the computers they ran on), is widely believed to have prevented government action on the same matter. Showing characteristic finesse, IBM thereupon lowered the computer prices almost imperceptibly, then slapped heavy price-tags on the programs that had previously been free.

Some improvement in IBM's wares came from the newer General Systems Division, founded in 1974. They built smaller computer systems (but mostly time-shared) which helped many people create a simpler information environment than was possible in a firm that had mainframes. These included Systems 3, 34, 38, 36, 1 and 7.

Meanwhile, though, the MFs got bigger and bigger. What was a mainframe in the sixties some called a "mini" by the late seventies. The line of biggies IBM called "303X" was announced in 1977. The 4300, a new name for the same sort of thing, came in 1979. Interaction on these machines is somewhat more tolerable due to incremental improvements in the operating system. (But real interaction, character by character, still is impossible. See pp. DM 8-12.)

Then, like mice, the personal computers arrived. Despite the efforts of the computer center politicians, they came through the windows, over the transoms. And people who brought them in, and could whomp up applications quickly on them, made the computer centers look foolish, which they are.

In response to this, IBM brought out the PC, an uninteresting but reasonable desktop computer.

With its open architecture, the PC permitted add-in accessories and software from all comers—a great departure from IBM's old policy. This plus IBM's clout and fame made it the world's standard desktop computer. In the face of many imitators or "clones," however, they have lost control of the market, and are struggling to regain a leadership position. They are threatened also by Apple's Macintosh initiatives, which have created a direction of market development unfamiliar to them.

The firm is now involved in partnership with, or part-ownership of, Aetna and MCI (satellites), Intel (processor chips), Rolm (telephones). They are probing a lot of related initiatives.

IBM's future, however, is definitely clouded.

**Important Current Divisions Of IBM**

*IBM's organizational structure is in constant change; this being one of the ways it adapts to a changing world and business mix. Hottest politics currently among:*

*ESD, Entry Systems Division, makers of the PC. (I don't know if they mean it's for users just entering the computer field, or for machines that they regard as really just for Data Entry. Quite possibly the linguistic question has not been settled, or even noticed.) This division is, inside IBM, the new kid on the block, fresh from extraordinary successes. Thus it has both clout and an ambiguous position. ESD is friend to the computer stores.*

*NAD, National Accounts Division, which caters to IBM's several thousand heaviest corporate customers. Big customers order PCs through NAD. This division is not friendly to the computer stores.*

*CPD, Communication Products (or Compatibility Prevention) Division. This is where they think up and enforce the twisted and complex communication protocols, like SNA and SDLC, that (seemingly intentionally) make it hard to communicate between IBM systems and more rationally-designed setups. (Some have alleged that IBM's local-area nets are better understood as political moves to defeat possible compatibility than attempts to develop a decent product.)*



But outside this pale—beyond the spiritual community of IBM—there are quite a few other computer people. Some simply ignore IBM, being concerned with their own stuff. Some like IBM but happen to be elsewhere. Others dislike or hate IBM for a variety of reasons, business and social. And this smoldering hatred is surely far different in character from anybody's attitude toward Kodak or GM.

While it is not the intent here to do any kind of an anti-IBM number, it is nevertheless necessary to attempt to round out the one-sided picture that is projected outside the computer world. In what follows there is no room to try to give a balanced picture. Because IBM can speak for itself, and does so with many voices, it is more important to indicate here the kinds of criticisms which are commonly

made of IBM by sophisticated people within the industry, so that IBM-worshippers will have some idea of what bothers people. But of course no attempt can be made here to judge these matters; this is just intended as source material for concerned citizens.

## 1. SOCIAL ASPECTS OF IBM

It is perhaps in the social realm, in-

cluding its ideological character, that a lot of people are turned off by IBM.

IBM has traditionally been the paternalistic corporation. (Paternalistic corporations were some kind of big philosophical issue to people in the fifties, but nobody cares anymore. Anyway, the rest were perhaps inconsequential compared to IBM.) Big IBM towns not only have a Country Club (no booze), but a Homestead for the comfort of important corporate guests. There are dress codes (although non-white shirts and below-the-collar hair are now allowed), and yes, codes of private behavior (now subdued). These irritate people with libertarian concerns. They do not bother employees, evidently, because employees knew what they were getting into.

Generalizations about IBM people obviously cannot be very strong. Obviously there is going to be immense variation among 265,000 people, half of whom have college degrees; but of course one of the great truths of sociology is that any non-random group has tendencies.

More than that in this case. In a way IBM people are an ethnic group. Impressive indeed are the general energy and singlemindedness of the people, galvanized by their certainty that IBM is true, good and right, and that the IBM way is the way. The righteousness is of course a big turnoff for a lot of people. Perhaps it leads in turn to the most-heard slurs about IBM people, that they are brainwashed or provincial.

## PROVINCIAL?

There would seem to be no question that IBM people are comparatively conservative and conventional. This partly because that's who IBM hires (though they reportedly urge tolerance of the unusual employee in a training film, "The Wild Duck"). A huge number of IBM people never worked for anybody else; obviously this affects the perspective, like staying at one university all your life, or in one city.

It may also be that because IBM places such a premium on dependability and obedience, new ideas (and the abilities needed to generate them) naturally run into a little trouble. Some critics find among IBM people a heavy concern with conventional symbols of achievement, and (unfortunately) seeing the world

## IBM Legal Milestones

The famous Consent Decree of January 1956. (In a consent decree, an accused party admits no guilt, but agrees to behave in certain ways thereafter.) In response to a federal anti-trust suit, IBM agreed to:

- sell as well as lease its computers, and repair those owned by others;
- permit attachments to its leased computers;
- not require certain package deals;
- license various patents;
- not buy up used machines;
- and get out of the business of supplying computer services, i.e., programming and hourly rentals.

Unbundling decision, late sixties. While this was not a government action but an internal policy decision by the company, it somehow had a public-relations appearance of official compulsion. Beset by pressures from makers of look-alike machines, users of competitive equipment, and the threat of anti-trust action, IBM decided to change its policy and sell programs without computers and computers without programs. Delight amongst the industry turned to chagrin as this became recognized as a price hike.

The Telex Decision, September '73: Telex Corp. was awarded \$352,500,000 in triple damages for losses attributed to IBM's "predatory" pricing and other marketing practices.

(This was reversed in a higher court, but it may have affected IBM's later behavior.)

THIBMK, cont.	ibmocracy	rule by IBM.
	ibmopoly	65% of the market, a great ibmposition.
	ibmunity	the safety and togetherness of IBM.
	ibmpunity	judgments against IBM, if any.
	ibmocasm	the breaking up of IBM by the Justice Department.

First, the good news...

They offer many computer programs for a variety of purposes.

Now for the bad news...

Bad news: These programs are not necessarily set up the way you would want them. (But if you take the trouble to adapt to them, you'll probably never get back.)

IBM programs are also notoriously inefficient. (That way you have to use bigger machines for longer.)

**BIG BROTHER AND AUNTIE TRUST (All in the Family?)**  
The government abandoned its monumental antitrust suit against IBM in January 1982—a proceeding begun under Johnson, dropped by Reagan. While AT&T was broken up by anti-trust, IBM is still one big company. Whether you consider this ironic depends on your point of view.

stuck all over with conventional labels and Middle American stereotypes.

Some of the most amusing material on this comes from an odd source: a writer named Heywood Gould who, all unprepared, became a consultant to IBM, earned unconscionable amounts of money (\$40,000 in six months), and lived to write a very funny and observant book about it (see Bibliography, p. 146).

But it is necessary on these matters to see how difficult things can be for IBM people. To be identified as an IBM person is something like wearing a ring in your nose, a yarmulka or a halo: an entrapment in a social role that makes the individual's position awkward among outsiders. IBM people often have to take guff at parties, unless they are IBM parties. Defensiveness may account for some of the Overdo, and some of the clannishness.

**BRAINWASHED?**

It is true that IBM people are essentially in their own world. One theory is that compartmentalization within the firm (rather visible in their designs) may tend to stifle. Indeed, because IBM people can expect to be briefed and schooled in every technical matter they will need to know for a given assignment, the incentive to follow technical developments through outside magazines and societies may be reduced. Between *Think* magazine and corporate briefings, it is possible for IBM people to be comparatively (or even completely) unaware of innovations elsewhere in the field, except as the new developments are presented to them within the organization. In this light it is easy to understand the ibmers' sense of certainty that their firm invented everything and is at the forefront.

Of course many fine research efforts do go on there, in considerable awareness of what's happening elsewhere. Particular individuals at IBM have done excellent research on everything from computer hidden-line imaging to the structure of the genetic code and computer-synthesized holograms. APL itself (see p. 58), as developed by Iverson at Harvard and later programmed by him at IBM, is another example of sophisticated individual creativity there. So it's entirely possible. But IBM certainly has no monopoly on understanding or creativity, and IBM-haters sometimes talk as if the reverse is true. ☹

**IBM In "Modern Times"**

*In the old days, an emphasis on status and authority was central to IBM's advertising in the public presses. To quote the original edition of Computer Lib:*

An interesting aspect of IBM publicity is its stress on status. Publicity photographs often show a subordinate seeking advice from a superior. IBM ads appeal to the corporation president in all of us—either Going It Alone (taking a long walk over an Executive Decision) or soberly directing a lesser employee. In one extraordinary case, we saw worshipful convicts at the feet of a Teacher implausibly situated in the corner of a prison yard.

*That sort of advertising couldn't go on, though. Somewhere around the time they brought out the PC, IBM realized they had to begin dealing with a clientele that wasn't necessarily awed by, or receptive to, authority. And what they did was masterful.*

*They chose Charlie Chaplin's Little Tramp as their new trademark (licensed by Bubbles Inc. Ltd. in Europe).*

*IBM symbolized by the underdog!*

*And they have the temerity to allude in their commercials to "Modern Times"! If you remember Chaplin's film of that title, it showed a mechanized and oppressive assembly-line factory that many would find representative of IBM's corporate style.*

**"When we went from IBM to National Cash Register, it was like the difference between night and day."**

**Retired hardware executive, talking about inventory program.**

One of IBM's more recent tricks is to overwhelm litigators by the quantity of documents supplied, many of which are stored on computers in full-text form. To give you an idea of the humongous magnitudes involved, some figures came up in litigation with Sanders Associates. Sanders, suing IBM, asked IBM how many documents IBM had that were 'pertinent' to the case. The reply: "Active files, approximately 906,054,000 pages; inactive files, approximately 421,660,000 pages." (*Datamation*, July '75, p. 129.)

An unfortunate aftermath of a suit by Control Data, in which IBM settled, was the destruction of the great indexes which had been constructed to the vast file of IBM's records; the index is gone and unavailable.

*Were it not for that splendidly-executed maneuver, the antitrust proceeding against IBM might have turned out quite differently.*

*"IBM is run by and for people who really believe in authority. IBM is, to my way of thinking, the way the Soviet Union would be if the Soviet Union worked."*

*Anon.*



# IBM'S CONTROL:

## THE VIRTUAL MECHANICS

IBM controls the industry principally by controlling its customers. Through various mechanisms, it seeks to enforce the principle that "Once an IBM customer, always an IBM customer." With an extraordinary degree of control, surely possessed in no other field by any other organization in the free world, it dictates what its customers may buy, and what they may do with what they get. More than this: the exactions of loyalty levied upon IBM's customers are similar, in kind and degree, to what it demands of its own employees. IBM makes the customer's employees more and more like its own employees, committing them as individuals, and effectively committing the company that buys from it, to IBM service in perpetuity.

Here are some of the ways this system of control seems to work. Even if these are really the mechanics, perhaps IBM doesn't mean them to be. It might just somehow be a continuous accident.

### A. Interconnection and compatibilities

IBM acts as if it does not want competitors to be able to connect their accessories to its computers. It's as though GM could design the roads so as to prevent the passage of other vehicles than its own.

This is done several ways. First, IBM has sometimes used contractual techniques to prevent such interconnections to its systems, either forbidding other things to be attached (or at least slapping on extra service charges if they are), or declaring that it would not be responsible for overall performance of such a setup, effectively withdrawing the hardware guarantee that is such a strong selling point.

Secondly, IBM does not tell all that needs to be known in order to make these interconnections—the details of the hardware interfaces.

Finally, IBM can simply decree, perhaps claiming technical necessity,

that interconnection is impossible. For instance, IBM said for a time that their big program, "VS," or Virtual System, wouldn't work (translation: would not be allowed) if competitive memories were used on the computer.

Now, there are many manufacturers who think this is very wrong of IBM; who believe they should have the right to sell accessories and parts—especially core and disk memories—to plug onto IBM's computers. It has been generally possible for these other manufacturers to work these interconnections out awhile after the computer comes out on the market, but it's getting more difficult.

Thus the Telex Decision of September 17, 1973, in which it was decreed by the judge that IBM would have to supply complete interface information promptly when introducing a new computer, was a source of great jubilation in the computer field. However, that part of the judgement has since been cancelled.

Much the same problem exists in the software area. IBM is less than interested in helping its competitors write programs that hook up to IBM programs, so the details of program hookup are not always made clear. Here, too, many smaller companies insist they should be made to do it.

### B. Control and guidance of what the customer can get

To a remarkable degree, if you are an IBM customer, you practically have to buy what they tell you. This IBM manages by an intricate system of fluctuating degrees of sales and support and contractual dealing. The IBM customer always has several options; but these are like forced cards. IBM is always introducing and discontinuing products, and changing prices and contractual arrangements and software options in an elaborate choreography, which applies calculated pressures on the customer. IBM has a finely-tuned system of customer incentives by which it controls product phasing, to use the polite term, or planned obsolescence, as some people call it.

Programs, especially, are available with different degrees of approval

from IBM. The technique of "support" is the concrete manifestation of approval. A supported program is one which IBM promises to fix when bugs turn up. With an unsupported program, you're on your own, God has forgotten you. Because so much of IBM's virtue lies in the strength and fervor of its support, the use of unsupported programs, or unsupported features of supported programs, is a difficult and risky matter, like driving without a map and a spare tire, or even going into the Himalayas without gloves. Effectively, the withdrawal of support is the death knell of any big program, such as TSS/360, even though customers may want to go on using them.

Availability of products is in general a matter of exquisite degree. It's not so much that you can or can't get a particular thing, but that the pricing and available contracts at a given time exert strong pressure to put you where they have chosen within their currently featured product line. Moreover, extremely strong hints are always available; the salesman will tell you what model of their computers is likely to be a dead end, or, on the other hand, what model is likely to offer various options and progressive developments in the near future.

Some things are half-available, either as "RPQs" (an IBM term for special orders—Request Price Quotation), or available to sophisticated customers at IBM's discretion.

With all the degrees of availability, it is easy for IBM to open or close by degrees various avenues in which customers are interested.

*An amusing variant is to offer apparent compatibility that is only partial. For instance, IBM recently brought out a new machine that combines the System 36 and the PC, the System/36 PC. For it you can obtain software packages also offered for the System 36, and run them on this guy in System 36 mode; but you can't move them back to your System 36, because they're different underneath. This is called by the press "portability control." (MIS Week, 26 June 85, 18.)*

Also, different sizes of computer will or won't allow given programs or

desirable program features. Many IBM customers have to get bigger computers than they would otherwise want because a given program—for instance, a COBOL compiler with certain capabilities—is not offered by IBM for the smaller machine. Indeed, an elaborate sizing scheme exists for matching the machine to the customer—or, a cynic might say, assuring that you can't get the program features you ought to be able to get unless you get a larger computer than you wanted.

What it boils down to is that you, the customer, have few genuine options, especially if your firm is already committed to doing certain things with a computer. And when IBM brings out a new computer, the prices and other influences are exactly calculated to make mandatory the jump they have in mind to the new model.

### C. Having to do things just their way

IBM systems and programs are set up to do things in particular ways. To a remarkable degree, it is difficult to use them in ways not planned or approved by IBM, and difficult to tie systems and programs together. Programs and features which the casual observer would suppose ought to be compatible, tend not to be so. For some reason compatibility always tends to cost extra. It is as though the compatibility of equipment and programs were planned by IBM as much as their product line.

Effectively the IBM customer tends to be frequently trapped in a cage of restrictions, whether this cage is intentionally created by IBM or not. One is reminded of the motto of T.H. White's anthill in *The Once and Future King*:

### THAT WHICH IS NOT FORBIDDEN IS COMPULSORY

The degree to which these restrictions are manipulated or intentional is, of course, a matter of debate.

### D. Captive bureaucracies running in place?

Perhaps the most unfortunate thing about IBM (from an outsider's



point of view) is that effectively their systems can only be used by bureaucracy whom they have trained. From keypunch operator up to installation manager, all are effectively enslaved to curious complexities that keep changing. The ever-changing structure of OS, and its quaint access methods, is just one example. It might even seem to the outside observer that IBM's game, intentional or not, is to keep things difficult and intricately fluid to retain utter control. In other words, it is as though they fostered a continual turnover of unnecessary complications to keep a captive bureaucracy running in place. People whom they have indoctrinated tend not to buy opponents' computers. People who are immersed in the peculiarities of IBM systems, and busy keeping up with mandatory changes, do not get uppity. They are too busy, and the investment of their time and effort is too high for them to want to change.

Anti-IBM cynics say that a lot of the work involved in working with IBM computers is self-generated, arising from the unnecessary complexities of OS/360, JCL, TCAM and so on. But of course that cannot be evaluated here.

## PROSPECTS

These remarks should clarify the bleakness of the prospect for man's future among computers if IBM's system of control really does work this way, and if it is going to go on doing so. Because it means the future that some of us hope for—the simple and casual availability to individuals of clear and simple computer systems with extraneous complications edited away—may be foreclosed if they can help it.

Let's all hope, then, that these things turn out not to be really true.

*Times have changed. The above remarks no longer apply at the low end. When it introduced the PC, IBM opened that system fully to interconnection, software and imitation—knowing that was now the way to an industry-sweeping product.*

*The mainframe world, however, has not changed very much.*

## 2. SALES TECHNIQUES

It is IBM's alleged misbehavior in pursuit of sales that has drawn some of the strongest criticism within the industry, as well as considerable litigation. Their "predatory pricing" (a term used by the judge in the recent Telex decision), and other mean practices, are (whether true or false) folklore within the industry.

These accusations are well summarized by "Anonymous" in a recent article (see Bibliography, p. 146). Basically the accusations against IBM's sales practices are that they play dirty: if you, say, the computer manager in a business firm, want to buy equipment from another outfit, IBM (so the story goes) will go over your head to your boss, accuse you of incompetence, try to get you fired if you oppose them, and Heaven knows what else. Anonymous claims that various forms of threat, intimidation, "hard-sell scare tac-

tics" and "behind-the-scenes manipulation" are actually standard practice in IBM sales; he or she alleges various instances in certain municipalities.

Such behavior is emphatically denied, though not in relation to that article, by Board Chairman Cary, in a recent letter to *Newsweek* (see Bibliography). Cary emphasizes the importance of IBM's 76-page Business Conduct Guidelines. Whether these are publicly examinable is not stated.

An interesting view on IBM's sales ethics was expressed by Ryal R. Poppa, president of Pertec Corp.

"In the past, when there have been sales situations where 'you can't honor the policy and win the deal,' IBM has violated the policy with the practice," he said.

However, he believes that situation is changing under IBM's new management,

so that the guidelines will be observed in the future. ("Poppa Sees Several IBM Changes," *Computerworld*, 21 Nov. '73, 29.)

The people who take these matters of IBM sales practices most seriously—IBM's competitors—now have their own organization, the Computer Industry Association. This is an association of computer companies, which has as its intention the "establishment and preservation of a sound and viable U.S. computer industry, based on... free and open competition." Emphasis theirs. Translation: they're out to get IBM. President Dan L. McGurk, formerly of Xerox Data Systems, has blood in his eye. Membership is open only to computer companies, but their newsletter *On Line* is available to individuals (see Bibliography). Anyone seriously interested in these matters is referred to them. ■

**A persistent rumor is that IBM fires all of its salesmen in a geographic area if a key or prestige sale is "lost," as when M.I.T.'s Project MAC switched over to General Electric computers in the sixties, or when Western Electric Engineering Research Center passed over IBM computers to get a big PDP-10.**

**Much as some people would like to believe these stories, there seems to be no documentation. You would think one such victim would write an article about it if it were true.**

*The same slick marketing could be applied to any other industry. But it wouldn't be IBM. Nowhere else could the mystery of the subject be met and enhanced with so many more mysteries.*

**First, the good news...**

**Good News:** A company or governmental agency can get immense amounts of "help" and "information" from IBM, which offers free courses, even IBM people on "released time" to look over the problems on the premises.

**Now for the bad news...**

**Bad News:** The courses indoctrinate with the IBM outlook, and the planted people spread it. Moreover, both mechanisms help IBM spot the people they can work with to make a big sale—and (it is alleged by some) those who stand in the way.

## THE ETHICS MANUAL

*IBM has a famous "ethics manual." I read it once.*

*It seemed to me to be a set of guidelines for exactly what sales practices were and were not allowed, written so as to help the salesman exactly skirt the forbidden territory while working as close to it as possible.*

*Which is rational and honest. But somewhat unsettling. And not so much about ethics as about hard borderlines for a dangerous game.*



### 3. TECHNICAL DECISIONS AND DESIGNS

#### A. Prologue

Part of the myth of IBM's corporate perfection is based on the notion that technical matters somehow predominate in IBM's decisions, and that IBM's product offerings and designs thus emerge naturally and necessarily and inevitably from these considerations. This is rather far from the truth.

IBM presents many of their actions as technical, even as technical breakthroughs, when in fact they are strategic maneuvers. The announcement of a new computer, for example, such as the 360 or 370, is usually made to sound as if they have invented something special, while in

fact they have simply made certain decisions as to "which way they intend to go" and how they plan to market things in the next few years.

An interesting example of an IBM non-breakthrough was the dramatic announcement in 1964 of the 360 computer, portrayed as a machine which would at last combine the functions of both "business" computers and "scientific" computers. But other companies, such as Burroughs (with the 5500) had been doing this for some time. The quaint separation of powers between scientific computers (with all-binary storage of numbers) and business computers (decimal storage) was based only on tradition and marketing considerations, and was otherwise undesirable. In amalgamating

the "two types," IBM was only rescinding their own previous unnecessary distinction. The drama of the announcement derived in large measure from the stress they had previously laid on the division. (*Fortune* ran an interesting piece on the decision struggles preceding the introduction of the 360 computer, and the internal arguments as to whether there should be one line of computers or two. See the five-billion-dollar gamble piece, Bibliography, p. 146.)

This ties in closely with another interesting aspect of the IBM image, the public notion that IBM is a great innovator, bringing out novel technologies all the time. It is well known in the field that they are not: IBM usually does not bring out a new type of product until some other

company has pioneered it. (Again remember the earlier point, that the product offering is a strategic maneuver.) But of course such facts do not appear in the promotional literature, nor are they volunteered by the salesman.

The expression for this in the field is that IBM "makes things respectable." That is, customers get that reassured feeling, when IBM adds other people's innovations to their product line, and decide it's okay to go ahead and rent or buy such a product. (This also sometimes kicks business back to the original manufacturer.)

A few examples of things that were already on the market when IBM brought them out, often making them sound completely new: transistorized computers (first offered by Philco), virtual memory (Burroughs), microprogramming (introduced commercially by Bunker-Ramo).

This is not to say that IBM is incapable of innovation: merely that they are never in a hurry about it. The introduction of IBM products is orchestrated like a military campaign, and what IBM brings out is always a carefully-planned, profit-oriented step intended not to dislocate its product line. This is not to say that they don't have new stuff in the back room, a potential arsenal of surprises of many types. But it is probable that most of them will never be seen. This is because of IBM's "impact" problem.

Unique in IBM's position is the problem of fitting new products into the market alongside its old ones. Its problem is much worse, say, than that of Procter & Gamble. The problem is not merely its size and the diversity of its products, but the fact that they may interfere with each other ("impact" each other, they say) in very complicated ways.

*The problem facing IBM now is its immensely successful PC line. If we think of the PC as a small boat, then the mainframe computer center is like a Roman trireme, a complex operation requiring complex cooperation, suffering, regimentation and great effort. But now both the galley-slaves and the cargo—the users and their programs—are escaping overboard in lifeboats. If there's a way to get your own computer, who wants to put up with the computer center anymore? (See "Computer Center Politics," p. 61.) So IBM's problem is how to save its mainframe business as many former prisoners*

---

### THE SUPPOSED STANDARDIZATION OF IBM'S MAINFRAMES

---

*Before 1964, IBM had several mainframe lines. In 1964, these were consolidated into a single computer series. (See "The 360," p. 88.) While this line of computers has constantly changed and been renamed, and some of the machines are now far bigger (the Sierra) there has been little basic change in its architecture.*

IBM announced a number of worthy objectives when the 360 line was announced in 1964. IBM should certainly be thanked for at least their lip service to these noble goals.

1. 'One machine for all purposes, business and scientific.' (Thus the name "360," for the "full circle" of applications.) By "business" this mainly meant decimal, at four bits a digit. Actually this meant grafting a 4-bit decimal hardware to an otherwise normal binary computer, and making both types of users share the same facility.

2. 'Information storage and transmission will be standardized.' The 360 was set up to handle information 4 bits at a time, 8 bits at a time, 16, 32, and 64 bits at a time. (The preceding standard had been 6, 18, and 36 bits at a time.)

In their 360 line, IBM also replaced the industry's standard ASCII code with a strange alphabetical code called EBCDIC ("Extended Binary Coded Decimal Information Code"), ostensibly built from the 4-bit decimal code (BCD), but believed by cynics to have been created chiefly to make the 360 incompatible with other systems and terminals.

3. '360s will all look alike to the program; thus programs can be moved freely from machine to machine.'

Unfortunately this compatibility has been undermined by numerous factors, especially the variety of operating systems, including half a dozen major types, and the lan-

guage processors, intricately graded according to computer size. Both these factors tend to make changes necessary to move programs between computers. While one effect of this "standardization" has indeed been to facilitate the moving of programs from small computers to big ones, a more important effect has perhaps been to make it hard to move from a big computer to a smaller one. Note the usefulness of this apparent paradox to IBM's marketing.

The secret of it all, of course, lies in IBM's keen understanding of how to sell big computers. The comptroller, or somebody like him, generally makes the final decision; and if he is told that the one computer will run "all kinds" of programs, that naturally sounds like a savings. Shades of the F-111. (Businessmen's trust and respect for IBM is discussed elsewhere in this article.)

*Nevertheless, IBM was able to impose this mess on the world by its Grand Reputation and its selling powers and practices.*

*The one good thing about the 360 series was the standardization of 8 bits as the world's storage and transmission standard. This we now call a byte. (Previously a "byte" had meant any length adopted for a programmer's convenience on, say, the PDP-10.)*

*Some might argue that this 8-bit standard was not important. I think it was. The 8-bit standard has, in its turn, made it possible to have standard textfiles and word processing, standard memory chips, standard modems, standard busses.*

*The way I've heard it, this 8-bit standard, now thought of as central to the 360, was not central at the beginning, but was largely accomplished through the politicking of a few idealistic and dedicated individuals like Bob Bemer.*



and applications let loose from their cages, swarm across the deck and leap overboard.

These are no small matters: the danger is that some new combination of products will save the customers money IBM would otherwise be getting. **Innovations must expand the amount IBM is taking in, or IBM loses by making them.**

These complications of the product line in a way provide a counterbalance to IBM's fearsome power. The corporation has an immense inertia based on its existing product line and customer base, and on ways of thinking which have been carefully promulgated and explained throughout its huge ranks, that cannot be revised quickly or flipantly.

Nevertheless it is remarkable how at every turn—notably when people think IBM will be set back—they manage to make policy decisions or strategic moves which further consolidate their position. Often these seem to involve restricting the way their computers will be used (see "IBM's Control" nearby.)

(The most ironic such countermove by IBM occurred a few years ago with the so-called "unbundling" decision. IBM at last agreed (on complaint from other software firms) to stop giving its programs away to people renting the hardware. Glee was widespread in the industry, which expected IBM to lower computer prices in proportion to what it would now charge for the software. Not at all. IBM lowered its computer prices by a minuscule amount and slapped heavy new prices on the software—often charges of thousands of dollars per month.)

Finally, there is the popular doctrine of IBM's infallibility. This, too, is a way from the truth. The most conspicuous example was something called TSS/360.

TSS/360 was a time-sharing system—that is, the control program to govern one model of the 360 as a time-sharing computer. According to *Datamation* ("IBM Phases Out Work on Showcase TSS Effort," Sept. 1, 1971, pp. 58-59), over 400 people worked on it at once for a total of some 200 man-years of effort. And it was scrapped, a writeoff of some 100 million dollars in lost development costs. The system never worked well enough. Reputedly users had to wait much too long for the computer's responses, and the system could not really compete with those offered elsewhere.

The failure and abandonment of this program is thus responsible for IBM's present non-competitive position in time-sharing; customers are now assured by IBM that other things are more important. IBM-haters thank their stars that this happened. Cynics think it conceivable that high-power time-sharing was dropped by IBM in order to shoo its customer base toward areas it controlled more completely.

*IBM time-sharing has improved incrementally in the last decade, but customers tend strongly to prefer the high interaction in small PCs to lurching interaction with the supposedly opulent facilities of the mainframe computer center.*

*The reason I spent so much space on IBM, originally, was to explain to neophytes why the outer world's adulation of IBM is so misplaced, and why IBM was in many ways the opponent of, rather than the bringer of, computer freedom and progress.*

*Now that has changed a lot, at least wherever there are IBM PCs or their clones, for these at least and at last permit high interaction and unfettered use.*

*But the mainframe world and its computer centers are still the strange captives of this company. So are all those who must use these computer centers—individuals (many of whom are innocent and suffering), departments (many of whom are innocent and suffering), and clients of these departments—such as welfare recipients and credit card holders and legislators, many of whom are innocent and suffering.*

*So it's still important to understand this captivity system—in our roles as citizens as well as computerfolk.*

EVERYTHING  
IS DEEPLY  
INTERTWINGLED®

Two other conspicuous IBM catastrophes have been specific computers: the 360 model 90 in the late sixties, and a machine called the STRETCH somewhat earlier. Both of these machines worked and were delivered to customers. (Indeed, the STRETCH is said by some to have been one of the best machines ever.) But they were discontinued by IBM as not sufficiently profitable. Therein is said to have been the "failure." (However, it has been alleged in court cases that these were "knockout" machines designed to clobber the competition at a planned loss.)

*"...this is the rock-solid principle on which the whole of the Corporation's Galaxywide success is founded—their fundamental design flaws are completely hidden by their superficial design flaws."*  
Douglas Adams, So Long, And Thanks For All The Fish, Harmony Books, 1985

## THIBMK, cont.

ibmologism	clumsy or inappropriate term, esp. one which misspeaks itself, such as Direct Access Device for indirectly accessible device.
ibmroglio	IBM software.
ibmconvenient	the way IBM wants it and you may not.

### Q. What's an IBM man-year?

A. Seven hundred guys trying to finish something before lunch.

*Note that this joke is the heart of Fred Brooks' book, The Mythical Man-Month (Addison-Wesley, 1974). Brooks was the leader of the team that produced IBM's horrendous operating system OS, and this book examines that failure—although, from IBM's point of view, the clumsiness, difficulty and oppressiveness of OS, and the way it affects us all—especially in corporations and government offices—simply generates more revenue for IBM, requiring bigger computers to run longer and be tended by more operators and systems analysts.*



## B. Negative views of IBM systems

In the technical realm, IBM is widely unloved because many people think some or all of their computers and programs are either poor, or far from what they should be. The reasons vary.

Some of the people feeling this way are IBM *customers*, and for a time they had an organized lobby, called SHARE (which also facilitated sharing of programs). Recently, however, SHARE has become IBM-dominated, a sort of company union, according to my sources.

The design of the 360, while widely accepted as a fact of life, is sharply criticized by many. (See "What's wrong with the 360?," p. 88.)

IBM's programs, while they are available for a broad variety of purposes, are often notoriously cumbersome, awkward and inefficient, and sometimes dovetail

very badly. However, the less efficient a program is, the more money they make from it. A program that has to be run for an hour generates twice as much revenue than if it did its work in thirty minutes; a program that has to be run on a computer with, say, a million spaces of core memory generates five times the revenue it would in two hundred thousand.

IBM programs are often thought to be rigid and restrictive.

The complex training and restrictions that go with IBM programs seem to have interesting functions. (See "IBM's Control," p. 138.)

## C. Theories of IBM design

**The question is, how could a company like IBM create anything like the 360 (with its severe deficiencies) and its operating system or control program**

**OS (with its sprawling complications, not present in competitors' systems)? Three answers are widely proposed: On Purpose (the conspiracy theory), By Accident (the blunder theory), and That's How They're Set Up (the Management Science theory). These views are by no means mutually exclusive.**

The Management Science theory of IBM design is the only one of these we need take up.

The extensive use of group discussion and committee decisions may tend to create awkward design compromises with a certain intrinsic aimlessness, rather than incisively distinct and simple structures. (See Gould's marvelous chapter, "The Meeting," pp. 58-80.)

Their use of immense teams to do big programming jobs, rather than highly motivated and especially talented groups,

is widely viewed as counterproductive. For instance, Barnet A. Wolff, in a letter to *Datamation* (Sept. 1, '71, p. 13) says a particular program

"remains inefficient, probably because of IBM's unfortunate habit of using trainees fresh out of school to write their systems code."

There may also be something in the way that projects are initiated and laid out from the top down, rather than acquiring direction from knowledgeable people at the technical level, that creates a tendency toward perfunctoriness and clunky structure.

Thus there may very well be no intentional policy of unnecessary complication (see "IBM's Control," p. 138). But the way in which goals are set and technical decisions delegated may generate this unnecessary complication.

(In this light it is not hard to understand IBM's stand on software copyrights vs. patents. IBM is against programs being patentable, which would cover abstracted properties, but argues in favor of copyright, whose protection is probably more limited to the particulars of a given program. If they have their way, it would be assured that IBM could use any ingenious new programming tricks without compensation, whereas all unnecessary complications of bulky, cumbersome software would be covered in entirety by copyright.)

Finally, it has not been demonstrated that IBM has any general ability to make systems conceptually simple and easy to use. There seems to be no emphasis on elegance or conceptual simplicity at IBM. Those who adopt such a philosophy (such as Kenneth Iverson) do so on their own.

As mentioned earlier, this has something to do with the fact that individuals generally use IBM's systems because they have to, being employees or clients of the firms that rent IBM equipment, so there is no impetus to design programs or systems to run on simple or clear-minded principles, or dress out intricate systems so they can be used easily.

(This has changed somewhat in the software for the PC, where IBM has discovered that ease of use—when there is competition—sells. Here, however, their strategy is to acquire from outside, since user ease generally is not well understood within the organization.)

I hope to be able to report in the future editions of this book that IBM has moved firmly and credibly toward making its systems clear and simple to use, without requiring laborious attention to needless complications and oppressive rituals. It's still possible.

One of the things we often forget is that public-spirited corporations can be reached, they do listen; and IBM is nothing if not public-spirited—except when it comes to the design of its systems.

YAY!

*It has indeed happened.*

*There have been several threads of improvement.*

1. In office products and its typewriter line, IBM has often had clean and simple design. Not so much so in its magic typewriters and word processors, but sometimes. (Although for years they wouldn't let you tie them to computers.)

2. In 1973 IBM created the General Systems Division, which went about creating allegedly easy-to-use systems for smaller businesses. System 32 came out the following year.

Astounding as it may be from the company that gave the world JCL and the MT/ST, in January IBM stepped into the world of easy computers, bringing out the System/32, a minicomputer for business. You can only rent it as an interactive terminal, with a program created by IBM which cannot be modified (called an Industry Application Package or IAP). But these little programs prompt users step-by-step through what they are supposed to be doing, and apparently are very clear and helpful for the naive.

This about-face is in many ways gratifying for those of us who have been advocating easy, screen-based systems for years and years. At long last it gives IBM's "legitimacy" to minicomputers for business, and it helps companies that already provide such services.

It will be interesting to see if IBM knows how to make things simple, considering the experience they have lavished on the opposite policy.

*General Systems Division's stuff has been successful; much of it is okay.*

3. In the personal computer world, IBM brought out a machine—the PC—that was a reasonable step up from the Apple II, with open architecture, and proceeded to invite in small software and hardware vendors from all over, including many whose concern was ease of use.

And they didn't question the color of their shirts, either; indeed, the great and beloved John Draper, formerly Captain Crunch the Phone Phreak (and considered by some to be computerdom's maddest madman), produced EasyWriter, which went out baptized as an Official IBM Product.

*The result of the PC has been a great flowering of new ideas, programs and accessories. And an immense availability of computing power at lower and lower prices. (As little as \$595 will get you a PC, at this writing, though of course not from IBM.)*

As usual, IBM goofed on the operating system. They obtained from the outside an operating system which is far too complex and dangerous for 95% of the population. This means that most PC owners must stay huddled in the few applications and commands they understand. Still, the PC is a considerable step forward.

#### 4. THE IMAGE

It is hard to analyze images, corporate or personal. They are often received in such different ways by different populations. But there may be a commonality to the IBM image as generally seen. The Image of IBM involves some kind of cold magic, a brooding sense of sterile efficiency. But other things are percolating in there. If we slide that connotation of efficiency aside, the IBM image seems to have two other principal components: authoritarianism and complacency. It is this mixture that longhairs will naturally find revolting. This same combination, however, may be exactly what it is that appeals to business-management types.

#### THE FUTURE

What will IBM do next?

Speculation is almost futile, but necessary anyhow. The prospects are fascinating if not terrifying.

No one can ever predict what IBM will do; but trying to predict IBM's actions—IBM-watching is something like Kremlin-watching—is everybody's hobby in the field. And its consequences affect everybody. With so many things possible, and determined only in the vaguest way by technical considerations, the question of what IBM chooses to do next is pretty scary. Because whatever they do we'll be stuck with. They can design our lives for the foreseeable future.

*Thank goodness that last sentence is no longer true—except inside corporations, government offices and some universities, where IBM mainframe captivity continues. But personal computing has restricted the inroads of IBM-style systems, and turned them into a holding action.*

We know that in the future IBM will announce new machines and systems, price changes (both up and down) in fascinating patterns, rearrangements of what they will "support," and changes in the contracts they offer (see "IBM's Control," p. 138). Occasional high-publicity speeches by IBM high officers will continue to be watched with great care. But mainly we don't know.

IBM's slick manufacturing capabilities mean that practically any machine they wanted to make, and put on a single chip, they could, and in a very

short time. (The grapevine has it that the Components Division, which makes the computer parts, has bragged within the company that it doesn't really need the other divisions any more—it could just put whole computers on teeny chips if it wanted to.)

IBM has conceptually consolidated its various magic-typewriter and text services under the name of "word processing," which means any handling of text that goes through their machines. This superficially unites their OPD efforts (typewriters and dictation machines) with things going on in DPD, such as Datatext, and allays inter-divisional rivalries for awhile. Also, by stressing the unity of the subject matter, it leaves the door open for later and more glamorous initiatives, such as hypertext systems (see p. DM 35).

In other words, the foot is in the door. Mr. Businessman has the idea that automatic typing and things like that are IBM's special province.

#### END OF THE DINOSAURS?

To a very great extent, IBM's computer market is based on big computers run in batch mode, under a very obtrusive operating system.

Many people are beginning to notice, though, that many things are more sensibly done on small computers than on big ones, even in companies that have big computers. That way they can be done right away rather than having to wait in line. Is this the mammal that will eat the dinosaur eggs?

On the other hand, a very unfortunate trend is beginning to appear, an implicit feud within large organizations, which may benefit IBM's big computer approach. Those who advocate mini-computers are being opposed by managers of the big computing installations, who see the minis as threatening their own power and budgets. This may for a long time hold the minis back, perhaps with the help and advice of computer salesmen who feel likewise threatened. But there will be no holding back the minis and their myriad offspring, the micro-processors (see p. 96). And the inroads should begin soon.

(Others are growing to know and love true high-capacity time-sharing as a way of life, like that offered for DEC, GE and Honeywell machines. This, too, may be-

gin to have derogatory effects on IBM's markets.)

Finally, it must be noted that almost all big companies have computers, usually IBM computers, and so an era of marketing may well have ended. It may be possible for IBM to go on selling bigger and bigger computers to the customers who already have them, but obviously this growth can no longer be exponential. ➡

ONE  
THING  
IS  
PERFECTLY  
CLEAR:

IBM has  
no monopoly  
on understanding  
or sophistication.

#### NEW CHIPS

IBM can put pretty much anything on a single chip, to make a functioning machine the size of a postage stamp; but so can a lot of other companies.

The question really becomes whether what goes on that chip is a *worthwhile* machine that does what people want.

*...BUT THE SAME OLD BLOCK?*

*It is by no means clear that IBM has any general ability to make computer systems easy to use.*

*This is a psychological problem.*

*As a corporation they are used to designing systems that people have to use by fiat, and must be trained to use, contributing to the captivity and inertia of the customer base. Thus the notion of making things deeply and conceptually straightforward, without special jargon or training, may not be a concept the company is ready for.*



# THE GOOD NEWS AND BAD NEWS ABOUT IBM (cont.)

First, the good news...

Good News: IBM offers various kinds of compatibility among its systems.

Now for the bad news...

Bad News: It always seems to cost extra.

## IBM LATELY

"It is a far, far better thing I do now than I have ever done."

—hero on his way to the guillotine in Charles Dickens' *A Tale Of Two Cities*.

IBM has achieved great redemption through the PC. The PC showed that a huge corporation that had done a lot of bad things could do one good and great thing.

But that Far Far Better Thing may have fatal consequences.

### THE FALL OF IBM

IBM is having a great fall. Its very success with the PC has put it in a somewhat precarious position.

Had there been no PC, the other dinkies and desktop computers would have eroded IBM's market more gradually. But its great success with the PC may have hastened the great reckoning. For the PC has created a grave disarray in IBM's previous selling system.

### A PATHLESS MARKET

IBM no longer has an ordered market where customers can be forever shep-

herded up specific growth paths. Everything (and particularly the PC) is getting in everything else's way, in an increasingly fluid product world.

IBM's internal problems in trying to control the market are greater than the external: they must seek to make a coherent system of products that keeps upholding the computer centers, and that keeps its different substrategies—its different divisions—from getting in each other's way.

### 1. PC VERSUS MF

Before personal computers, IBM had a system of computer centers that provided the only computer access for individuals in thousands of companies across America. Every computer initiative of every person in those companies—on their employers' behalf, remember—had to be funneled through these non-interactive monstrosities.

But the desktop machines now have legitimacy—that is, credibility to stuffy minds—and many are equivalent to the "mainframes" of the early sixties.

Customers now know they can get many applications off the mainframe and

out of the computer center—text handling and word processing, simulation, analysis, even small database stuff—and go to desktop machines. And a wholesale desertion of mainframes has begun. (Note that "wholesale" doesn't have to be a big percentage. Even a moderate drop of lost usership has a heavy impact on the local center's budget and position—and on IBM's prospects.)

And perhaps most important, **high interactivity**—character-by-character, usually—impossible on IBM mainframes—has acquired legitimacy; though not amongst the old guard and the computer centrists, who can't imagine the point of it. ("Who wants sex when we've got **Playboy**?" )

### THE THIN THREAD

Right now IBM is FIGHTING TO KEEP THE MAINFRAME NECESSARY, with all its cost and complexity and huge staff. But the mainframe-desktop connection is a thin thread, since there's very little you can't move to the smaller machines if you really want to. IBM is trying to make this thread a lasso to corral and bring back the escaping users, at least those in corporations.

### RE-SUBORDINATION

IBM is making a substantial, perhaps company-wide, effort to recapture the PCs as satellite machines, making the PC into a terminal.

Keeping as much tied in as possible, not letting the people and their programs escape to the PC, and ensnaring the PCs in the embrace of mainframes: this is now the strategy being shared by IBM and the computer center politicians out in the companies.

The trick is tying them back to mainframes with software that runs partly on each, only letting the PC have a little part of the application program at a time. "Download" and "host session" are the watchwords. (The term "host" sounds as if it were something taking care of **you**, rather than a beast that must be fed and housed and and paid for and thrown warm bodies.)

An example is the "Model G" color workstation (actually the 3270 PC/AT-G and its big brother the GX), which are PCs intended to function as color terminals to mainframes. Another example is

In the author's opinion, the issue is the one big usual question, the issue of freedom in our time; and that is not a matter of bigness, but the style of IBM's control.

Computers should make things easier in both our work and our private lives, and should help lighten our loads and enlighten our minds, clarifying the complexities of everything. Unfortunately, IBM's method of making money has a little too much to do with creating rigid and oppressive and pointlessly complex systems, fobbing them off as "scientific," and ensnaring its customers in complications by the techniques discussed on pp. 138-139.

People should be free to use computers as they ought to be used, each in his personal style regardless of his job title, amidst rushing menus of options and clarifying screen graphics, rather than each person and office worker being locked into his own "sternly allotted sandpile," as cummings put it. And that is the problem.

Only one company can force hideous complexity down the throats of its customers and the world, and make the victims think it's modernism, progress and "high technology."

In a way, IBM's setting this up has been one of the most monumental achievements of our time.

## CAN IBM PULL ANOTHER PC?

The staggering success of IBM's original PC has created a universal desktop standard, referred to in this book as just "the PC," regardless of who makes it. IBM controls that market with its usual over-50% share, but many other firms make highly competitive products (see p. 48).

But now, in my opinion, that market is pretty well saturated; anyone who wants to work anything that complicated, or who can be forced to work anything that complicated, now has a PC at home or work.

So what can IBM do for an encore, to keep making money and growing in the desktop computer world? Can they revive a sagging market with a new upgraded model? Can they decree a different standard desktop computer for the early nineties?

### UPGRADE PATHS

IBM has built its empire on the "controlled migration" of users—offering upgrades that herd the customers in specific directions. But IBM's success with upgrades to the PC line has been limited; they do not control the situation.

They brought out the AT, a bigger and faster PC, but it has not caught on to nearly the same extent.

### HARDWARE MOMENTUM

With the old PC, the stock PC, standardization took; people and companies have universally accepted this particular machine as the basic desktop standard computer. There is immense momentum for it. We have a true market in compatible clones, priced according to features and performance and intangibles; there is great momentum for the add-on boards made by the many compatible vendors.

### SOFTWARE MOMENTUM

And there are oceans of software available. (See pp. 52-57.) Programs for everything they could think of so far. Mostly incompatible with one another, yes; each program requiring days or weeks of study to use; but a lot of them. And all running on the standard configuration.

Look at it from outside IBM's perspective. A software vendor would be very dumb to write programs that would only run on IBM's new line when compatibil-

ity comes free if he writes them for the old, and puts in extra features to take advantage of the new.

What these new machines do, then, is strengthen the old standard, making the new line just one direction of upgrade (see pic., p. 49).

### A DIASPORA OF DIRECTIONS

There are now many disparate directions in which PCs go, divergent semi-standards, all with validity. We have—

1) the AT series, representing one direction of change.

2) Stretch Clones: IBM has itself created a number of extended PC clones which go all in different directions. By now there are half-a-dozen types, including the 3270 PC, the 370 AT and various graphic PCs.

3) Standard Memory Upgrade: Intel and Lotus have collaborated on a standard way of expanding PC memory past 640K, called the "Above Board" standard.

4) Upscale Competition: the North territory, the world of high-performance workstation machines, has been strongly staked out for some time. The strongest contenders are presently the Sun workstation and the Lisp machines. More will be joining this mixed bag; we shouldn't rule out Big Macintoshen, Silicon Graphics workstations, the Tektronix Smalltalk machine, even the Xerox Star. But it is not clear when IBM will appear in this market. Without a considerable graphics upgrade, PCs cannot compete in this market.

5) Graphics upgrades: PGA, EGA. But with other hot boards, especially Targa from AT&T, IBM does not control the graphics of the PC any more. You can get PC boards with graphics upgrades to any degree of beauty, sophistication and dimensionality.

IBM might well cast aside tradition and create a takeover workstation that would sweep the high-performance field as the PC did. But such a move would either have to be geared to IBM's Mainframe Imperative (diminishing its local power), or would threaten IBM even more deeply.

This variety of upgrade paths is likely to deprive any future IBM personal computer of leadership momentum as strong as that of the original PC.

## THIBMK, cont.

<b>ibmperceptible</b>	officially noticed by IBM.
<b>ibmccile</b>	someone ibmersed in, or ibmued with, the ibmage.
<b>enibmatic</b>	puzzlingly ibmish.
<b>ibmpeccunious</b>	having \$2 billion in available cash.

"The PC is segmented just the way the 360 was. How could they make the same mistake twice?"

—A designer



the System/36 PC, which takes regular PCs as *terminals*, and thus reinforces captivity to System 36 (a small mainframe, as far as this discussion is concerned).

Now, there's no reason to criticize this if it leads to better computer use and better computer experiences. (Even if it is restricted to corporate environments and beyond the reach of us folks.) But the problem is that it usually doesn't. If you need crunch for graphics, a local box does it better and with real interaction. And if you need the storage, you can get all you want for any computer.

## 2. PC VERSUS WP

Before personal computers, IBM controlled much of the "word processing" world (see p. DM 27).

They had a well-knit line of magic typewriters and a well-tuned system of upgrades, various stand-alone machines

separate from the mainframes (and with storage equipment, such as the magnetic cards and 16mm sprocketed magnetic tape cartridges, that they refused to connect to computers).

Now the personal computers (their own and others') have eroded and fragmented that control. (As has their own General Systems line.)

Users everywhere (though not, alas, everyone) do text handling and storage now on smaller machines, where it belongs. Mainframe IBM can't prevent it, contain it or recapture it, except perhaps with new software that moves storage back to the MFs.

### WHAT SORT OF A FALL?

I predicted the fall of IBM in my 1977 book, *The Home Computer Revolution*, and IBM did better than ever; but I am standing on it. At this writing IBM has posted losses over several years, and the

company has been seriously reevaluated by investors. **THE STOCK IS DOWN.**

IBM will of course not go out of business. But its sales and its stock will continue to go down appreciably, definitely a "fall" as far as the stock market is concerned. (And ironically, investors and market analysts will see this as the decline of High Tech, whereas it will be the decline of Fat Oppressive Tech.)

**NO WAY CAN IBM GROW AS FAST AS IT USED TO ANY MORE.** IBM will never get back to the same growth rate so beloved of investors in the sixties. And that must lead to an adjustment of its position. IBM will have to find a very different place in the world. And smaller. I predict by 1995 the stock of the corporation will have lost 25% to 50% of its 1980 total value.

### THE SECRET OF IBM

IBM's secret has not been that it's so smart, but that it's so fierce and flexible.

Its form of organization is not to be seen from the way it's divided into divisions, which is just a visible snapshot of the company, but in the many competing task groups that are set up to explore new possibilities. IBM has a dozen major strategies and a huge possibility-grid to choose from at any time.

Most of IBM's projects (it is said) never see the light of day. Which projects finally do emerge, what to call them and how to price them, are command decisions based on many factors—principally, now, the mutual impact (and impaction) of the various product lines.

But its local flexibility is only that. Like a bunch of snakes joined at the tip of the tail, IBM is a huge complex of fiercely competing divisions and projects, able individually to probe for crevices not yet discovered (and occasionally lunging at each other), but unable to escape a common destiny.

And the mutual conflict is growing, and the number of new crevices is fewer.

### IBMOGRAPHY

William Rodgers, *Think*. Stein and Day, 1969. Subtitled: *A Biography of the Watsons and IBM*. Concentrates on the days before computers. Fascinating profile of Watson, a business tiger; but the view of the corporation in an evolving nation is general Americana that transcends fiction.

Would you believe, Rodgers says, Watson was the kingmaker who put General Ike in the White House?

Unfortunately, the book has relatively less on the computer era, so the inside story of many of their momentous decisions since then remains to be told. Out in paperback from the New American Library.

William Rodgers, "IBM on Trial." *Harper's*, May 1974, 79-84. Continues where *Think* left off; examines some of the dirt that came out in the Telex case, and other things.

### THE UNTOLD INSIDE STORY

It is unfortunate that Rodgers' remarkable book does not follow the details of IBM's computer designs and politics in the computer age, i.e., since 1955. Later work, perhaps helped by some Pentagon Papers, will have to relate the decision processes that occurred in this unique national institution to the systems it has produced and the stamp it has put on the world.

# THE BIG QUESTIONS

1. How clean is their salesmanship?
2. Are their systems unnecessarily difficult or cumbersome on purpose?
3. How deep is their system of entrapment and forced commitment of the customer? How necessary are the de-standardizations and the constant changes?
4. Do they have a final liberating vision? Do they really, after all, intend to bring about a day when life is easier for people? When the difficulties of present-day computer systems, especially theirs, wither away? I think that history's judgment on IBM in our time may narrow down to that simple question.

One of the things I wanted to tell people in this book was that IBM had been screwing the world for years, making computers hard to use so they could keep control.

This was true, and greatly needed to be said.

But since then I've found out something else that puzzles me deeply, and deserves a lot of thought.

They didn't know it.

IBM—people from top to bottom of the company—sincerely thought that's how computers were supposed to be.

That somewhat mitigates their crimes against humanity.

Heywood Gould, *Corporation Freak*. Tower (paperback.) Marvelous; hard to get, Gould thinks IBM quietly bought up all the copies.

The musings of a sophisticated, clever and observant cynic who began knowing nothing about IBM, Gould's wide-eyed observation of its corporate style and atmosphere is a jolt to those of us who've gotten used to it. And he thought it was just another big company!

T.A. Wise, "IBM's \$5,000,000,000 Gamble." *Fortune*, Oct '66.

Daniel J. Slotnick, "Unconventional Systems." *Proc. SJCC*, '67, 477-481. Interesting, among other reasons, for the heaviness of the sarcasm directed at IBM and its larger computers.

Robert Samuelson, "IBM's Methods." *New York Times*, Sunday financial section, June 3, '73, p. 1.

This article gives a unique glimpse of some of the interesting things that came to light in the Control Data suit against IBM—citing trial documents never publicly released.

Harvey D. Shapiro, "IBM and All the Dwarfs," *New York Times Magazine*, July 29, '73, pp. 10-36.

An objective, factual article, sympathetic to IBM, although it drew at least one irate letter from an ibmer who didn't think it sympathetic enough.

"IBM: Time to THINK Small?" *Newsweek*, October 1, '73, pp. 80-84.

Frank T. Cary, letter to the editor, *Newsweek*, Oct. 15, '73, p. 4. A snappish reply to the above by the IBM Board Chairman, who evidently didn't like the article very much.

Anonymous, "Anti-Trust: A New Perspective." *Datamation*, Oct. '73, pp. 183-186.

Richard A. McLaughlin, "Monopoly is Not a Game," *Datamation*, Sept. '73, pp. 73-77. Questionnaire survey intended to test truth of common accusations against IBM. *Datamation* devoted large sections of its February and March '75 issues to material on IBM problem.

Robert Sobel, *IBM: Colossus In Transition*. Times Books, '81; Bantam edition. '83.

Almost any issue of *Computerworld* or *Datamation*, the two main industry news publications, carries articles mentioning complaints about IBM from

various quarters on various issues. *Datamation's* letters are also sometimes juicy on the topic.

Same now for Infoworld.

Any issue of *On Line*, a news sheet of the Computer Industry Association, ten bucks a year. (CIA—no relation to the intelligence agency—16255 Ventura Blvd., Encino, CA 91316.)

*A Computer Perspective*, by the office of Charles and Ray Eames, Harvard U. Press, \$13.

Buck Rodgers, Robert L. Shook, *The IBM Wave: Insights Into The World's Most Successful Marketing Organization*, Harper/Row, '86.

Nancy Foy, *The Sun Never Sets On IBM*. Richard T. Delamarter, *Big Blue: IBM's Use and Abuse of Power*, Dodd, '86.

*The IBM Songbook*, any year—they haven't been issued since the fifties— is definitely a collectible.

## "THERE IS A WORLD ELSEWHERE." - Coriolanus

There is no way to escape IBM entirely. IBM mediates our contacts with government and medicine, with libraries, book-keeping systems, and bank balances. But these intrusions are still limited, and most of us don't have to live there.

It is possible to work productively in the computer field and completely avoid having to work with IBM-style systems. Many people do.

### THIBMK, cont.

#### ibmcongruity

one thing getting in another's way: as in ibmecilic or ibmconsistent software, ibmpaction of the product line.

#### ibmcompatibility

fitting together only in certain cases and at great expense.

#### ibmpotent

not being able to get up a decent system.

## Folding Of The IBM Umbrella

For a long time, during the sixties, IBM's high prices provided an environment that made it easy for other companies to come into the field and sell computers and peripherals. These high prices were referred to as "the IBM umbrella."

However, this era has ended. IBM now cuts prices in whatever areas it's threatened.

They want profits, sure, but also the lion's market share. This applies down to the personal-computer level. Let the smaller companies beware: IBM sometimes only gives them a year or two to make their bucks and move on before dropping its own prices. But no way is IBM going to get into K-Mart discount PCs against the Koreans.

"IBM is not a necessary evil. IBM is not necessary."  
—Anon.

An obscure joke for those of us who used to feed punch cards into electronic devices:

Q. How did they bury Thomas J. Watson?

A. Nine edge down.



Rich Hall and friends, in Sniglets, called bar codes

## SCANDROIDS

You can get bar code equipment to hook to your PC. This is good for libraries, record collections, stores, goodness knows what.

A Uniform Product Code (UPC) is the number that's actually stored in the zebra stripes. If you have a product you hope to sell in the zebra stripes, you use automatic scanners, you have to have a UPC zebra stripe.

The first five digits are your manufacturer's number; you get to designate the other five for your particular product. To apply for a manufacturer's number, you apply to the Uniform Product Council, Inc., Dayton, Ohio.

When used by the Defense Department, bar code is called LOGMARS, Logistics Application of Automated Marking And Reading Symbols. (Pentagon studies found it to be three million times more accurate than typed input.)

## BIBLIOGRAPHY

Reading Between the Lines: an Introduction to Bar Code Technology, Russ Adams and Craig Harmon, North American Technology Inc., 174 Concord Street, Peterborough, NH 03458.

You will note that all bank checks now have funny-looking numbers along their bottoms. They go like this:

0123456789

12 34 56 78 90

The numbers are odd but recognizable. The last four thingies are punctuation marks, which presumably can mean anything the programmer wants them to. (In other words, frankly, I don't know their names or standard functions.)

The name of these numbers is

**MICR,**

which stands for Magnetic Ink Character Recording. They are printed in magnetic ink-- not magnetic so's you could record on it, like magnetic tape, but chock full of iron and vitamins so that as its blobs whiz past a special read head, they cause a specific sequence of pulses in the parallel circuits of the read head that can be decoded as the specific number or mark.

The MICR system was designed in the late fifties, with the technology convenient at the time, and would certainly not be designed that way now. Nevertheless, these weird-looking symbols have inspired various

## RIDICULOUS TYPEFACES,

which apparently look to the public like the latest hotcha whizbang zippity up-to-date futuristic stuff, even though to the knowledgeable person they bring back the late fifties. (In fact there are no letters in the MICR character-set.)

We don't need stuff like that any more. Machines are available that can read ordinary typing, so such wonky fonts are obsolete in principle. (If they were designing a checking system now, it would be a lot different.) BUT SUCH STUPID TYPE-FONTS CONTINUE TO PROLIFERATE IN ADS, LETTERHEADS AND MAGAZINES, LOOKING "COMPUTERISH" TO THE PUBLIC. Such fonts are QUINTESSENTIAL cybercrud. (See p. 27.)

# THINGS YOU MAY RUN INTO

Everywhere you go, computers lurk. Yet they wear so many faces it's impossible to figure what's going on.

*Virtually all automatic equipment contains computers or processors now, whether they tell you or not.*

*Hand-held gadgets—even wristwatches—contain processors (see "Bucky's Wristwatch," p. 68).*

*On-line services—things you call up to—are of course computers. These include databases, bank services by phone, stock quotation services, and on and on.*

*How to recognize all this stuff is still tricky for beginners.*

Guidelines are hard to lay down here, but if you look for examples of things you've already run into in this book, it may help some.

Terminals you can presumably recognize.

Microprocessors are harder, because you don't see them. Good rule-of-thumb: any device which acts with complexity or apparent discretion presumably incorporates a terminal, minicomputer or microprocessor.

Two other things to watch for: *transaction systems* and *data base systems*.

A transaction system is any system that takes note of, and perhaps requires verification of, transactions. *Example:* the new *point-of-sale* systems (POS). This is what's about to replace the cash register.

In the supermarket of the future, every package will have a *bar code* on a sticker, or printed on the wrapper. Instead of the checkout clerk looking at the label and punching the amount of the sale into the cash register—an error-prone and cheat-prone technique which requires considerable training—your New Improved Checkout Clerk will *wave a wand* over the bar code. The bar code will be sensed by the wand, and transmitted to a control computer, which will ring it up by amount and category (for tax purposes), and even *keep track of inventory*, noting each object as it is removed from stock.

Here is what your bar code will look like. (A circular code, which was already turning up on some TV dinners, has been

eliminated by the bar code. This is unfortunate, since the scanner necessary to read the bar code is electronically more complicated, but there we are.)



(Incidentally, while this does arrest the classic cashier's cheat—ringing up excessive purchases on the customers, then having a confederate walk through equivalent amounts—the consumer is still entirely prone to cheating by the store in the computer program. Remember, it's 1974. So you still may have to check your tapes, folks.)

*Data base systems* are any systems which keep track of a whole lot of stuff, often with complex pointer techniques (see "Data Structures," p. 70). A cute example is the message service now offered by Stuckey's snack/souvenir stands all over the country. You may leave messages for your friends or loved ones on the road; they can stop at any Stuckey's and ask for their messages, just as if it was a telephone answering service. (You're listed by your phone number—is this to avoid pranks? And what about people with no phones?) It's free and a neat idea. (Obviously, the messages are stored on the disk of a big central computer, and queried from terminals at the individual stands.)

(See "Data Banks," p. 165)

Now, most of the big systems you run into tend to be a combination of transaction and data-base system. For instance, suppose you make an airline reservation. The airline has a large data base to keep track of: the inventory of all those armchairs it's flying around the country, and the list of who so far have announced plans to sit in them, and in some cases what they intend to eat. When you buy your ticket, that *transaction* then gets you put in the listing. Same for car rentals and so on.



The potential dangers of transaction systems are fairly obvious from the super-market example, but they fan out in greater complexity as the systems get more complex. *Credit cards*, for instance, were only made possible by computers and computerized credit verification; but it is only now, fifteen or so years into the credit-card era, that laws protect the cardholder against unlimited liability if he loses it.

Yet we plunge ahead, and it is obvious why. Transaction systems managed in, and by, computers allow more flexible and (in principle) reliable operations. For instance, in the securities' business, thousands of stock certificates are lost and mislaid, and the transaction paper must be typed, shuffled, put in envelopes, sent, opened, shuffled again, compared...all by hand. Little wonder they're working on an Automated Stock Exchange System. But if it's taken fifteen years to get the implicit bugs out of credit cards...not to mention the frequent allegations that much Wall Street "inefficiency" is actually the disguised marauding of Organized Crime...uh-oh. (If they can buy the best lawyers, they can probably buy the best programmers.)

Then there is the Checkless Society. This is a catchphrase for an oft-advocated system that allows you to transfer money instantly by computer; supposedly some such thing is working already in France. Again, they better get it pretty safe before a sane man will go up in it.

Well, *checkless bank cards and Teller Machines are on us now with a vengeance, and most people seem to be pretty satisfied—except for a few people here and there who claim to have been robbed of a lot of money through fraud. There is still room for suspicion and worry.*

The safety of such systems is of course a matter of immense general concern. IBM portentiously announced its intent to spend millions of dollars on "computer security" a few years ago. However, a few million dollars is not going to plug the security holes in the IBM 360, and evidently the 370 is just about as vulnerable.

It is interesting that one profession seems to be stepping forward to try to improve this situation: the *auditing* profession, devoted to verification of financial situations of companies, seems to be branching into the verification of computer programs and the performance of

complex systems. This will be great, if it works. Cynics, however, may note that auditors have permitted some remarkable practices in the "creative" accounting of recent years. (Obviously the way to check out the safety of big systems is to *offer bounty* to those who can break its security. But who is willing to subject a system to a test like that?)

Hereabouts are a few other computerish things you may run into which more or less defy categorization.

## SIMULATION

is an imposing term which means almost anything. Basically, "simulation" means any activity that represents or resembles something. Computer simulation is using the computer to mimic something real, or something that might be, for any purpose: to understand an ongoing process better, or to see how something might come out in the future.

Here again, though, the Science myth steps in to mystify this process, as though the mere use of the computer conferred validity or some kind of truth.

(On TV shows the Space Voyagers stand in front of the "computer" and ask in firm, unnaturally loud voices what will be the results of so-and-so. The computer's oracular reply is infallible. On TV.)

Let there be no mystery about this. Any use of a data structure on a what-if basis is Simulation. You can simulate in detail or crudely; your simulation can embody any theories, sensible or stupid; and your results *may or may not correspond to reality*.

A "computer prediction" is the outcome of a simulation that someone, evidently, is willing to stand behind. (See "Computer Election Predictions," p. 154.)

These points have to be stressed because if there is one computer activity which is pretentiously presented and stressed, it is simulation. Especially to naive clients. There is nothing wrong with simulation but there is nothing supernatural about it either.

Another term which means more or less the same is *modelling*.

In the loose sense, simulation or modelling consist of calculations about any describable phenomena—for instance, optical equations. In optical modelling (and this is how they design today's great lenses), a data structure is created which represents the curvature, mounting, etc. of the separate glasses in a lens. Then "simulating" the paths of *individual rays of light* through that lens, the computer program tests that lens design for how well the rays come together, and so on. Then the design is changed and tried again. ■

## Tracking Of Animals And People ■

*Wild animals, including bears, whooping cranes and salmon, now are being fitted with beepers (or in the salmon's case, transponders) that permit tracking of the individuals for purposes of wildlife management. Typically the creature's wanderings go into a computer database for later analysis.*

*But these same techniques have an ominous side. For instance, they are now being used in some states to enforce house arrest: probationers wear a location transmitter, unremovable, so that their whereabouts can be always known—an electronic ball-and-chain.*

*Combine this technique with the approach to controlling whole populations taken by South Africa and before that Nazi Germany, and the prospect for immense new tyrannies requires no stretch of the imagination.*

*The automatic teller machines are actually fairly interesting. They have two modes of operation—Connected to their host computers, and Disconnected, as when the computer goes down. During the down times the machines actually do not show their delicate condition; but they go to a second mode of operation, such as assuming that the last current balance recorded magnetically on the back of your card is still correct, and acting accordingly.*

## ● Very Important Statement: All Simulation Is Political

*Every simulation program, and thus every simulation, has a point of view. Just like a statement in words about the world, it is a model of how things are, with its own implicit emphases: it highlights some things, omits others, and always simplifies. The future projections made by a simulation only project these views forward in time.*

*In the politics of the future, all sides will have simulations, projections, charts, ostensible results. Some will be good. Most will be biased. Many will be rhetorical, like speeches, but supplying dummy numbers to their presenters, and thus looking important and factual when, like speeches, they are emptier than they seem.*

*But we must all use simulations, and the good-faith political processes of the future must merge them (see p. 174).*



# PREDICTIONS and PROJECTIONS

*are just the extension of simulations into the future.*

*Their accuracy depends on the accuracy of the theories and models they represent, not on the fact that a computer is doing them. And, alas, their verification depends upon the particulars (and detail-dripping research) of each individual field.*

## That's What Makes Horseracing

"Simulation" means almost anything that in any way represents or resembles something. Which is not to say it's a useless or improper term, just a slippery one.

*Examples.* Here are ways we could "simulate" a horse race: Show dots moving around an oval track on a completely random basis, and declare the first to complete the circuit The Winner.

Assign odds to individual horses, and then use a randomizer to choose the winner, taking into account those odds. (This is how the PLATO "horserace" game works; see p. DM 20.)

Give conditional odds to the different horses, based on possible "weather conditions." Then flip a coin (for the computer equivalent, weighted randomization) to test the "weather conditions," and assign the horse's performance accordingly.

Program an enactment of a horse race, in which the winner is selected on the basis of the interaction of the horoscopes of horse and rider.

Create a data structure representing the three-dimensional hinging of horse's bones, and the interlaced timing of the horse's gait. (This has been done at U. of Pennsylvania on a DEC 338.) Then have these stick figures run around a track (for the data structure equivalent).

Using a synthetic-photography system such as MAGI's Synthevision (see p. DM 106), create the 3D data structure for the entire surface of a running horse over time; then make several copies of this horse run around a track, and make simulated photographs of it.

And so on.

So don't be snowed by the term "simulation." It means much, little or nothing, depending.

Another type of simulation, an important and quite distinct one—is that which represents the complex interplay of myriad units, finding out the upshots and consequences of intricate premises. In traffic simulations, for instance, it is easy enough to represent thousands of cars in a data structure, and have them "react" like drivers—creating very convincing traffic jams, again represented *somehow* within the data structure.

Basically simulation requires two things: a representation, or data structure, that somehow represents the things you're simulating in the aspects that concern you; and then a program that does something to these data, that is in some way *like* the process you're concerned about acting on the things you're modelling. And each event of significance enacted by the program must somehow leave its trace in the data structure.

The line between simulation and other programming is not always clear. Thus the calculation of the future orbits of the planets *could* be called "simulations."

The most intricate cases, though, don't particularly resemble any other kinds of programs. The intricate enactments of physical movements, especially swarms and myriads with mixed and colliding populations, are especially interesting. (In a recent *Scientific American* article, simulation helped to understand possible streamers of stars between galaxies as resulting from normal considerations of inertia and gravitation. (Alar and Juri Toomre, "Violent Tides between Galaxies," *Sci. Am.* Dec. '73, 38-48.)

Models of complex and changing rates are another interesting type. Enacting complex things, whose amounts are constantly changing in terms of percentage multipliers of each other, sound easy in principle, but their consequences can be quite surprising. (See "The Club of Rome," p. 172.)

To imagine the kinds of mixed-case myriad models now possible, we could on today's big computers model entire societies, with a separate record describing *each individual* out of millions, and specifying his probabilities of action and different preferences according to various theories—then follow through whole societies' behavior in terms of education, income, marriage, sex, poverty, death, and anything else. Talk about tin soldiers and boats in the bathtub.

Any computer language can be used for *some* kind of simulation. For simulations involving relatively few entities, but lots of rates or formulas, good old BASIC or FORTRAN is fine. (MAGI's "Synthevision" system, which could be said to "simulate" complex figures in a three-dimensional space, is done in FORTRAN; see p. DM 59.) For simulations involving a lot of separate objects, special cases and discrete events, TRAC Language (see p. 61) is great. If numerous mathematical formulas are involved, and you want to change them around considerably in an experimental sort of way, APL is well suited (see p. 58).

There are a number of special "simulation" languages, notably SIMSCRIPT and GPSS. These have additional features useful, for instance, in simulating events over time, such as "EVENT" commands which synchronize or draw division-lines in time (the simulated time). Simulation languages generally allow a great variety of data types and operations on them.

The list-processing fanatics, of course, insist that their own languages (such as LISP and SNOBOL) are best. And then there's PLATO (see p. DM 93), whose TUTOR language is splendid for both formulas and discrete work—but allows you only 1500 variables, total (60 bits each).

The thing is, any set of assumptions, no matter how intricate, can be enacted by a computer model. Anything you can express exactly can be carried out, and you can see its consequences in the computer's readout—a printout, a screen display, or some other view into the resulting data structure.

Obviously these enactments (or sometimes "predictions") are wholly fallible, deriving any validity they may have from the soundness of the initial data or model.

However, they have another important function, one which is going to be very important in education and, I hope, general public understanding, as computers get spread about more widely and become more usable.

The availability of simulation models can make things easier to understand. Well-set-up simulation programs, available easily through terminals, can be used as Staged Explanatory Structures and Theoretical Exploration Tools. The user can build his own wars, his own societies,



his own economic conditions, and see what follows from the ways he sets them up. Importantly, *different theories* can be applied to the same setups, to make more vivid the consequences of one or the other point of view.

(Indeed, similar facilities ought to be available for Congress, to allow them to pour a new tax through the population and see who suffers, who gains...)

I should point out here that for this purpose—Insightful Simulation—you don't always need a computer. I have in mind the so-called "simulation games," which if well designed give extraordinary insights to the players. Ailen Calhamer's brilliant game of Diplomacy, for instance, teaches more about international politics than you could suppose possible. I am also intrigued by a game called "Sim-soc," worked out by a sociologist to demonstrate the development of social structures from a state of random creation, but I haven't played it. (Clark C. Abt, of Abt Associates, Boston, has also done a lot of interesting design here.)

A last point, a very "practical" application. Simulation makes it possible to enact things without trying them out in concrete reality. For instance, in the lens-design systems mentioned earlier, the lenses don't have to be actually built to find out their detailed characteristics. Nor is it necessary to build electronic circuitry, now, to find out whether it will work—at least that's what the salesmen say. You can simulate any circuit from a terminal, and "measure" what it does at any time or in any part with simulated meters. Similarly, when any computer is designed now, it's simulated before it's built, and programs are run on the *simulated* computer, as enacted within a *real* computer, to see if it behaves as intended. (Actually there are some hot-wire types who insist on building things first, but one assumes that the more *sensible* computer designers do this.)

With automobiles it's harder; but GM for instance, simulates the handling characteristics of its cars before they're ever built—so that designers can redistribute weight, change steering characteristics and so on, till the handling characteristics come out the way the Consumers seem to like.

## BIBLIOGRAPHY

*Simulation* magazine is the official journal of Simulation Councils, Inc., the curiously-named society of the Simulators. Simulation Councils, Inc., Box 2228, La Jolla, CA 92037.

For all I know you get annual membership free with that. I've always wanted to join but it was always the one thing too many; but their conference programs are sensational. Where else can you hear papers on traffic, biology, military hardware, weather prediction and electronic design without changing your seat?

## OPERATIONS RESEARCH

is an extension of Simulation in a fairly obvious direction.

If simulation means the Enactment of some event by a computer, Operations Research means doing these enactments to try out different strategies, and test the most effective ones.

Operations research really began during World War II with such problems as submarine hunting. Given so-and-so many planes, what pattern should they fly in to make their catching submarines *most likely*? Building from certain types of known probability, (but in areas where "true" mathematical answers were not easily found), operations researchers could sometimes find the best ("optimal") strategies for many different kinds of operation.

Basically what they do is play the situation out hundreds or thousands of times, enacting it by computer, and using dice-throwing techniques to determine the outcomes of all the unpredictable parts. Then, after all entities have done their thing, the program can report on what strategies turned out to be most effective.

*Example.* In 1973 the *Saturday Review* of something-or-other printed a piece on the solution, by OR techniques, of the game of Monopoly. Effectively the game had been played thousands of times, the dice thrown perhaps millions, and the different "players" had employed various different strategies against each other in a varying mix: Always Buy, Buy Light Green, Utilities and Boardwalk, etc.

A complete solution was found, the strategy which tends (over many plays) to work best. I forget what it was.

Using another technique, the game of football was analyzed by Robert E. Machol of Northwestern and Virgil Carter, a football personage. Their idea was to test various maxims of the game, to find out which common rules about beneficial plays were true. What they did was replay fifty-six big-league football games on a play-by-play basis, rate the outcomes, and see which circumstances proved most advantageous on the average. I've mislaid the reprint (*Operations Research*, a recent year), and being totally ignorant of football can remember none of the findings. Anyhow, that's where to look. (Citation found below.)

The earlier explanation of Operations Research wasn't quite right. It's any systematic study of what works best. Computers can help.

## BIBLIOGRAPHY

Irvin R. Hentzel, "How to Win at Monopoly." *Saturday Review of Science*, Apr. '73, p. 44-8.

Virgil Carter and Robert E. Machol, "Operations Research on Football." *Operations Research*, March 1971, p. 541-544.

## MILITARY USES OF COMPUTERS

A lot of people think computers are in some way cruel and destructive. This comes in part from the image of the computer as "rigid" (see "The Myth of the Machine," p. 31), and partly because the military use so many of them.

But it's not the *nature* of a computer, any more than the nature of a typewriter is to type poems or death warrants.

The point is that the military people are gung ho on technology, and keen on change, and Congress buys it for them.

No way is there room to cover this subject decently. But we'll mention a few things.

The Pentagon, first of all, with its payroll of millions, with its stupendous inventories of blankets and bombs and toilet paper, was the prime mover behind the development of the COBOL business computing language. So a vast amount is spent just on computers to run the military establishment from a business point of view.

Of course that's not the interesting stuff.

The *really* interesting stuff in computers all *came* out of the military. The Department of Defense has a branch called ARPA, or Advanced Research Projects Agency, which finances all kinds of technical developments with vaguely military possibilities.

It is thus a supreme irony that ARPA paid for the development of: COMPUTER DISPLAY (the SKETCHPAD studies at Lincoln Labs; see p. DM 96); TIME-SHARING (e.g. the CTSS system, see p. 103); HALFTONE IMAGE SYNTHESIS (the Utah algorithms; but see all of pp. DM 100-119); and lots more. Some folks might say that proves it's all evil. I say let's look at cases. While they have military applications, that's simply because they have applications in *every* field, and the military are just where the money is.

Just to enumerate a few more military things—

***Simulation has opened vast new windows everywhere. Simulation is now a central aspect of every field, including politics, war (conventional and nuclear (gasp)), business, engineering, and all the sciences. Indeed, simulation is essentially the frontier of every science now, permitting exploration of untold new realms of ideas, some of which can be checked experimentally now and some of which can't.***

***There is no one form of simulation for any field. For instance, in chemistry the most popular simulation programs—all different, but with a lot of overlap—are FRODO, MOGLI, SYBYL, HYDRA, Macromodel, Biograf, Chem-X, and "Discover & Insight."***

***The frontier of the theory of evolution, in biology, is also computer simulation. John Holland, of the University of Michigan, built early models which searched through imaginary possibilities in much the way that genes do, seeking benefit as organisms do. Now, much later, the frontiersman is Douglas Lenat at the University of Texas, whose models deal with studying the tricks that genes use for optimum speed of adaptability.***



☛ **Command and control**—the problem of keeping track of who's done what to whom, and what's left on both sides, and getting orders through.

It is a solemn irony that the great "465L Command and Control System"—a grand room with many projectors driven by computer, only something like those in "Dr. Strangelove" and "Fail-Safe"—may be a prototype for offices and conference rooms of the future.

"Avionics"—all the electronic gadgets in airplanes, including those for navigation. (A recent magazine piece described how wonderful it felt to fly the F-111—which has a computer managing the Feel of the Controls for you.)

"Tactical systems"—computers to manage battlefield problems, aim guns and missiles, scramble your voice among

various air frequencies or whatever they do.

"Intelligence"—computers are used to collate information coming in from various sources. This is no simple problem—how to find out what is so from a tangle of contradictory information; think about it. Don't think about how we get that information.

"Surveillance"—it can't all be automatic, but various techniques of pattern recognition (see p. DM 121) are no doubt being applied to the immense quantities of satellite pictures that come back. (Did you know our Big Bird satellite either chirps back its pictures by radio, or parachutes them as Droppings?)

Of course, the joker is that all this obsession with gadgets does not seem to have helped us militarily at all. The army seems demoralized, and the navy losing

ground to a country that hardly even has computers.

## JUNK MAIL



### DIRECT MAIL THE PERSONAL MEDIUM

The people who send it out like to call it personalized advertising and the like. But most of us call it Junk Mail. And its vagaries are NOT THE POOR COMPUTER'S FAULT. What gets people angry derives from the system built around the poor computer.

Much of the success of the political right wing in recent years has been their highly effective use of mailing lists. (For the Democrats' incompetence with mailing lists, see p. 42.) The mastermind has been "Conservative Activist" Richard Viguerie, whose computers maintain huge mailing lists of persons with any conservative leanings, indexed by the issues to which these individuals are most sensitive. This permits highly-targeted mailing campaigns to go out incessantly on specific issues. These mailings can pinpoint individuals who care most about a number of differing specific issues whether or not these individuals would be generally sympathetic to Mr. Viguerie's overall point of view.

## Your New Favorite Annoyance:

### ● Ju n<sub>k</sub> Phone Calls

**Yup, now the recorded messages call you, and give you a chance to fill in the blanks by voice. This is called TELE-MARKETING.**

**Sometimes they're fun, but the annoyance will increase as more and more calls of the day are instigated by unattended machines hustling products and candidates. One defense is an answering machine. THERE IS A GOOD SIDE, however, and that is that soon you ought to be able to send phone messages to and from your friends the same way—making "telephone tag" useful, without anybody being interrupted and without the extraneous social preliminaries. Tele-marketing setups are still several thousand bucks; but home setups ought to bring that down in a couple of years.**

**SIOP, the Single Integrated Operation Plan, is the second-by-second schedule on which the United States is prepared to conduct World War Three. It is updated constantly.**

**According to Canadian military researcher Guynne Dyer, SIOP calls for Moscow to be hit by no less than sixty thermonuclear warheads.**

#### READING

**Guynne Dyer, War, Crown, 1985.**

The army's battlefield computers of the seventies were big, ungainly, ruggedized machines, custom-built to military specifications.

Today, however, the army has gone over to standard off-the-shelf commercial PCs, having found them nearly as reliable, and far more disposable.

According to my sources, the modern army is planning to go into battle not only with standard office equipment, but using—what else? Ordinary commercial word processing, spreadsheet and database programs.

You may wonder why you get more and more seed catalogs, or gift-house catalogs, as time goes on, even though you never order anything from them. Or why a deceased member of the household goes on getting mail year in and year out, regardless of your angry postcards.

How does it keep coming?

Through the magic of something called the Mailing List. And especially the peculiar way that mailing lists are bought and sold.

Now, a mailing list is a series of names and addresses of possible consumers, stored on computer tape or disk.

You can buy the use of a mailing list.

But you cannot buy the mailing list itself.

Suppose you have a brochure advertising pumpkin-seed relish, which you suggest has rejuvenating powers. You want this brochure to go out to rich college graduates.

You go to a mailing-list house.

"I cannot sell you this mailing list outright," says the jolly proprietor, "for it is my business to sell its use again and again, so I do not want anybody else to have a copy of it." So you leave 2500 pumpkin-seed relish brochures with the mailing list company, and pay them a lot of money. And they swear on a stack of bibles that they have mailed the brochures to their special list of rich college graduates.

Well, let's say you get 250 sales from that mailing (10% is fantastically good.) But out of curiosity you go to another mailing-list house and have another mailing sent out—this one to people who have low incomes and little education. This time you get 15% orders.

Now guess what you are acquiring.

A mailing-list of your very own. Of people who eat pumpkin-seed relish.

Mailing lists are, you see, generally rented blind, with no chances to see the addressees or check as to whether they've already been mailed to.

And that explains all the duplications.

If an advertiser is going after a certain type of customer, and goes to several mailing-list houses asking for mailings to that particular type of customer, chances are some people will be on several of the lists. And since there's no way to intercompare the lists, these poor guys get several copies of the mailing.

(Another way this can happen is if some cheapskate has his own mailing list and doesn't check it for repeats of the same name. But writing the computer program to check for repeats of the same name is not easy—there might just be a Robert Jones and a Rob Jones at the same address—and these things are not usually checked manually. They're big.)

Another possibility exists for eliminating duplications when you rent mailing lists. You can bring in a magnetic tape with *your* mailing list on it, and they can send out the mailing only to the members of *their* list who are not already on *your* list. That way you still can't steal their list, since the tape is on their premises. The trouble is, they can steal your list, by making a copy of the tape. Oh dear.

One possibility, nice and expensive, is to rent a number of mailing lists from a single mailing-list house, with them guaranteeing that they'll compare all the lists you choose and not send to any person more than once.

But as you may be suspecting, this costs money. All this screening and inter-comparing requires computer time, and so, even though you are getting a more and more perfect mailing, you are paying more and more money for it. So you can see why reasonable businessmen are willing to send out ads even when they know some recipients will get several duplicates.

Another interesting point. There are mailing lists for all kinds of different possible customers. The possibilities are endless. Minority-group doctors. People interested in *both* stamp collecting and flowers (you'd have to get a company with both lists, and have them go through them *for the duplicates*...you get the idea).

Note that mailing lists are priced according to their desirability. Weeded mailing lists, featuring only Live Ones, people who've ordered big in recent times, are more expensive. Lists of doctors, who buy a lot, are more expensive than lists of social workers. And so on. Then there's the matter of pitch.

The ad's phrasing may be built around the mailing plan. Some circulars come right out and tell the recipient he's going to get several copies because he's such a wonderful person.

THEN there are those advertisements that are actually *printed by the computer*,

or at least certain lines are filled in with the recipient's name and possibly some snazzy phrases to make him think it's a personal letter. Who responds to such things I don't know. My favorite was the one—I wish I could find it to include here—that went something like:

You'll really look swell, Mr. Nelson walking down Main Street of New York in your sharp-looking new slacks...

I don't know whether I enjoyed the spaces or the Main Street more.

But you see how this works. There's this batch-processing program, see, and the names and addresses are on one long tape, and the tape goes through, and the program takes one record (a name and address), and decides whether to call the addressee "Mr.," "Ms." or whatever, and then plugs his name into the printout lines that give it That Personal Touch; and then the mailing envelope or sticker is printed; and the tape moves on to the next record.

We may look forward to increasing encroachments on our time and trust by the direct mail industry: especially in better and better quack letters that look as though they've really been personally typed to you by a real human being. (It is apparently legal for letters to be signed by a fictitious person within a company.) In the future we may expect such letters to be sent on fine paper, typed individually on good typewriters, and convincingly phrased to make us think a real personal pitch is being tendered.

There is, however, a final solution.

YOU CAN GET OFF ALL MAILING LISTS—that is, the ones "participating" in the Association—by writing to:

Direct Mail Marketing Association  
6 East 43rd Street  
New York, NY 10017

They will send a blank. If you fill it in they'll process it and delete your name from mailing lists of all participating companies.

Presumably this won't help with X-rated or stamp-collecting lists, but it ought to keep you from getting semianual gift catalogs from places like The House of Go-Go Creative, Inc., and those million solicitations from Consumer Reports and that File Box Company.

**CONSUMER BANK**

Metropolitan Division  
P.O. Box 5144 Church Street Station, New York, N.Y. 10049

Senior Vice President

Theodor H Nelson  
458 W 20th St  
New York, NY 10011

Branch 014

**FILL-INS**

Great news for the Nelson Family!

Wouldn't you like your money to work for you full time... even when you're asleep?

Now the Nelson family can save...right at their own bank.

... 5% Passbook Savings Plan which  
... quarterly or even

NO-RISK...  
If not delighted with  
package, I may return it within  
10 days and my membership  
will be canceled. I will owe  
nothing.  
DO NOT RETURN THIS FORM IF  
YOU ARE ALREADY A MEMBER

(Whoever  
you are)

Yes, Mr. Ainsworth, as a former member of...  
Literally build you qualify for this exciting offer in  
5 books for 10¢ sent to you as soon as you enroll in  
the BARGAIN BOOK CLUB. And instead of taking a book a  
month for 6 months as regular members do, you'll have 12  
months -- TWICE THE TIME -- to find the 6 books you want.

**FILL-INS**

We can make this offer to you because as a member of  
Doubleday's book club family for 20 months, you proved  
an interest in reading at bargain prices. Since you  
left us in the autumn of 1972, Bargain Book Club members  
in Chicago have enjoyed such big books as Irving  
Wells' THE WORD, Dorothy Eden's SPEAK TO ME OF LOVE,  
and Horfield's TO SERVE THEM ALL. MY DAYS and Frank  
... and enjoyed them all at

Dear Reader:

If the list upon which I found your name is any indication, this is not the first -- nor will it be the last -- subscription letter you receive. Quite frankly, your education and income set you apart from the general population and make you a highly-rated prospect for everything from magazines to mutual funds.

You've undoubtedly "heard everything" by now in the way of promises and premiums. I won't try to top any of them.

If you subscribe to ~~the book~~, you won't get rich quick. You won't bowl over friends and business with clever remarks.

1255 PORTLAND PLACE, BOULDER, COLORADO 80302

Dear Mr. Nelson:

Let's get straight to the point. This is not an ordinary letter. It's a subscription offer, from a magazine. It can save you money off the regular price. And if it perhaps won't appeal to everyone, Poughkeepsie, we think it will appeal to you, Mr. Nelson.

Because it  
gives you

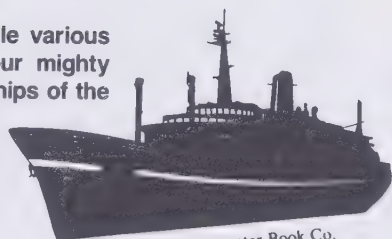
LOOSY  
MAILING  
LIST  
I HADN'T  
LIVED IN  
POUGHKEEPSIE  
FOR SIX YEARS.



Is it too soon for a computer pornography contest? (Is it too late?)



Minicomputers handle various control functions in our mighty new Aeroplanes and Ships of the Ocean.



© Copyright 1986 Artmaster Book Co.

## Electronic Dashboard READOUTS

Maps and snaps of what's happening in the car are on lots of new dashboards. Most are flat panels; the 1986 Buick Riviera has a CRT.

### ELECTRONIC DASHBOARD MAPS

Various schemes exist to provide the driver with electronic maps on a dashboard screen. Some of these involve disk storage, some involve radio.

Some are supposedly operational, but none has caught on big yet. The problem of update is enormous, the problem of telling which way the car is facing (so the screen image can rotate correctly) is not trivial, and the cost of the systems are so far in the thousands. One system, ETAC, clocks mileage by your wheels and direction by a compass on the roof; it costs about \$1500.

## “COMPUTER ELECTION PREDICTIONS”

This is an outrageous misnomer. The computer is only carrying out, most speedily, what hardened politicians have always done: FACTIONAL ANALYSIS, now possible with newfound precision on the basis of certain election returns.

This is based on the cynical, and fairly reliable, view that people vote according to what faction of the greater populace they belong to—middle-class white liberals, blue-collar non-union members, and so on. The factions change slowly over time, and people move among them, but the *fact* of factionalism remains unchanged.

Well. By the close of a major election campaign, most factions can be pretty well predicted, especially as to presidential choice, or *what proportion* of that faction will go for a given candidate.

But some factions' reactions are not certain up to the day of the ballot.

So. “Computer predictions” of elections basically break the country into its factional divisions, state by state and district by district, and then tabulate who can be predicted to vote for whom on a factional basis.

Then what's the suspense?

The suspense comes from the *uncertain factions*—groups whose final reactions aren't known as the election starts.

Certain election districts are known to be chock full of the types of people whose reaction isn't known.

The final “computer prediction” simply consists of checking out how those districts voted, concluding how those factions are going in the present election, and extending this proportion through the rest of the country.

It's often painfully accurate—but, thank God, not always. When it isn't, don't blame “the computer.” Thank human cantankerousness.

## “COMPUTER DATING”

should of course be called MATCHUP DATING, since there is nothing particularly computerish about either the process or its intended result. But there we go again: word-magic, the implicit authority of invoking the word Computer. (See “Cybercrud,” p. 27.)

In the early sixties, a perky young fella at the Harvard B-School, I believe, one Jeff Tarr, came up with the notion of a computerized dating service. The result was Operation Match, an immense financial success, which sort of came and went. No followup studies were ever done or success statistics gathered, unfortunately, but they certainly had their fun.

The basic principle of “computer dating” is perfectly straightforward. Applicants send in descriptions of themselves and the prospective dates they would like to meet. The computer program simply does automatically the sorts of things you would do if you did this by hand: it attempts to find the “best” match between what everybody wants and what's on hand.

Obviously this could be a matter for serious operations research: attempting to discover the best matchup techniques among things that never really fit together, detail for detail; trying to find out, by follow-up questionnaires, what trait-matchings seemed to produce the best result, etc. But such serious matchup-function research remains, so far as I know, to be even begun.

Obviously there are several problems. Demographically it is almost never true that “for every man there's a woman”—in every age-bracket there's almost always an imbalance of the opposite sex in the corresponding eligible age-bracket, either too many or too few. But more than that, there is little likelihood that the traits women want are adequately represented among the available males, or vice versa. For introduction services it's obviously worse: there is no balance likely between what comes in one door and what comes in the other. The service can only do its best with the available pool of people—and make believe it's somehow made ideal by the use of the computer. It's like an employment office: applicants don't match openings.

Numerous other dating services have appeared, some of which don't even pretend to use the computer (and others which claim to be a registry for nonstandard sexual appetites), but none that's gotten the attention of the original Project Match.

But there's no question who got the best dates out of that one. Jeff Tarr.

# SIC TRANSIT

## YOUR AUTOMOBILE COMPUTER

Didja know, huh, we're going to have computers in our cars? We refer here to two things—

*anti-skid controllers*, which are really just special circuits—you know, “analog computers”—to compensate among skidding wheels. Turns out that this is apparently more sensitive and reliable than even your good drivers who *enjoy* controlling skids. Already advertised for some imports.

*grand bus electronics* (see p. 98). Since the electrical part of the automobile is getting so blamed complicated, the Detroit Ironmongers have decided to switch to a grand bus structure instead of having all those switches and things separate anymore. Should make the whole thing far easier to service and customize.

Presumably this will all be under the control of a microprocessor. (See p. 96.) This means that the car can have things like a Cold-Weather Startup Sequence—a program that starts the car, turns on the heater, monitors the engine and cabin temperature, and bleats the horn, twice, politely when it's all ready—all at a time preset by the dashboard clock.

Presumably Detroit is not yet planning to go this far. But because of the auto industry's anomalously huge influence in America, some have expressed the fear that this move—toward the integrated-circuit, digitally-controlled grand bus—would effectively put Detroit in control of the entire electronics industry.

The ever-clever Japanese are computerizing faster, better and more deeply than we are.

They now have a prototype taxi operating under computer control. They're calling it, at least for export, Computer-controlled Vehicle System (CVS).

Basically it's like an Elevated Railway—you climb up and wait—but when you get in, you punch a button for your destination. According to Hideyuki Hayashi of the Ministry of Industry and International Trade, the system will be operational in Tokyo within the decade,

and is the “cleanest, safest, quickest transport system ever devised by man.” Think fast, Detroit.

(A nice point: one of the most important features of such a system is that the vehicles *don't react to each other*, as do vehicles in the existing Human-controlled Vehicle System (HVS). A whole line of the cars can be accelerated or slowed simultaneously, a crucial aspect of their flexibility and safety. Nothing can possibly go long.)

(Leo Clancy, “Now—Computer-Controlled, Driverless Cars,” *National Enquirer*, 3 Mar. '74, pp. 24-25.)

## THOSE THINGS ON THE RAILROAD CARS

As we lean on the fence a-chawin' an' a-watchin' the trains go by, we note strange insignia on their sides, in highly reflective Scotch-Lite all begrimed by travel.

Basically it's a stack of horizontal stripes in red, blue and other colors. This is ACI, for Automatic Car Identification. It may yet straighten out the railroads.

In this neolithic industry, it is not known at any given time where a railroad company's cars are, and some peculiar etiquette governs their unrequested use by other firms in the industry. Yet the obvious solution may come about: a running inventory of where all the cars are, where each one is going, what's in it, and who that belongs to. But, of course, that's still in the works. Revolutionary ideas take time.

## GETTING THE REAL SCHEDULES

*Most airline reservation systems are sponsored by particular airlines, and are programmed to operate with blatant prejudice against other carriers. For instance, let us suppose you are using a reservation system operated by Pegasus Airlines and you want to make a connection at a certain airport where your first flight gets in at 6:00. Well, the Pegasus computer will show you its flight out at 6:30, but it may not show you flights out by other carriers leaving before 7:00.*

*That's called a half-hour bias.*

*Some reservation systems have a bias of over an hour. And they get away with this.*

*Well, anyone can get on-line access to an unbiased system—the Official Airline Guide—via Western Union EasyLink. (Can't reserve through it, though.)*

## The VW Checkout Coupler

may or may not be a real computer—friends have told me it isn't—but it's certainly a good idea.

When you pull your late-model Volkswagen into a dealer's service area, the guys can just roll out a cable and plug it into the corresponding socket in your vehicle. At the other end of the cable is some sort of device which tests a series of special circuits throughout the car for Good Condition. These circuits indicate that things are working properly—lights, plugs, points, brakes and so on.

This is the same technique used by NASA up to the final moment of COMMIT LAUNCH—a system of circuits monitors the conditions of whatever can be monitored, to make sure all's functioning well. It's more expensive to wire it up that way, but it makes checking out the rocket—or the car—that much easier.



## Things You May Run Into Include Other

### AIR PLANES

*The air traffic control system is outdated, overloaded, understaffed, and hell on the people who run it. Their computer screens constantly notify them of problems (such as planes too close) which they must handle by voice contact with pilots.*

*Now the old margins for error are being shaved. The system has recently been reprogrammed to notice only smaller distances between airplanes. Many flybys which would have been considered near misses in 1985 are now considered acceptable; meanwhile, the reported near misses under the new standards are higher than ever.*



## THIS PETTY PACE

New brands of expensive running shoes have little computers in them to count how far you went by the timing of your strides. You can fool them, however, by just jumping up and down.

### DO YOU GOT RHYTHM?

A device called the BIO-COMPUTER (trademark) purportedly helps you predict your "body beats," telling you what days are the right sort of time to do particular things in terms of your own biological energies. The object costs \$15 postpaid from BIO-COMPUTER, Dept. CLB/DM (why not?), 964 Third Ave., NY, NY 10022.

*A portable clock for Moslems gives the exact time for the five daily prayers and the correct direction to face, wherever in the world the user may be.*

"Beeping the Faith," Scientific American, March 1987, p. 74.

The question with all such special purpose devices—"fishing computers," horse-racing computers, etc., is always whether the theory and formulas which are built into them are correct. There is no ready way to tell.

## "THE WORD" PROCESSING

Bible Research Systems, Austin, TX, sells the Bible on disk in different translations, as well as various research tools to search it, mark it, index it and whatnot.

### ASTROFLASH, etc.

There are various computerized astrology services. Given your date of birth, and hour if known, they'll type out your signs, explanations, etc. Presumably there is a text network which the system selects among according to "reinforcing tendencies," etc., among the entities thought to be influential.

Conceivably this could do nine-tenths of what a talented human astrologer does, and with the same validity, whatever that may be. In any case it's probably a lot cheaper.

### BETCHA DIDN'T KNOW...

that the IRS hasn't been able to do instant matching of W-2 forms to tax returns. That'll be fixed in fiscal '74, and interest and dividend payments in '75. (TIME, 31 Dec. '73, p. 17.)

*The IRS has a lot of problems, you*

*know. In 1985, they cut over to a different computer system without running the old one as a backup—a trapeze act that could have left the government in a pretty pickle had it failed.*

Two of the ten new untested computers that Sperry delivered never worked, and the IRS felt compelled at this same delicate moment to make major revisions in their entire software system of 3.5 million lines of code, 500 programs.

*All this while the IRS budget was going down. Who says bureaucracies have it easy?*

*And they talk about stopping thousands of incoming missiles with government computers! (See p. 171.)*

(David E. Sanger, "At IRS, New Computer System is More Troubling Than April 15." New York Times, 15 April '85, p. 23.)

*The IRS is now accepting tax returns on floppy disk and via modem on a small scale, but they definitely prefer that you use your personal computer to prepare your return with such programs as Macintax, which produces a printout that looks just like the IRS form. This is the WY-SIWYG philosophy (see p. DM 28).*

## NAGGING APPLIANCES

In The Hitchhiker's Guide to the Galaxy, a number of pieces of equipment have objectionable personalities—unctuous doors that make oily remarks as they open and close, and Eddie the Shipboard Computer, who is insufferable.

Author Douglas Adams has his finger on something here. Appliance etiquette is too important to be left to engineers.

Sugar Creek, Texas will have 3000 homes with a minicomputer-based alarm system. Evidently various automatic sensors around each house sniff for fires and burglars, as well as providing panic buttons for medical emergencies.

The system uses dual Novas (one a backup), and prints out the news to fire and police dispatchers on a good old 33ASR Teletype. (Digital Design, May '73, p. 16.)

*Such things are all over the place now.*

### "COMPUTERS" THAT DON'T ANSWER

Few of us can help feeling outrage at the book clubs, or subscription offices, or billing departments, that don't reply to our letters. Or reply inappropriately, with a form printout that doesn't match the problem.

First let's understand how this happens.

These outfits are based on using the computer to handle all correspondence and transactions. The "office" may not have any people in it at all—that is, people whose job it is to understand and deal sensibly with the problems of customers. Instead, there may just be keypunch operators staffing a Batch System, set up by someone who has long since moved on.

The point of a batch system (see p. 102) is to save money and bother by handling everything in a controlled flow. This does not mean in principle that things have to be rigid and restrictive, but it usually means it in practice. (See "The Punch Card Mentality," p. 72.) The system is set up with only a fixed number of event types, and so only those events are recognized as occurring. Most important, your problem is assumed to be one that will be straightened out in the course of the system's flow. While there may be provision for exceptions—one clerk, perhaps—your problem has not seemed to him worthy of making an exception for.

Here is my solution. It has worked several times, particularly on book clubs that

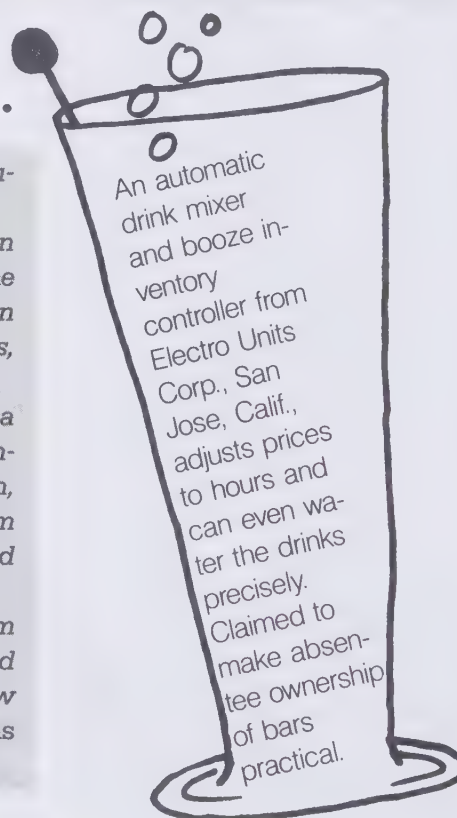
.....

*The problem with computer backups is usually that you fail to make them.*

*An opposite situation occurred, however, in connection with the Irangate scandal (the sales of arms to Iran to fund the Nicaraguan contras, as persued extensively in the press, 1986-7).*

*In mid-February 1987 it was revealed that a lot of possibly-incriminating memoranda concerning meetings of principals Oliver North, John Poindexter and CIA director William Casey, which supposedly had been destroyed by someone, had been recovered.*

*Someone had erased these memos from computer storage, but they were found stored on an automatic backup system. ("New Memos on Iran Deal Come to Light," Dallas Morning News, 14 Feb. '87, p.1.)*



ignored typed letters and kept billing me incorrectly.

Get a roll of white shelf paper, two or three feet wide and twenty or more feet long.

Write a letter on the shelf paper in magic marker. Make it big, perhaps six inches to a word. Legibility is necessary, but don't make it too easy to read.

Explain the problem clearly.

Now take your punch card—you did get one, didn't you, a bill or something?—and mutilate it carefully. Tear it in quarters, or cut it into lace, or something. But *make sure the serial number is still legible*. Staple it lovingly to your nice big letter.

Now fold your letter, and find an envelope big enough for it to fit in, and send it, registered or certified mail, to ANY HUMAN BEING, ACCOUNTING DEPARTMENT, or whatever, and the company's address.

This really works quite well.

I am assuming here, now, that your problem has merit, and you have been denied the attention required to settle it. If we want justice we must ourselves be just.

There is only one further step, but, again, to be used only in proportion to the offense. This step is to be used only if a meritorious communication, like that already described, has not been properly responded to in a decent interval.

We assume that this unjust firm has sent you a reply envelope or card on which they must pay postage. Now carefully drafting a follow-up letter, explain once again, in civil language, the original problem, your efforts at attention, and so on. Now put it in a package with a ten or twelve-pound rock, affix the reply envelope to the outside, and send it off.

The problem, you see, has been to get out of the batch stream and be treated as an *exception*. Flagrantly destroying the punch card serves to remove you from the flow in that fashion. (However, just tearing it a little bit probably won't: a card that is intact but torn can simply be put in a certain slot of the card-punch and *duplicated*. Destroy it good and plenty.)

In all these cases remember: the problem is not that you are "being treated as a number," whatever that means, but that your case does not correctly fall in the categories that have been set up for it. By forcing attention to your case as an exception, you are making them realize that more categories are needed, or more people to handle exceptions. If more people do this when they have a just complaint, service will improve rapidly.

*This section could be expanded to infinite length; but every other page of your favorite magazine probably has a New Computerish Item of more interest to you than the ones I'd dig up.*

## The Computer Graveyard



In the mid-sixties there was a junkyard in Kingston, N.Y. that was like an automobile graveyard—except piled high with dead computers.

They were from various manufacturers. The guys would smash them with sledgehammers, or other awful things, to make sure they could never work again. Then you could buy the circuit cards. I saw 1401s five high, Univac File Computers, tape drives...it was an electronic nut's paradise. You could decorate your den with huge old control panels, mag disks and whatnot. It seems to be gone now. They forbade pictures.

**Computers  
can be a  
tool for  
freedom  
or for  
oppression.  
What makes  
the  
difference  
is whether  
we speak up  
and work  
together  
for a  
world we  
want.**

—John S. James

## GREAT ISSUES

*There are plenty of issues and they're big, not just important but vast and sprawling. The biggest are:*

*PRIVACY, FREEDOM (now and later—and tomorrow's Information Wars about access);*

*IMPOSITION (and "We Know What's Right for You");*

*and, of course, there are War, Pollution and Starvation.*

*Computers are of course part of every problem, being used in big decisions, and increasingly the warp and woof of our decision processes; and computers are always being phonily blamed.*

*There is, as always, a lot of rhetoric in the computer field about "freedom," but the information freedoms are in danger everywhere in the world.*

Until now, the obscurity of computers has kept the public from understanding that anything like political issues were involved in their use. But now a lot of things are going to break. For instance—

*Do not take on faith any technical proposals you do not understand. They have often been whomped up by guys in the back room who had to think of something laid down by the front office under a deadline.*

*Supposedly Madame Tussaud's wax museum in London removed its figure of George Orwell in 1985. They figured no one would be interested in him now that 1984 was over.*



"The human mind is effective in identifying the structure of a complex situation. But human experience trains the mind only poorly for estimating the dynamic consequences of how the parts of a system will interact with one another."  
—Jay W. Forrester, World Dynamics

We just don't see things coming. With the proper visualization tools, for instance, people might have recognized sooner that AIDS, the new fatal venereal disease, threatens to kill as much as a twentieth of the U.S. population before the end of the century if we don't find a vaccine, a cure, or a lifestyle drastically different from any the human race has previously known.

## IMAGINING THE FUTURE, FACING THE COMPLEXITY

An interesting fable tells that the inventor of chess was offered anything he wanted as a reward by the then-Rajah.

"All I want," said the inventor modestly, "is one grain of wheat for the first square, two grains of wheat for the second, four grains of wheat for the third square, and so on, doubling the amount for each square of the chessboard."

"It shall be yours," said the Raj, not thinking ahead well. For he was promising two-to-the-sixty-fourth grains of wheat, or more than the value of the kingdom—a kingdom which the inventor of chess at last duly received.

This is an example of how hard it is to visualize compound interest and geometric progression. And this inability to visualize, to project ahead, is also behind the national debt situation, what we paid for the Vietnam War, what we as individuals pay on our charge accounts.

Among the priorities for tomorrow, then, are systems which allow us to visualize growth and complex interaction. These will be far beyond today's "business graphics," but will be highly-interactive visualization tools for quickly modelling and imagining.

Already computers have permitted us to visualize complex causations as we never could before (see "The Club of Rome," p. 172). Speeding such visualizations up on the personal computer is something we all need badly—and which our lawmakers, governments and presidents need especially.

AIDS is a case in point.

Acquired Immune Deficiency Syndrome (called SIDA in French-speaking countries) is a new, fatal venereal disease with a very long incubation period. It first appeared in the United States in 1978, in Los Angeles, among half-a-dozen homosexual men. The infections then proceeded to **double every six months**, among gay men, bisexual men and needle-drug users; yet the public in general, and the Executive branch of the government, were ever-so-slow to sense the menace, like the Rajah with his chessboard promise. Even though, estimated conservatively, there are millions of each of those three categories for the numbers to keep doubling into. Only in 1986 was it acknowledged that the disease could be spread by women as well. As of January 1987, there were some 30,000 developed

cases of AIDS in the United States, with one million Americans believed to be carrying and spreading the virus.

It was estimated that by January 1992, there would be some 270,000 developed cases (not counting a hundred thousand already dead by then).

This 1992 figure is the most recent public projection anyone has made. It was probably selected to be alarming but not **too** alarming; because if we take the figures along another five or ten years, the numbers are staggering.

No vaccine may be possible: this is a virus that moves through cell walls, where antibodies do not reach, and mutates faster than anything previously seen.

Middle-class Americans have changed their freewheeling sexual habits **somewhat**, but not much, according to Leishman's grim but excellent piece. Teenagers and the poor have changed their behavior little.

Result: it is regrettably safe to predict that the history of the 1990's will be the story of AIDS, and the upheavals to come will be like those of the Black Plague.

## BIBLIOGRAPHY

Katie Leishman, "Heterosexuals and AIDS: The Second Stage of the Epidemic." *The Atlantic*, February 1987, pp. 39-58.

## THE PROBLEM OF ESCALATING RECOURSE:

## AN OMINOUS PARADOX OF MOD- ERN MEDICINE

We think of the progress of medicine as all benefit. An opposite argument can be made.

As medical technology grows, the number of techniques, tests, and possible recourses for any given medical situation are growing without limit. There are more and more things to try, and our traditions demand it. The use of every medical advance becomes mandatory. For humanitarian, religious, Hippocratic and liability reasons, doctors are under pressure to use them **all**.

One reason is to give the best everyday medicine to those whose lives are likely to be restored. But some of these new techniques, ever more extensive, simply grant life extension to more and more marginal cases, keeping the human husk and marginal fetus alive. Many medical situa-

tions can be prolonged, some indefinitely, by pouring on resources. That number, and the amount of possible prolongation will increase and increase. As will the cost.

Genuine life-saving techniques, which can make people well and productive again, are one thing; but there are many other techniques that tend to stretch painful and unproductive life by marginal amounts. These are often called "heroic measures." This becomes harder and harder to justify from any non-religious point of view; but medical practitioners, acting according to the current standards of the profession, tend to prescribe more and more at whatever cost; and a family stretched thin emotionally can rarely turn heroic measures down.

A generation ago, 4.4 percent of the U.S. Gross National Product went to the medical sector. Now it is 11 percent, 360 billion dollars a year. How much of this increase is better medicine, and how much is superfluous recourse, is very hard to say. But like the enormous sums spent by the ancient Egyptians on burial, it is clear that these treatments are becoming a severe drain on the economy and the future, also (in part) for religious reasons.

Some parts of the system, like "free" care provided by municipal hospitals, are self-limiting because at any instant there are only so many beds, nurses and so on. At the individual level, more and more people are bankrupted at death, the savings they worked a lifetime for pointlessly vaporized at the end. This robs the young to keep the old alive a little longer, but no one dares say it's too expensive.

On the other hand, the medical insurance plans have had the resources and position to fight back. Medical plans are increasingly run on a blue-book basis, rationing treatment (regardless of particulars) to strict and simple schedules.

There is a new paramedical specialist, the DRG—a medical-records inspector, selected by doctors, who kicks people out of hospitals on the basis of their charts. (This in turn is to avoid problems with the insurance inspectors, who come around later.) But the DRG is the equivalent of the tax collector or hatchet man, making life-or-death (often deathward) decisions that are disguised as routine, by-the-book procedure, statistically justified.

This also creates the channels by which additional pressure and cutbacks can be applied at a later time, under budget pressure.

## NEW FORMS OF IMPRISONMENT

We take "freedom" for granted, or say we do. But computers make it possible to imprison people in many new ways.

**COMPUTER-ASSISTED INSTRUCTION** (see p. DM 129) is intended to make you wise, but those who design its sessions have a license to imprison you as no previous teacher ever could, keeping you from moving on, changing the subject or doing anything but what they have decided. I submit that this opens dastardly possibilities and that the user must always have the option of "turning the page," or its electronic equivalent. This is one reason I advocate the much freer hypermedia (see p. DM 64).

**KEYSTROKE COUNTING AND AUTOMATIC SUPERVISION** have become new issues in labor relations. See next column.

**HOUSE ARREST** with an electronic equivalent of the ball and chain is now a simple matter. (See p. 149.) The grave danger is that such punishments and Creative Detentions will be used with increasing casualness by arbitrary authority.

**DATABANKS** are, in a sense, the reading, writing and publishing systems of tomorrow. If we are not extremely careful, governments will find it all too easy to shut off citizen access to information by the electronic equivalent of turning a faucet (see p. 167).

## VERY ERGO

Ergonomics originally meant the economics of energy, but it's become an ideology and a labor issue, especially in Europe.

In offices where employees sit at screens all day using canned software, the employees get a lot of headaches, eyestrain and other unpleasant symptoms.

This has led to the creation of very expensive furniture (called "ergonomic" or "very ergo")—tilting seats and crank-up tables, and rules about distance-from-screen intended to make the employees as comfortable as possible.

### ...AND THE VDT

There's also been a lot of research about supposed radiation from the

screens (CRTs are called VDTs, Video Display Terminals, by the labor-relations people).

Since they're essentially TV sets, there's no radiation to speak of. The real point, I think, is that people get headaches, eyestrain, and a lot of other unpleasant symptoms on jobs they hate.

## CLOSE SUPERVISION

Computers now make it possible to supervise a clerk's work automatically, down to the last keystroke. And to monitor the movements of delivery people in their trucks, second by second. No more coffee breaks.

Management sees this as providing increased control. But studies have shown that keystroke-by-keystroke supervision makes some workers more tense even than air traffic controllers. This may be very self-defeating.

(When people say, "Computers are turning us into robots," this is what they mean. But it's not the **computer** that decides to put in oppressive systems. It's **management**.)

## CRACKERS AGAIN

(continued from p. 112)

Crackers, you may recall, are those who break into computers and otherwise act in a lawless fashion. Here are some interesting items:

## BLACK AND BLUE AND RED ALL OVER

The phone system is bruised and bleeding from the depredations of people who have found out how to cheat the phone company electronically. Such people are called Phone Freaks (or Phreax); articles on them have appeared in such places as *Ramparts*, *The Realist* and *Oui*. For no clear reason, the electronic devices they use have been given various colorful names:

**black box:** device which, attached to a local telephone, permits it to receive an incoming call without billing the calling party; it "looks like" the phone is still ringing, as far as the billing mechanism is concerned.

**blue box:** device that generates the magical "inside" tones that open up the



A Frenchman named Closier has pointed out that having computers tally what office-workers do increases the rigidity of the workers' behavior, since everything can be seen—and will presumably be judged on simple-minded criteria.

## A Curious Violation of Computer Privacy by Individuals:

## ELIZASCAM

The game of "Eliza" is famous in the computer field. A computer is programmed to play doctor, responding as if it were a real person. (See p. DM 124.) Used to fool a lot of people. (Thus in a way passing the "Turing Test" of intelligence; see p. DM 123.) But lately most everybody knows it's responding fakily, and chuckles at its tricks.

However, they rang an interesting change on this at Yale a few years ago. There was a room full of terminals where students were busily at work. Then unexpectedly, across all the screens, would flash:

## IS THERE A DOCTOR IN THE HOUSE?

First to say Yes would be put on-line to the Patient—a victim who thought he or she was talking to an Eliza-like program. And so the Patient would say all kinds of compromising things, anecdotal, painful, playful and sexual, to this supposedly unconscious entity, and the student who was Playing Doctor would alternate between innocuous and cut-to-the-bone replies—until the victim caught on and shut off.



## HOT SCREENS

In a minor scandal a few years ago, two employees quit a major newspaper when they found out that their fellow employees had been reading their love letters, which were in data files other reporters could reach. They thought the files were private, their fellow journalists did not.

Is this a grey area? I think not. The ethics of privacy in the western world are clearly understood, and indeed what the government may do to your privacy—at least the non-secret branches of the government—is quite circumscribed.

But individuals violate the rights of other individuals all the time. Not merely by such crimes as theft and spying, but now by new technological variants of these.

For instance, in the forties, when phones were shared in neighborhoods, it was commonplace to listen on the party line.

Today in the parking lot we may study others' cars, or glance at their garbage in the dumpster. But tomorrow, as computer networking grows, there may be a lot of snooping against private files just out of personal orneriness.

Do we need a new code of ethics for this? NO! When a U.S. espionage service was disbanded at the close of World War One, the rationale as stated by a public official was, "Gentlemen do not open other people's mail." This still seems a simple and excellent guideline. We forget it all too easily.

The bad feeling about copyright seem to date from Paul Terrell's trying to sell Bill Gates' 8080 BASIC at the Byte Shop for \$500 in 1978. Well, Gates parlayed that into a billion-dollar empire, so the resentment continues to smolder among the brilliant but impoverished.  
—(Moritz, The Little Kingdom)

phone network and stop the billing mechanism. Possession of a blue box can put you in prison.

As with so many things, the phone system was not designed under the assumption that there would be thousands of electronic wise-guys capable of fooling around with it. Thus the phone system is tragically vulnerable to such messing around. The only thing they can do is get ferocious laws passed and really try to catch people, both of which are apparently happening. Supposedly it is *illegal* to possess a tone generator, or to inform anyone as to what the magical frequencies are—even though a slide whistle is such a tone generator, and any engineering library is said to have the information.

*red box*: device that simulates the signals made by falling coins.

The fact the names of these devices are given here is not to be construed as in any sense approving of them, and anybody who messes around with them is a fool, playing with napalm.

Even if people were entitled to steal back excess profits from the phone company—the so-called "people's discount"—the trouble is that they mess things up for everyone. We have a beautiful and delicate phone system, one that stands ready to do wonderful things for you, including bring computer service to your home; even if, for the sake of argument, it is run by dirty rats, messing around with it is like poisoning the reservoir for everybody.

Some of the zappier new Urban Transit Systems give you a ticket with a magnetic stripe on the back. Each time you ride you must push the card into an Entrance Machine, which presumably does something to the stripe, till finally the ticket runs out and you have to pay more money.

Secrecy of the recording code is an important aspect of the thing. Indeed, waggish gossip claims that some such systems start with a *blank* magnetic stripe and just add stuff to it, meaning the card can be washed clean with a magnet by larcenous commuters. But this seems unlikely.

A recent bust in the Bay Area involved illegally recharging BART tickets—the renewable Rapid Transit slips with magnetic strips on the back—by copying from an Enriched ticket to a Poor one, just using magnetic heads and an amplifier.

## THE COPYRIGHT ISSUE

is still controversial among programmers (see Steven Levy's book *Hackers*). Here are some relevancies:

Now that Altairs and LSI-11s have got a lot of you guys dreaming about selling software, an important question is how to protect your work—ways that small independent underground-type programmers can protect their developments.

### COPYRIGHT AND COPYWRONG

One individual I know, who relishes his counterculture image, told me with angry and shaking voice that he doesn't believe in copyright and that anything that gets near his computer belongs to *him*. Well, don't leave your manuscripts near such a person. (Why is it always the guys with cushy and secure jobs who tell you twee-dee-dee, ideas should be free, and patents and copyrights are selfish?)

Actually, for the individual, one of the strongest forms of protection available is copyright. Far from obsolete, the copyright makes publishing, and the better computer software, possible. (It is not generally known that copyright violation is a *felony*.) (And ripping off a program you're supposed to pay for is not a brave guerrilla affirmation, like hitting Harold Geneen with a pie, but grand larceny.)

Incidentally, when you think something you've written belongs to you—a computer program, poem or whatever—slap the following at the beginning, under the title:

© 1975 IRVING SNERD

substituting, of course, your own name, and the year currently in effect. If computer printing is used, (C), using parentheses, is considered an acceptable substitute for c-in-a-circle.

This not only gives notice to potential Borrowers, but it has certain strong magical properties as a legal incantation. See your lawyer for details, but don't hesitate to apply it liberally to your own work; you may be glad you did.

### WHERE ARE THEY NOW?

I still run into that guy at conferences, and he still makes snotty little boy jokes about copyright every time I see him.

I think it's like kid's jokes about the bathroom. If he would learn about intellectual property I think it might relieve some of the anxiety and anger he seems to feel about it.

Programs are protected by copyright—that's the only way there can

be a software industry at all—but since there has been no court litigation in the field, nobody knows what the law really is or what it covers. Everybody agrees that traditional copyright precedent covers a lot of ground—“derivative works” definitely violate copyright, even study guides to textbooks—but no one knows how far this goes.

Same for patents. The Patent Office has granted program patents, notably the one on the sorting program of Applied Data Research, Inc., but the Patent Office has a profound distaste for this potential extension of its duties, and is telling everyone that programs aren't patentable, even though they clearly fall within its mandate as unique, original processes.

People who only read the headlines think that the Supreme Court struck down the patentability of programs. No such thing.

In this light the patents that the University of Utah has gotten on the halftone image synthesis programs of Warnock and Wylie and Romney (see p. DM 104) are of considerable interest.

These patents use the “software-as-hardware” ruse: the program is described in detail as taking place in a fictitious machine shown in many detailed drawings whose nebulous character is not readily seen by the uninitiated: events vaguely taking place in “microprogramamble microprocessors” have been neatly foisted on the Patent Office as detailed technical disclosure. It's a great game. The idea is that the claims are so drawn as to cover not just the fictitious machine, but any program that should happen to work the same way. But such approaches, though common to previous patent practices, have not yet been litigated in this field.

## CRACKING PROGRAMS

*At one level, program copyright is a big moral issue with earnest, concerned and indignant people on both sides. But on another level, it's a cat-and-mouse game of skill against skill, Mischief against Legitimacy.*

*Software publishers are continually figuring out schemes to keep you from copying their disks. (For instance, one of the protection mechanisms used on the Apple II is to position the disk-arm at nonstandard track positions, making bit-by-bit copy impractical.)*

*But for every protection scheme, a new unlocking program appears soon. (Like Locksmith for the Apple II and Copy II for the PC and the Mac.)*

*There are also quasi-legitimate cracker programs. For instance, Nostradamus offers a program that will supposedly let you run 1-2-3 or Symphony straight off a hard disk without needing the original in a floppy drive. Used for your own copy, this would appear to be wholly legitimate, and Nostradamus doesn't license the program to violators of Lotus copyright, but it clearly could be used to spawn copies illicitly.*

*Sprawling programs, programs that you can never have all of, represent another direction of copyright protection. The program stays off on time-sharing and sends you the part you need when you ask for it. This form of distribution should be much easier to control, since no user will get it all and bootlegging will be much more difficult.*

*I think tomorrow's Octopus applications will be more of this type. Handling big problems thoroughly will require hundreds of sections that come as the subscriber requires them.*

## COMPUTER CENTER POLITICS

*In the beginning, computer centers were created because it seemed reasonable. “After all, a computer is a big investment, it's more efficient to have just one.” Then IBM would make the sale and everybody else would be forced out.*

## Krag!

**Prolok Plus from Vault Corporation was announced as an exceptionally user-hostile copy-protection scheme. Attempting to copy it would not merely cause your computer to crash, but might scrog your hard disk (by erasure), damage your equipment, or place a 'worm' in your operating system to cause damage later. All this according to W. Krag Brotby, chairman of Vault Corporation.**

—(PC Magazine, 27 Nov. '84, p. 54)



# SYSTEMS PEOPLE

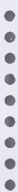
are the folks who bring you the computer.

That is, they're the ones who try to keep the operating system working. And make the changes it needs to adapt to new equipment and working rules and schedules and software. And change the parts through which mischievous users crash the system.

Systems people often look like dirty rats to users of computer systems. To each other they often look like harried, overworked, unsung heroes, their fingers (and whatever else) in the dike, trying to hold back the tide of Disorder.

Systems people deserve more thanks than they get.

Thank you, systems people.



This case is so classic, it's almost a Punch and Judy show.

One of the nastiest people I have ever met was the head of security for a big-computer installation. Several people agree with me that he delights in telling people they can't do specific things on the computer, merely for the sake of restricting them.

Anyway, at the same installation there was a programmer, let's call him A, who disliked authority, and disliked this director of security, let's call him B, with a moody passion.

B spent much of his time intensely, obsessively contemplating possible ways that users might break into the system, and elaborately programming defenses and countermeasures into the monitor. How do I know this? I know this from A, who constantly went through B's wastebasket. A still plans incessantly for the day B will get a big taunting printout, coming unexpectedly to him off the machine, that shows him all his secrets are known.

## WHERE ARE THEY NOW?

A has for a long time been a technical jack-of-all-trades; but I hear he has recently had bookings as a stand-up comedian. (As far as I know he never stuck it to B.) And last I heard, B had his own computer-security firm. Of course.



☛ Actually this probably was not true even then. Little computers in separate departments would have been superior in many cases.

But to create a department is to create a political entity with politics, and politicians and policies. A computer center has a Director and assistants, with jobs and an empire to defend. It has a bureaucracy with vested interests and rules.

The politics fit into the whole organization's politics: the computer center vies for funds and endeavors to provide services, but generally those services suited to its most important client departments. The less clout you have, the less service you get (facilities, scheduled hours, software, etc.).

A great show is made of "doing the best they can;" those who complain are sticking their necks out for counterattacks of cross-blaming.

Power plays involve the selection of computers and systems. Winners (whose choices get bought) get promoted, losers leave. Salesmen pull strings however they can, sometimes to get rid of people at their customer locations.

**Territory and empire** means suppressing insurrection and defending the borders: especially getting rid of complainers, and preventing the arrival of any rival computers not under the computer center's control that might take away from the center's mandate and lower the center's budget. When other machines do arrive, "territory" extends to making them dependent on the computer center, and preventing the development of any lines of communication that bypass the center.

**Budget fights** often involve the computer center saying it can provide no more services without more of your department's budget.

And making sure that everyone's needs for programs and equipment are funneled through a Purchasing Committee (which they control), effectively taxing all resources with the hidden overhead of "things that have to be gotten first."

Among such tangles, the computer center of your company or campus used to own you, body and soul. They told you when you could use the machine and for what, and even where to sit. And interac-

tive systems, which make every kind of programming and use easier, were postponed in most places until the late seventies.

So this is what the dinky-computer revolution in American business was really about. People were able to bypass this mess and simply buy their own, often through subterfuge. ("I'll sell you a computer under any name you like," said Portia Isaacson, and her computer store prospered.)

A pre-publication advertisement for a new magazine, around 1983, carried a headline for an article called

25 WAYS TO HIDE A PERSONAL COMPUTER.

But by the time the first issue came out, the article had grown to

50 WAYS TO HIDE A PERSONAL COMPUTER,

indicating how much the subject meant to people.

Now the floor is being pulled out from under these dictatorships. But they're fighting back every way they can, claiming they can Serve You Better if they control you. We've heard that one before.

#### FERNANDO FLORES' CONTROVERSIAL SYSTEMS

Marxist governments have generally made a terrible economic mess, producing badly and inadequately.

However, one of the more interesting aspects of Salvador Allende's Marxist government in Chile was to be a new factory-management system, using computers as a coordination mechanism.

Now in fairness it is reasonable to have certain of a company's databases centralized, making a single main picture of a corporations' life, a unified database. Somebody needs to get this up, as well as procedures for maintaining it, and this is a reasonable role for a computer center. But it is very easy for such systems and procedures to become oppressive to those who must work under them.

If it gets to the point where legitimate uses within the company become seriously hampered by bureaucratic rules, or alternative viewpoints cannot be considered, the company may be doing itself harm.

Under the system, factory problems would have been referred to larger and larger helping teams, communicating through the computer.

The system was designed by Stafford Beer of the London School of Economics, and Fernando Flores, a Chilean.

We don't know how well the system would have worked, since Allende was deposed by the Junta before it became operational.

Flores, now in the United States and sometimes associated with Werner Erhard, now has another piece of software. This one, curiously, has been called "fascist" by one irate critic. But Esther Dyson, a prominent software industry commentator, said no, it is completely neutral.

The system is called the **The Coordinator**, from Action Technologies. It is an electronic mail-and-message system that runs on PCs.

The interesting thing about it is that you are encouraged to categorize your messages in such terms as request and commitment. The system keeps track of all your commitments, when they are due, and which ones you have kept.

This formalizes a lot of ordinary relations in an office, making them visible and countable. Some people like it a lot, some don't. Its long-term consequences should be quite interesting.

#### DIFFERENT COUNTRIES

have some pretty strange ideas about data—especially about data coming and going across their borders.

CANADA officially charges duty on software coming into the country, though not on the actual disks. Sample dialogue:

**Customs inspector:** What's this?

**Friend of mine:** A floppy disk.

**Inspector:** Any software on it?

**Friend:** No.

**Inspector:** All right, then.

This is particularly ludicrous considering how much data and software cross the border on the phone. (Anticipating this difficulty, my friend had sent his programs ahead by modem. And what would be the "value" of the program he was working on?)

A computer center is like a locomotive yard. Big engines, whose use is centrally planned, are turned around and set to doing first this, then that.

Wanna ride? Study the schedule carefully, pack your bags, go to the station, buy your ticket, and wait where they say.

Whereas your personal computer is like a car or bicycle. No schedule, no waiting. Problems, sure, but they're all yours.

→ → → → → → → → →

**Computer center chauvinists are hardly ever aware of their biases. For instance, Datamation magazine (now mostly a sort of mainframe/computer center journal) featured the world of desktop machines on a cover in January 1987.**

**What was the title? "LIFE AT THE LOW END."**

**"Although the hottest buzz phrase of the 1970s was distributed processing, so sluggishly and tyrannically was the concept implemented—with plenty of help from IBM—that the stage was set for the PC revolution."**

—John Gantz in InfoWorld, 6 April 1987, p. 30.



EUROPE (especially Scandinavia) has various laws against records being kept in foreign countries concerning their citizens. This is understandable; their memory of the Nazis is long. But in a world where all sorts of records are kept, these laws are unwieldy and unenforceable.

BRITAIN has a "Data Protection Act," passed in November 1985, that requires all computer users to specify in a public file where they obtained personal information and how they intend to use it.

After "registration" of such information, users must work according to guidelines of the Data Protection Registrar, who oversees the Act. Failure to comply can result in unlimited fines. A **Data Protection Tribunal** oversees disputes with the Registrar. Every computer owner must register, at a cost of 22 pounds a year. (LM News: The Magazine of Direct Marketing, 1 March 1986.)

This grotesque business seems to have as its intent to protect people against false credit ratings and the like, but it would seem to apply to anything mentioning anybody on computers including diaries and personal correspondence mentioning other people. "I hear old Podger has been giving his cats kippers again," sent in a letter without having been previously registered, could subject the writer to unlimited fines. (UNIX HUMAN-NETS, a gossip newsletter distributed electronically to UNIX users, would seem to be a clear and continual violation.)

My impression is that they simply don't know what they're doing. Or is it a government ploy against investigative journalists?

"THE THIRD WORLD INFORMATION ORDER," proposed by the non-industrial nations, appears to be another lunacy. The two main provisions, as I understand them, are:

1. Data originated in a third-world country is controlled by that country.
2. Data coming into a third-world country belongs to that country, i.e., cannot be private or restricted to private purposes.

We can understand that these countries are seeking any possible leverage for their own future betterment, and with some historical cause are suspicious of foreigners. However, restrictive laws that propose to fish in people's pockets will

work against their own long-term interests.

In 1977, Martin Gardner announced in his Scientific American column that a team of MIT researchers had discovered a new method of creating unbreakable codes. And there hangs a story, and a lot of sinister implications.

### TRAPDOOR CODES, the GOVERNMENT SLAM, and the PHONY CODE

Two Stanford guys, Diffie and Hellman, had proposed a littler earlier a magical and symmetrical system of private communication, called a "public-key cryptosystem." It was beautiful in its symmetry and in the power and freedom it promised to individuals. Let me explain the general idea first.

We are all scared of being spied on by the government, by corporations, and by individuals. We want to be able to communicate privately. This means some kind of coding scheme. But most codes require **key exchange**. That is, you decide on a coding method, and then, with a private understanding as to the general method you will be using, you go in advance to the person or organization you want to exchange messages with, and you exchange keys. He, she or it gives you one key—that is, the decoding word or number you need to use for their incoming messages—and you give her, him or it the key they will need to decode **your** messages.

Now, this works fine if you will only be communicating with, say, your spouse of Home Office. But what if there are fifteen others you intend to indulge in private communications with? You must privately give your key to all of them, and get theirs. Just keeping track is a pain in the neck, not to mention how it gets if you want to give each of them a **different** key so they can't spy on your messages to each other.

Now imagine the problem if you engage in correspondence with **hundreds**.

Worse, what if you want to communicate privately with someone you have never met?

You could call up and tell the party your key over the phone. But the line might be tapped. So, uh...

Diffie and Hellman's idea cut through this marvelously. They proposed the idea of **public keys**.

The idea is to have **two encryption keys**. They need to be symmetrical. That is, you can code with key A and decode with key B, or vice versa. (But you mustn't be able to derive A from B or B from A.)

Now you **publish one of them**. You can publish key A in a big phone book or something, and **anybody can send you a private message**, by encoding their message using A. Only YOU can read it, however, because only you have secret code B.

Now you can reply to the letter, **also** with a coded message. But you encode the message with your **secret** code, B. Your correspondent now decodes it using the public number A. This acts like a signature: since only you have the secret code B, only you could have sent the message.

This system has several wonderful properties. One is that **strangers** can send each other private messages, using the book of published keys, with none of that key-exchange business.

## AUTHENTICATION

Another wonderful property of the public-key cryptosystem method is that it can present evidence on the origin of something, just as a signature does. The first example I gave already: encoding a document with your private key is evidence that you wrote it, since others may decode it with your public key.

But also, if certain mathematical properties are satisfied, **several** people can sign a document. Each of them encodes the result of the previous author's encoding. Now if that whole result is decoded in the same sequence, the original message is obtained.

That was the original proposal of Diffie and Hellman, but they had not actually created such a system.

This is what the MIT guys had done, as related by Martin Gardner in the Scientific American.

For the full information, said Gardner, you had but to write to one of the

In 1978, I believe it was, I got an astonishing phone call. It was from an officer of SHARE.

SHARE was the official IBM users' organization, which especially represented mainframe computer centers. SHARE had several functions: An obvious one was technical conferences. It was also a pressure group by which users could make their wants insistently known to IBM, and at the same time IBM could use it as a political arm in various ways, making sure such pressures didn't get out of hand. For instance, they could answer complaints with hints of future products and services, and reward IBM loyalty with various perks.

Anyway, the caller asked if I could fill in as keynote speaker for the scheduled semi-annual SHARE conference, replacing someone else who couldn't make it. He knew my point of view and thought this audience should hear it.

With pleasure, I said. As the title of my speech I gave him  
DINOSAURS UNITE! THE MICE ARE COMING.

A week or so later I received another call, this from a higher officer of SHARE, saying there had been a total misunderstanding, I had not been invited, the person who called had no authority, I had misunderstood the conversation, and the original caller would henceforth be unavailable for comment. Too bad. I would have told them, of course, that the new desktop computers would deeply threaten the computer centers and mainframe computing in general; that there would be a new spirit of freedom and initiative among the users thus liberated (and users newly hatched, never having known captivity); that this would bring an explosion of all kinds of computing uses the computer centers had suppressed—and had never imagined—uses they could now no longer co-opt and control; and that instead of manipulating a captive clientele, they would have to turn to providing the services people wanted in the most convenient ways—convenient, that is, for users, not for the computer centers. Their customers, no longer prisoners, would now have to be enticed to stay on—a rather different situation. Well, they may not have heard it from me, but since then they've found out anyway.





Data is so strong and yet so fragile. In the coming world, computers will be, more than any other instruments, the reading-and-writing and exploratory tools of individuals; transporting our personal lives into new realms of autonomy and understanding. To hinder what an individual may write and read by computer in the new era is to cripple at once all the freedoms that the Founding Fathers intended.

## Loused-up Records | A Case In Point

The question of "privacy" in the abstract isn't really an issue. Who cares if God sees under your clothes? The problem is what happens to you on the basis of people's access to your records.

Margo St. James is a case in point.

Ms. St. James is a celebrated West Coast prostitute, once well known for her activities with Paul Krassner as "The Realist Nun"; she is now Chairmadam of an organization called COYOTE, campaigning for the decriminalization of prostitution.

She originally had no intention of becoming a prostitute. Rather, she learned that there was a false record of her arrest for prostitution; and despite her efforts to clear her name, the record followed her wherever she tried to get a job. Finally she said the hell with it and *did* become a prostitute.

(Membership is \$5 a year. COYOTE, Box 26354, San Francisco, CA 94126.)

## Loused-up Records | Another Case In Point

The National Crime Information Center (NCIC) is the FBI's list of wanted persons. Listings arrive from all over, and have various problems. One study found a fifth of the records to be inaccurate.

For instance, Terry Dean Rogan lost his identity cards on a visit to Detroit in 1981. A year later, someone using those cards was wanted for murder and robbery in Los Angeles.

Rogan himself, probably innocent, has been repeatedly arrested and released over this matter, and has brought suit against the Los Angeles Police Department for failing to correct the record on the system.

But the FBI wants to enlarge the scope of the NCIC system, to list on it people who are not officially charged with crimes, but are merely "under investigation for white-collar crimes, or associates of such people." (David Burnham, "Victim Files Suit Over Error That Led To Five Arrests," New York Times, 12 February 1985.)

Why did they shorten the DES key to 56 bits? "Because it's easier to remember."

—Marianne Hsiung

researchers (Rivest, Shamir and Adleman) at MIT.

I did, and so did many others; but what we got instead of the paper was a note saying that for legal reasons they could not send the paper out.

What had happened was this:

As Rivest, Shamir and Adleman were preparing to send out the paper, they received a notification that they had better not. The order came from an official of the National Security Agency, J.A. Meyer.

Now everything hit the fan.

It seemed clear to a lot of people that the government (and the Spooks to boot) were attempting to thwart forever any hope of privacy and freedom in the future.

People sprang into action. Already a few copies of the paper by Rivest, Shamir and Adleman had been handed out; brave souls throughout the country got busy duplicating these, and sending them out to whomever they thought might care. Hundreds of researchers and concerned computerfolk got plain brown envelopes in

which the details of the RSA code (for Rivest, Shamir and Adleman) were explained.

Rivest, Shamir and Adleman did not get into trouble; and because that cat was already out of the bag, the government backed off, and allowed the paper to be published. But thereafter, under a "gentlemen's agreement," all articles on codes have been checked with government agents, and published only with permission.

So basically nobody in the academic world can publish articles on codes any more, and a spook from the NSA attends every session on Number Theory at mathematical meetings.

## THE DES, AN APPARENT FRAUD

But there is another code, one you can get cheap. It is the DES code, short for Data Encryption Standard, that the National Bureau of Standards says is just what you want for sending private messages.

But the obvious question is this: why should the government make it easy to get one coding system, the DES, and hard to get the other, if they are both secure ways of sending messages?

(Pause while you think.)

Well, everybody else thinks so too. It is widely believed—by computer scientists, that is—that DES is a simple fraud, created to look like a reliable and private code, but in reality a glass box to the National Security Agency; a "doormat" code, with the key right where anybody could find it if they knew.

Oh, they say DES is all very proper, and that it is merely a simplified version of the LUCIFER algorithm so carefully developed by many researchers, and the reason they shortened the key to fifty-six bits is that you don't really need that much protection, heh heh. And then there is the matter of the mysterious S-boxes, portions of the software which do certain permutations that have never been publicly explained. (If you run a Fourier analysis on something truly encrypted, the result is supposed to be completely random nonsense. Eric Gullichsen discovered, in his master's thesis, that if you did a Fourier analysis of DES inputs and outputs, you get interesting patterns.)

But the upshot is that the government has moved to prevent citizen access to decent coding systems, and left us with an

## TODAY NEW HAMPSHIRE, TOMORROW THE WORLD

New Hampshire, a small state, shows the data-bank problem in a microcosm, as reported in the New York Times, 6 May 1985 ("New Computer Splits New Hampshire Officials"). The legislature paid for a system to keep track of the state's funding, but Governor John H. Sununu would not give legislators access to its contents, saying that their desire for access was like wanting to go through his filing cabinets.

"Most people in the New Hampshire legislature just don't grasp the ramifications of this...the advent of the information revolution is sort of a double-edged sword, and the question is whether everybody's going to have access to information, or whether we're going to concentrate it in the hands of a few."

—State Representative Wayne King

"We don't need the scrutiny of people who might draw conclusions that are not in the best interest of the state."

—Steven M. Kennedy, State Director of Plant and Property Management



evidently fraudulent one which is proclaimed as "the standard."

The worst aspect of this situation has to do with authentication (see p. 163 and "Archives and Fraud," p. 167), for if we can no longer tell what documents are real and which are bogus, all of history will become a never-never-land run by whoever controls the intelligence agencies, just as Orwell foretold.

## BIBLIOGRAPHY

W. Diffie and M.E. Hellman, "New Directions in Cryptography." IEEE Transactions on Information Theory, November 1976, p. 644-654.

Ronald Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems." Communications of the ACM, February 1978, pp. 120-126.

The Puzzle Palace: The Story of the National Security Agency.

## RECENT THOUGHTS ON KEEPING YOUR TRANSACTIONS ANONYMOUS

David Chaum, "Security Without Identification: Transaction Systems to Make Big Brother Obsolete." Communications of the ACM, October 1985, pp. 1030-1044.

Chaum is an earnest guy who offers hope for decentralization, smaller organizations, and returning privacy to individuals—escaping our growing dossier society—by new kinds of "personal computer cards," changeable pseudonyms, and transaction brokerage houses.

He hopes to create a new symmetry: in the current computer world, organizations can protect themselves against individuals, but not vice versa. Chaum hopes to change that. (His system also relies on RSA-like codes.)

## THE DATABANK INDUSTRY

now has perhaps a million users in the United States, three thousand different databases maintained by thirteen hundred organizations, as well as federal, state and local governments and intelligence agencies. It is big business, and it has many political ramifications as well.

For all issues in the future may come down to this one: access by citizens to remote databases, and the freedom of citizens to publish them. The database (which includes text storage) will become

the new printing press, the new medium of research and interchange, and indeed the final archive for the written word. So that all a wicked government need do is cut off citizen access to these services, and all else is cut off at once. Those who want to diminish a political issue—for example, toxic waste—need only diminish the toxic waste **data** to prevail.

## "DATA BANKS"

The term "data bank" doesn't have any particular technical meaning. It just refers to any large store of information, especially something attached to a computer.

For instance, at Dartmouth College, where the social scientists have been working hand-in-hand with their big time-sharing project, an awesome amount of data is already available online in the social sciences. The last census, for instance, in detailed and undigested form. Suppose you're at Dartmouth and you get into an argument over whether, say, divorced women earn as much on the average as women the same age who have never been married. To solve: you just go to the nearest terminal, bat in a quick program in BASIC, and the system actually re-analyzes the census data to answer your question. If only Congress had this!

The usefulness should be evident.

Because of the way census data is handled now, it is not possible to ask for the records of a specific individual. But this kind of capability leads to some real dangers.

There is a lot of information stored about most individuals in this country. Credit information, arrest records, medical and psychiatric files, drivers' licenses, military service records, and so on.

Now, it is not hard to find out about an individual. A few phone calls from an official-sounding person can ascertain his credit rating, for instance. But that is very different from putting all these records together in one place.

The potential for mischief lies in danger to individuals. Persons up to no good could carefully investigate someone through the computer and then burglarize or kidnap. Someone unscrupulous could look for rich widows with 30-year-old unmarried daughters. Organized crime could search for patsies and strong-arm victims. ☛

## DATA-BANK ERRORS CAN BE A TWO-EDGED SWORD

The story of the kid who put his teachers in the criminal data bank (p. 112) is amusing in a retaliatory sense, but does not point any sensible direction for citizen defense from the government.

One of the things about the Cold War is that a whole lot of things have somehow been decided that aren't in the law anywhere. One of these is the assumption that the National Security Agency has a right to read anything they like. That is not in the Constitution. (In fact, the Supreme Court has even told us there is a right to privacy.)

Now, I'm not going to argue about whether this is right or wrong, but it is never debated, and THAT is wrong.

"One of the standards for accepting an encryption algorithm should be that no one knows more about it than you have access to."

—Mark S. Miller

.....

The RSA method had to do with the factoring of large numbers.

It seems that if you multiply two very large prime numbers (say, a hundred digits each—there are lots of primes up there), you get another very large number which is very hard to factor. Since it is the product of two primes, there is only one pair of numbers of which it is the product, but those two numbers are very hard to find.

Since publication on this matter is essentially forbidden, you hear a lot of gossip about "boles" in the RSA code and whatnot. And recently they supposedly factored a 150-digit number into constituent primes on a Cray. (David S. Fry, letter to the editor of InfoWorld, 23 March 1987.) So RSA may not be the best method—but it should do for a while.





A facsimile edition of "1984" is available in Orwell's own hand, reproduced from his own manuscript—or SO THEY WOULD HAVE US BELIEVE.

In one of the Oz books, an army of generals was led by a certain "Private Files."

*The National Archives (with 3 billion pieces of paper), faced with the crumbling of much of the paper, recently rejected the notion of computer storage:*

*"Magnetic recording tape and disks last perhaps 20 years, and today's computer may easily be outdated by then."*

*This shows a complete misunderstanding of the possibilities of digital storage, biased by their paper experience. The important thing about magnetic storage is not how long the individual medium lasts, but the fact that it may be copied and backed up with total flexibility onto whatever new media come along, forever. Of course the paper should be preserved. But the sensible use of computer storage will be a vital aspect of tomorrow's archiving.*

☛ In the face of this sort of possibility, computer people have been worrying for years; noteworthy is the study by Alan Westin that originally sounded the alarm, and his too-reassuring follow-up study of some data-gathering organizations (see bibliography). But the scary data banks, the ones that evidently keep track of political dissenters, aren't talking about what they do (see Schwartz piece).

Basically, the two greatest dangers from data banks are organized crime and the Executive branch of the Federal Government—assuming there is still a distinction.

It may seem odd, but Nixon has said he is concerned about computers and the privacy problem. Cynics may joke about what his concern actually is; but a more credible stand was taken by Vice-president Ford at the 1974 National Computer Conference. Ford expressed personal concern over privacy, particularly considering a proposed system called FEDNET, which would supposedly centralize government records of a broad variety.

Not mentioned by Ford was the matter of NCIC, the National Crime Information Center. This will be a system, run by the FBI, to give police anywhere in the country access to centralized records. THE QUESTION IS WHAT GETS STORED. Arrest records? Anonymous tips? (It would be possible to frame indi-

viduals rather nicely if a lot of loose stuff could be slipped into the file.)

Many people seemed to be concerned with preserving some "right to privacy," which is certainly a very nice idea, but it isn't in the Constitution; getting such a "right" formalized and agreed upon is going to be no small matter.

But that isn't what bothers me. Considering recent events, and the character of certain elected officials whose devotion to, and conception of, democracy is lately in doubt, things are scarcely as abstract as all that. Considering how helpful our government has been to brutal regimes abroad—notably the Chile overthrow, which some say was run from here (and which used sports arenas for detention just as John Mitchell did)—we can no longer know what use any information may find in this government. Tomorrow's Data Bank may be next week's Enemies List, next month's Protective Custodial Advisory—and next year's Termination List. (I don't know if you saw Robert Mardian's eyes on the Watergate hearings, but they chilled my blood.)

#### BIBLIOGRAPHY

- Heather M. David, "Computers, Privacy, and Security." *Computer Decisions*, May '74, p. 46-48.  
 Alan F. Westin, *Privacy and Freedom*, 1967.

Alan F. Westin and Michael A. Baker, *Databanks in a Free Society: Computers, Record-Keeping and Privacy*. Quadrangle, \$12.50.

"Landmark Study of Computer-Privacy Problems Completed." CACM, Dec '72, 1096-7. Complacent review of Westin & Baker.

Herman Schwartz, review of Westin & Baker book *NY Times Book Review*, 8 July '73, 19-20. Notes that the optimism of Westin and Baker is based on their ignoring various "much-feared information centers" already maintained by the government.

Stanton Wheeler (ed.), *On Record: Files and Dossiers in American Life*. Russell Sage Foundation (NYC), \$10.

"How Fair Are Those Fair Credit Guides?" *Datamation*, May '73, p. 120-124.

Phil Hirsch, "Computer Systems and the Issue of Privacy: How Far Away is 1984?" *Datamation*, Dec '72, p. 90-93.

D. Burnham, *The Rise of the Computer State: the Threat to Our Freedoms, Our Ethics and Our Democratic Process*, Random House.

*Privacy Journal*, a monthly newsletter on problems of privacy, many or most of which involve computers. P.O. Box 8844, Washington, D.C. 20003; \$15.00 a year.

## Adversary Databases●

Some database systems contain material about people who wish it weren't there. These include TRW Credit systems, the principal bearer of bad tidings about people's credit, and the National Crime Information Center, or NCIC, which keeps tabs on suspects at large.

EVERYTHING  
IS DEEPLY  
INTERTWINGLED.

## IMAGINE IF THE WATERGATE MOB

had had control over national data banks. Enough said.

*That's what it said in the old Computer Lib. It was all too prescient.*

*This is the story of an apparent Iran-gate conspirator who (with confederates) struck without warning at the heart of American freedom. And lost. So far.*

### THE POINDEXTER DIRECTIVE

On September 17, 1984, President Reagan signed a secret directive drafted by Rear Admiral John W. Poindexter, his national security adviser. It was officially entitled "National Policy on Protection of Sensitive, but Unclassified, Information in Federal Government Telecommunications and Automated Information Systems." The computer industry now calls it the Poindexter Directive.

In 1985 it was made public, but (like thousands of other government statements) caused no stir. It provided for the restriction of access to data banks by people in foreign countries—not just from the Eastern bloc, but including our supposed friends in places like France, Japan and Canada.

But nobody heard about it for a year or so. It was dropped as a bombshell on November 11, 1986, at a New York meeting of the Information Industry Association, a trade group for vendors of

privately-run data banks. A civilian from the Department of Defense named Diane Fountaine, sitting on a panel, announced the database industry was suddenly to be drastically altered by new government limitations. Among other things she said:

"The question is not will there be restrictions or controls on the use of commercially available on-line data bases—the question is how will such restrictions or controls be applied?" (AP dispatch, printed 13 November, 1986.)

Ms. Fountaine explained that from now on the use of some databases was to be restricted, though in ways not yet decided.

Most disturbing, however, was the notion that the restriction involved a new category—"sensitive but unclassified data," an elegant phrase for material that had been freely available (unclassified) but now was to be suddenly restricted.

The definition of "sensitive but unclassified," as set forth by Poindexter, was a rat's-nest of criss-crossing definitions so broad that basically it could include **any information whatever**. (See below.)

"Sensitive" could be anything a government department head wanted to suppress. For the authority to decide what was "sensitive" was to be delegated, severally and independently, to department heads throughout the cabinet, the intelligence agencies, the Department of Defense, and even the General Services Administration. ■

# Archives And Fraud

We are entering the age of digital archives. And quite rightly: for digital storage, compared to anything else, is far more reliable, delivers far more quickly and makes highly accessible vast archives of writings, history, science and art to everyone. However. Stuff stored on computers is streamlined. It has no watermarks, no signatures, no postmarks, no antique paper, no patina; all the evidence by which we esteem an object to be old, authentic, official, real.

Photographs stored on computers can be fraudulent, especially with the new image-synthesis techniques. The only way to authenticate a photograph as real in the future will be to believe the guy who says he took it. (In Woody Allen's movie "Zelig," the title character (played by Allen) keeps appearing in historical footage among Great Personages.)

In George Orwell's Nineteen Eighty-Four, the hero works at the Ministry of Truth, where archives and books are continually falsified to reflect the ruling party's current official view of the world.

Digital archives promise, on the one hand, enormous and inexpensive new access to text, pictures, sound and the world's treasures, for everybody, immediately.

On the other hand, digital archives threaten (as Orwell foretold) to become the only available repository of an ever-changing, ever-more-false system of manipulated history, where no one has access to the truth any more, not even the hidden rulers of the intelligence agencies and the Party.

The changes to history made by each generation of Party and intelligence bureaucracy appear as fact to the next, so each wave of changes obliterates another part of what was true and what was false.

For these reasons, systems of authentication and proof which are not under the control of the intelligence agencies represent the only hope for freedom of information a century from now. And for these reasons, the current arrangement whereby the National Security Agency prevents the development of real authentication methods, not under their control; is one of the greatest threats to our future freedom that we face.

*The Latest in All-Inclusive Doubletalk*

## "SENSITIVE BUT UNCLASSIFIED INFORMATION"

"Information, the disclosure, loss, misuse, alteration or destruction of which could adversely affect national security or other federal government interests."

"Those unclassified matters that relate to national defense or to foreign relations of the U.S. government."

"Those which are related to, but not limited to, the wide range of government-derived economic, human, financial, industrial, agricultural, technological and law enforcement information, as well as the privacy or confidentiality of personal or commercial proprietary information provided to the U.S. government by its citizens."



Who was not to get this information? Well, foreigners and **anybody who might pass it on to foreigners.**

At about the same time, a list came out of the sorts of folks who might pass information on to foreigners—firms who were not supposed to be given information by NASA anymore. The list held 33 companies and one individual, a business-school professor. Various of the prohibited firms were quite prestigious, including **University Microfilms**, a subsidiary of Xerox and the principal manager of the archives of academia and the popular press.

Meanwhile, in the fall of 1986 various database companies were visited by

agents from the FBI, the CIA, the Air Force and other intelligence agencies, asking what sorts of information they had stored and what means could be used to identify and restrict dial-in users.

And just to make it clear what would happen if the databank vendors didn't cooperate, they were given sample contracts making it plain that government information would no longer go in their databases if they didn't play ball. That's **any** government information, such as small airplane crash statistics; wheat production statistics, broken down by state and county; population figures and their breakdown by location, age and sex. . . . So it wasn't to be just government databases that were to

be restricted and monitored. It was **all** databases.

The heavy-handedness and stupidity with which the Poindexter forces handled the whole thing were fortunate indeed. Clearly, the military minds who were involved thought that civilians could just be given orders. Fear was used as a second motivator.

But these coordinated steps to intimidate the industry aroused an equal and opposite coordination.

Towering outrage swept the computer industry, which united to fight the directive. The notion of "sensitive but unclassified" material clearly struck at constitutional freedoms. Hearings were

held, the American Civil Liberties Union testified against the restrictions.

Now comes the picture-book happy ending. Admiral Poindexter resigned in the wash of the Irangate scandal. Summoned before the subcommittee on national security of the House Government Operations Committee and asked why he had done these things, Poindexter took the Fifth Amendment. And it was announced that the new national security adviser had rescinded the directive.

And the database industry, now alerted, will be watchful against similar moves in the future.

## So What If They Make It Harder For Us? WHO CARES ABOUT DATABANK ACCESS?

You'd better.

There are lots of reasons.

1. The vendors don't want to lose business. Yeah, but there's much more at stake than that.

2. The straight rational-economic: it does us more harm—to our economy and our research—to restrict information than it does good to restrict our potential enemies.

2a. They'll get it anyway. Henry Eric Firdman, a Russian emigre and former Russian computer developer, was asked what efforts the Russians would make to get specific information. His reply: "The effort is as strong as is necessary to get the information, it is as simple as that." (Quoted in MIS Week, 2 March 1987, p.14.)

2b. Ramifications of clumsiness: such programs would be "administered by a bureaucracy who are likely to be quite insensitive to researchers' needs." (Gerjuoy, p.34.)

3. A new in-between category is a bad idea. Classified data is available only to certain people; unclassified material is available to anyone else. That is simple and straightforward. A new middle category would be slippery, changeable, and subject to abuse. (Those who classify information are motivated to over-restrict; thus it would be a tightening noose, cutting off more and more people from more and more information.)

4. Problems of democracy. Governments will always have pretexts for suppressing what they choose. The question is whether our institutions can withstand future misuses of power. Freedom of research has been and must remain a principal defense of our other freedoms.

"Restrictions to access of unclassified information cuts at the very heart of the First Amendment and Americans' right to know."

—Kenneth B. Allen, an officer of the Information Industry Association (quoted in MIS Week, 23 February 1987, p.14).

5. Hopes for the information future. If, like so many computer people, you look forward to a day of information plenty for everyone—copious access to the world's libraries and archives, through computer screens, at negligible cost, as the great hope of humankind—then government moves to choke off such access, for only marginal reasons, are horrific, revolting and damnable.

"In such a climate freedom of information, and all other freedoms related to it, will continue to be in the greatest peril. Only the most strenuous exertions by those who understand the degree to which information is vital to our liberty will preserve it against those who think secrecy is vital to our safety."

—The Freedom of Information Committee of the American Society of Newspaper Editors, in the New York Times, 21 April 1955.

When the Founding Fathers established Freedom of the Press as part of the Bill of Rights, they did not mean freedom only of the press, but freedom of the only information storage and transmission medium they knew. Databases and computers are our new forms of speech and printing, protected by our elemental Constitutional rights.

## THE ABOVE STORY

is neat, uplifting, and all true. Unfortunately for the elegance of the tale, there's a lot more to it.

The Reagan administration began its information isolationism much earlier. Lo and behold, look what a concerned physicist said in 1982, **four years** before the shock of the Poindexter Directive, in the Bulletin of the Atomic Scientists:

"...the government apparently is proposing to create an ill-defined vast new category of unclassified yet restricted information, open to all American citizens but closed to foreigners without Federal authorization." (Gerjuoy, p. 33.)

So the grotesque Poindexter doctrine, with all it implies for the crippling of discourse within our republic, by complex barriers aimed at foreigners, was percolating in the interior of government long before. Can you imagine having to check whether each person you talk to is in turn on a list of people who talk to foreigners?

Let's go back further.

Organizers of a conference on Bubble Memory in Santa Barbara in February of 1980 were shaken by a letter warning of criminal penalties if Eastern bloc delegates attended, and potential fines for the individual organizers of **five times the value of any equipment demonstrated.**

So various people had to be disinvented, and a new pall lay upon the scientific community.

Admiral Bobby Inman, then Deputy Director of the CIA, underscored this in a most peculiar fashion, in a speech to the American Association for the Advancement of Science in early 1982. He accused the scientists of being **hypocritical** in their outrage over the suppression of free publication and meetings, since those who worked with funds from industry were quite used to secrecy and compartmentalization. The academic scientists, of course, thought this a strange and narrow viewpoint. (But in an odd turn of events, Inman later went on to found MCC, a think-tank in Austin—sponsored by many companies—which is perhaps the **most secret-compartmentalized research firm in the world.** So perhaps that is his inner obsession.)

And even before that, during the Carter administration, it was Inman who moved to suppress the trapdoor codes (see p. 164).

What am I getting at?

Deep and mysterious factions want to suppress free publication in this country for potential Cold War benefits in slowing down the progress of other nations. (The others we want to slow down may even include our allies, who are our technological competitors.)

"A lot of members don't feel the government can control information, but again, that is what they thought in Europe—until it was too late."

—Anonymous member of the Information Industry Association, quoted in MIS Week, 9 March 1987.

**The point of Orwell's Nineteen Eighty-Four was that they made people give up. Don't.**

**NECESSITY HAS BEEN THE EXCUSE FOR EVERY INFRACTION OF HUMAN FREEDOM. IT IS THE ARGUMENT OF TYRANTS: IT IS THE CREED OF SLAVES.**

—EDMUND BURKE

**It seems there are all sorts of areas where our Cold Warriors want to hamper our ability to function at home in order to spite The Enemy. ("If they could declare our local road maps secret, they would," says a friend of mine. "After all, they might someday help a potential enemy.")**

**But if you can Go Quantitative, and show in concrete terms what harm a Cold War policy is doing at home, things can get better quickly.**

**For instance, we used to have an asinine policy of not selling personal computers to the Soviet Union—even though the personal computer would be the most freedom-provoking, dissent-building tool we could possibly distribute to them—it's like handing out printing presses, for God's sake!**

**Anyway, somebody came along and did a study showing that not selling personal computers to the Russians was depriving the U.S. of nine billion dollars in business annually. Poof! There went the inane concerns about "security," and our salesmen were over there in a flash. ("Special Report: Washington's Reaction," Datamation, 15 March 1987, p. 56.)**



## HOW IT IS IN RUSSIA

**"If you want to copy an American article—one already published in the United States—you have to go to the deputy director in charge of security and get his written permission," said a Soviet scientist. "It can take two or three days."**

—United Press International dispatch, 5 January 1986.

**The only thing necessary for the triumph of evil is for good men to do nothing.**

—Edmund Burke



"Now  
we're all  
sons of  
bitches."

—J. Robert Oppenheimer, 1945

Now, it is all too easy to suppose that there are dark forces within government, and especially the intelligence agencies, forces eager for any opportunity to reduce our freedoms, to create new laws to restrict us, to create new jobs for themselves suppressing, harassing, intimidating, arresting, eavesdropping; striving to shake their last light fetters, so their adventures and assassinations can go unchecked, so their supervision of our lives and interference with everything can become unbounded, and their hidden gangs can vie for control of nation, palace, company and racket, so they can clasp the world in eternal night.

But it may not be as simple as all that. These men from the Powers of Darkness may be fighting entirely different wars in their minds.

Well, we can put the argument for freedom in Cold War terms. The argument for freedom is simply that it benefits *us* to be free more than it benefits *them*, whoever they may be.

But to put it in Cold War terms is morally wrong and stupid, and lowers the argument to the level of the adversary. We need freedom because it is *right*.

#### BIBLIOGRAPHY

N. Dorsen and S. Gillers, eds., *None of Your Business: Government Secrecy in America*. Viking, 1973.

Christopher Bine, "Admiral Inman's Tidal Wave," *Bulletin of the Atomic Scientists*, March 1982, p. 3.

Edward Gerjuoy, "Embargo on Ideas: the Reagan Isolation," *Bulletin of the Atomic Scientists*, November 1982, p. 31-37.

Bill Dooley, two-part series: "Unsecrets' Data Curbs Worry IIA," *MIS Week*, 23 Feb 1987, p. 1; and "Some High-Tech Information Too Easily Available?," *MIS Week*, 2 March 1987, p. 11.

Bill Dooley, "IIA Rips Poindexter Controls At Security Bill Hearing," *MIS Week*, 9 March 1987.

Bill Dooley, "Reagan Axes Secret Edict," *MIS Week*, 23 March 1987, p. 1.

#### "WILL IT BE PEACE, OR WAR?"

(See p. DM 123.)

#### BUTTON, BUTTON

At least, we like to think, the great computer systems which oversee our national defense are the best and the latest and the safest.

Not so, it turns out.

In a chilling two-part series, the New Yorker recently exposed the obsolescence, undependability and overall crumminess of the command-and-control computers at SAC. The fictitious computer in "War Games" was top-of-the-line compared to the reality. Because of inter-service rivalry, according to the author, flaky old vacuum-tube machines (which work only sometimes) run the system, and the software hasn't been updated for a long time either.

And when a general attempted to demonstrate the Hot Line, the line was dead.

#### BIBLIOGRAPHY

"The Button," *New Yorker*.

Ron Rosenbaum, "The Subterranean World of the Bomb," *Harper's*, March 1978.

## Disarmament And Negotiation ▼

"I have a dream..."

P. My feeling frankly is this. That you know I was just thinking tonight as I was making up my notes for this little talk, you know, what the hell, it is a little melodramatic, but it is totally true that what happens in this office in the next four years will probably determine whether there is a chance, and it's never been done, that you could have some sort of an uneasy peace for the next 25 years.

E. Uh huh.

(Nixon to Ehrlichmann. Apr '73.)

Thank you, Mr. President.

Most of the "national security" secrets that have come out in the last two decades have been concerned, not with the mechanics of our defense, but with systematic misconduct, lying and coverup by the U.S. government, as in the Pentagon Papers, the Watergate tapes, and the Irangate affair.

## A Note To The Intelligences In The Intelligence Agencies ▼

Hi there.

I presume you believe in what you're doing.

And I presume that you, too, want to see beyond this temporary mess that the world is in, fighting over dumb political and economic systems, to cope with the real messes we are about to face around the corner, at the turn of the century. Population. Resources. Getting space colonies no matter what. Making sure of the survival of the planet. Making sure of the survival of this our race, ratty and stupid though we often are, because we have a lot to offer. To each other, to be sure, but who else is there, right?

All I'm saying: there is a time to be serious beyond serious, beyond departments and beyond turf and beyond slogans, and to figure out what is going to make it all work. Not just for "our side." Because in the long run there is only one side.



# THE RAY-GUN INITIATIVE

also called

"Star Wars"

SDI,  
the Strategic Defense Initiative  
the Upholster

and

Hogwash.

Many in the new military see space not as the High Frontier, to be colonized for the betterment of all, but as the High Ground to be seized—the Astroturf, as it were, onto which the Cold War can be extrapolated.

So the news of the last few years has been busy with "Star Wars"—SDI, the Strategic Defense Initiative—which is not a planned system but a dream, something that a lot of companies and their engineers think they can build, if given enough money and time. (Though most American scientists think it's nuts.)

There are hundreds of possible components, each unproven, that their supporters think could do "part of the job"—X-ray lasers, rail guns, particle-beam weapons, Battle Mirrors—and an underlying credo that it can all be made to work. Reliably.

This will need computers and communications as yet undeveloped, running new software of a size never seen before for its "battle management system." And since no human programmers can design or understand software on such a scale, supposedly "artificial intelligence" will tie it together—i.e., since we can't do it and we don't understand it, let's leave it all to something else we don't understand.

In other words, the plan is to put every far-fetched gimmick we can name into space to save us, after we've just, uh, made it operational.

There is much room to be dubious.

In 1985, the "Sergeant York" cannon, designed to hit low-flying aircraft, was scrapped because it couldn't. In 1986, the space shuttle program was grounded because the Challenger disaster made it clear that the system was not reliable. And yet some people believe that a system with hundreds of different components, some

themselves as complicated as the shuttle, can reliably knock down 90% of a thousand incoming missiles on the first try, under an untested computer program with ten million lines of code. Clearly this is not technology but religion.

In the late sixties there was something called the ABM, with a similar intent. What the original Computer Lib said about that really tells the whole story.

"And the rocket's red glare,/The bombs bursting in air/Gave proof through the night/That our flag was still there.

"Oh say, does that star-spangled banner yet wave/O'er the land of the free and the home of the brave?"—E.S. Key

## THE ABM

Its name has kept changing, possibly to lull the public, possibly to gull the Congress. **Anyhow, would you believe a system, totally controlled by computers, designed to shoot down oncoming missiles?** If you would, read on.

It's been called Nike-X, Safeguard and goodness knows what. (It's even been called a "thin shield"—masculine, huh? Perhaps Congress would pay more if they called it the Trojan 4X.) But generally we refer to it as the ABM (Anti-Ballistic Missile). It's the anti-missile missile people have talked about, and in it lie many interesting morals, possible comparisons, etc., for which there is no space here.

Western Electric is the prime contractor. They're the manufacturing arm of the telephone company, remember, the same people who make the Princess™ phone. Of the hundreds of millions of dollars they are taking in on this project, much of it has to go back out—to Univac, which makes the computers; to Bell Labs, which guides the project, whose Whippany, NJ facility is totally given over to it; to the rocket-builders and so on.

The system is a turkey.

Note that in telling you this I am drawing only on information that is publicly available, and drawing conclusions from it the way one usually draws conclusions.

Here is how the great ABM is supposed to work.

Immense radars scan over the horizon looking for possible reflections that might be intercontinental missiles.

The radar images are forever constantly analyzed by computers, using every trick of Pattern Recognition (see p. DM 121).

Aha! Something is coming.

Yes, yes, I'm quite sure now, says the computer. We have fifteen minutes.

Great doors swing open, and a long phallic shape arises. It has jagged angular fins, inherited from the smaller anti-aircraft Nike (we say Nikey) rockets that preceded it. This missile is called the Spartan.

It takes off.

The computer system is tracking the oncoming missile. Here it comes—it's dodging now—the Spartan is turning, going faster and faster—they're coming together—

Oncoming missile speed: maybe 15,000 miles an hour. Spartan speed: maybe 10,000, who knows. In these few minutes the Spartan has gone 400 miles.

How's your tennis?

Can you hit a tennis ball fired out of a cannon?

But now comes the good part.

The Spartan goes off. Yay! It too contains an atomic bomb.

If it goes off within five miles of that attacking missile, the hope is that the attacking missile's thermonuclear warhead will get heated on one side and misfire. So it lands in Times Square, just breaks a few buildings and spreads radioactive contamination.

But wait.

What if Spartan missed.

Oops, sorry, Montreal.

Never fear! Have you forgotten Sergeant York? Have you forgotten the Alamo?

There is another missile. It is called Sprint. It is shaped like the point of a pencil. It is almost all propellant. When the great computers realize that the bad guy has gotten through, up goes Sprint! Sprint is eloquently called the "terminal defense system." It only has a couple of minutes.

Brighter than a thousand suns! Sorry, Scarsdale. Can't win 'em all.

If you find this description mind-boggling, that's because it is. Anybody who imagines that this project, on which billions of your dollars have already been

spent, can work, is a wishful thinker indeed.

Even if missiles stayed like they were in the good old days of 1962, big helpless clunkers they had to fuel up just before the shoot, the likelihood of the 5-mile ABM detonation they count on was pretty low. (Supposedly ARPA was hoping that Spartan and Sprint could be replaced with ultrapower, fry-in-the-sky laser beams, zapping down all comers with sky-piercing stabs under computer control—but that is said to have been abandoned.)

But even given, and only for the sake of argument, the feasibility of Spartan-Sprint for fish-in-a-barrel shots, look what's happening now.

## MIRVs and FOBs

MIRV (Multiple Independently Targeted Re-entry Vehicle) basically means Multiple Warheads. One rocket can carry all these little guys, see, that fan out when it gets near the target, and each one goes

## How The Right Wing Imagines It

"...President Reagan's March 1983 call for a defensive Space Shield against nuclear missiles knocked the rhetorical props out from under the anti-nuclear movement for good."

—syndicated columnist William Rusher, May 25, 1986

Clearly the stage is set. When this nonsense is eventually scuttled after the waste of a few billion dollars the right wing will be able to claim that it could have worked, had not the noble SDI been "stabbed in the back" by the politicians.

The best thing on anti-missile defense was a piece in MAD about 1963, which showed armadas of retaliatory and countermeasure cutlery that would supposedly fly back and forth in space, finally making the Eskimos rich when sold for scrap.



to its own target city or installation. FOB, or Fractional Orbital Bombardment system, just means that they send the thing into an orbit around the world, and the warheads come in from the opposite side. *Any side.* Meaning that all those radars pointed at Russia would make good drive-in movie screens.

ABM is sort of a dead duck: the one face-saving installation is in North Dakota, and there won't be any others. But one wonders how such things could ever be funded. But then again I remember once hearing Eric Sevareid, whom some call a liberal, pontificate on this subject. "They describe it as a 'thin shield,' (he said). Why can't we just spend a few billion more and get *complete* protection?" Otherwise canny people, if fooled by the technologists, will believe anything.

But the ABM is a beautiful example of top-down planning—like the Vietnamese war. I imagine that the Sprint came about something like this:

"Garfield, our people in Operations Research have concluded that Spartan won't work."

"Mmm, yes, sir."

"Garfield, I want your team to get on it and find something *additional* that will make it work."

*Of  
COURSE  
we can  
build a  
tapioca-  
powered  
rocketship!  
We've  
ALREADY  
GOT THE  
TAPIOCA!*

Now goes Garfield to his cubicle and calls meetings, and it becomes clear: "Lessee now, I can't just say it'll *never* work, they want something additional, well, I guess it would have to be..." Same as Vietnam. "Gee whiz, they say to search and destroy, I guess that must mean..." Something new, this: the top-down project of the worst sort, where the orders go down, and only news of partial success goes up, rather than the facts of total hopelessness. As in Vietnam.

The sophisticated argument is that the ABM effort lets our nation "keep its hand in," "sharpen skills," in case something vaguely like this is ever really needed—and possible. But this overlooks the overall strategic problem. All this foolishness leads away from the stability of the deterrent; and that may be what keeps everybody alive.

(An interesting point to note: a biologist and population geneticist named Sternglass claims it doesn't matter: that human reproduction is so susceptible to radiation poisoning that just the fallout from the ABM defense itself—a few dozen bombs, say—would end human reproduction around the planet. But nobody listens to Sternglass.)

Incidentally, an illustrious computer person, Rev. Dan McCracken (author of good programming texts on most of the major languages) goes around lecturing on the futility of the ABM system.

The main reason computer people should take an interest in this is simple. Only we know how funny the thing really is:

**All those computer programs have to work perfectly the first time.**

*In the opinion of the author, the ABM was an asinine fraud in 1968 and SDI is a bigger asinine fraud in 1987.*

*But GUESS WHAT! Most of the different space weapons they're talking about would really be more effective against targets on the ground than against fast-moving rockets. In other words, overall, if it works at all, SDI is really an offensive system. (This was brought out at American scientific conferences in 1986.)*

*In that case, George Lucas should have the last word.*

*He astutely saw space-based attack weapons coming, and gave a name to what is now a pet hope of politicians and generals:*

*Death Star.*

## THE CLUB OF ROME

One of the world's most exclusive clubs is also one of its most dismal. It is The Club of Rome, founded by Italian businessman Aurelio Peccei, having (as of 1972) some seventy members from twenty-five countries.

Their concern they call The Predicament of Mankind, or the "problematic." It is the problem of growth, pollution, population and What's Happening in general.

On funds from Volkswagen, they have sponsored studies which thinking men can only regard as the most dismal in portent of anything we've seen in years. Or ever.

Basically the prediction is that mankind has perhaps forty or fifty years left.

Not because of war, or bombs, or dirty movies, or Divine retribution, but for simple economic reasons. However, the studies are often called "computer studies," because computers are the viewing mechanism by which we have come to see these coming events.

### MALTHUS AGAIN

In the nineteenth century, a pessimistic economist named Thomas Malthus predicted that there would always be starving people, because people increased *geometrically*—expanding at compound interest, with a fixed rate of increase creating an ever-steeper growth—while agricultural production, which must feed us all, expands arithmetically, not as a rate but a few acres or improvements at a time.

This meant, Malthus thought, that there would always be the starving poor. For various reasons this did not happen in Europe. But the regrettable soundness of the general principle persists: when rates of food production can't nearly keep up with rates of population growth, people are going to starve.

This is basically the prediction.

### DYNAMIC MODELLING

Basically what has happened is this. One Jay Forrester, of MIT, has for some years been studying "dynamic models" of things, a new breed of simulation which couldn't have been done without

computers. And now dynamic models of the world's entire economic system can be created and tried out.

Basically dynamic models are mathematical complexes where things change at rates that change themselves over time. For instance, the more you eat, the fatter you get, and the fatter you get, the hungrier you are going to be. Now, just because this is simple to say in words, and *sounds* as though mathematicians would have had solved the whole class of problems centuries ago, that's not how it is. The intricacy of such models, even for just a few variables, made it impossible to foresee what happens in such complexes exact by techniques of computer enactment. Forrester, who has studied such systems since the fifties, has become alert to their problems and surprises. The culmination of his work has been a model of the entire world's economic growth, agriculture, population, industrialization and pollution; this is described in his book *World Dynamics*.

The insidious portents of Forrester's work did not go unnoticed. The dangers of population increasing at compound interest on a planet of unchanging size, and further derivatives of these changes, suggested that things might be getting worse than anybody thought. An alert Italian businessman brought together a group of scholars from all over the world to study these problems, and called the group The Club of Rome. Their first work is out now, and it is very scary and all too real. The book is called *The Limits to Growth*.

Basically what they have done is a very elaborate computer simulation, modelling the entire economy of the planet in the years to come as a structure of rates. They have taken into account population, food-growing capacity, industrial growth, pollution, and a lot of other things. The model is precise and elaborate.

Unfortunately the findings are precise and simple.

They tried all kinds of alternative futures using the model—what would happen if the birth rates were different? What if there were no pollution? What if resources were infinite?

The results of the simulation are always the same.

According to all the simulations, the human race will be wiped out—mostly or completely—by the year 2100.

Let's go briefly through the model. Note that it can't be exact, and we can't know what years things are going to happen. The curves themselves—the shape of things to come—tell the story all too clearly. (For those who would like a little more drama with their numbers, finding these matters too abstract, I strongly recommend the very beautiful Indian film "Distant Thunder," a sort of "Mr. Smith Starves to Death." Or just stick around awhile.)

## HUH?

The model assumes that birth rates stay relatively constant in particular parts of the world, and that new land and agricultural techniques increase food production in relatively well-understood ways.

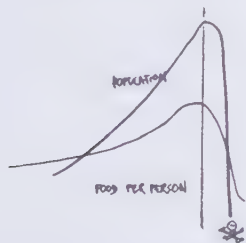
Of course, population continues to go up, on the familiar but deadly curve.



Now for the good news. Food production also tends to increase:



Now for the bad news. The running ratio of food to people, Food per Capita, takes a sudden nosedive. And then so does population.



It is not any individual prediction that is frightening, since the numbers plugged into the separate runs are merely hypotheses, to show the shape of the consequences. It is the overall set of runs that is so ghastly, because they always come out the same.

## PAY CLOSE ATTENTION

Now, it is important to clarify what is happening here and what is not. What is *not* happening: an oracular pronouncement by "the computer," showing some transcendental prediction by a superhuman intelligence. What is happening: people are trying out separate possible assumptions to see what their consequences are, enacted by the computer according to the economic rules they set up. Result: always the same. *Any* set of rules, played out in the unstable exploding-population world beyond the seventies, appears to have similarly dire results.

## WHAT HOPE IS THERE?

The original model is only an approximation, and the basic results, as published in *The Limits to Growth* (see nearby) reflect those approximations. One of the things that can be done is to fill in and expand the model more, to see whether any hopes can be found in the details and fine cracks which don't appear from the gross results. And, of course, to study and re-study the basic findings. (For instance, a small error was recently found: a decimal point was misplaced in the "pollution" calculation, leading to an overstatement of the pollution in some of the runs. (But pollution, remember, is only part of the problem.)

So there you are. This is a study of the greatest importance. We may, just may, be getting wind of things in time to change the outcome. (If only we knew how. But again, this study is where serious discussion must begin.)

Answer. With Stephen King, the scare is over when you close the book.

Why do so many people read Stephen King, and so few people read the Club of Rome?

The complex causations of the world, as modelled in the Club of Rome studies, concern multi-loop non-linear feedback systems.

## Grim Paradoxes Of Tomorrow

*"Given a stated set of assumptions, the computer traces the resulting consequences without doubt or error. This is a powerful procedure for clarifying issues."*

—Jay W. Forrester, *World Dynamics*

## GRIM DEMOGRAPHICS

The middle class tends only to reproduce itself, but the poor appear to follow Malthus' grim law, reproducing till limited by food. Perhaps the worst super-Malthusian estimate was made by Heinz von Foerster, who predicted that human population growth may be hyperbolic, going to near-vertical increase when circumstances permit—which can only last a very short time, of course.

Note that by this evaluation, no government can continue to subsidize the poor much longer, or to withhold from them any means of birth control.

In both business and government, policies often have the opposite of the intended effect. Trying to help things along may make them worse. Planning on population growth (as Forrester points out in *World Dynamics*) will cause population to grow far beyond the planned levels. Population will probably at last be limited by hunger and crowding and pollution. Thus it may be more sensible and humane to encourage hunger, crowding and pollution sooner, thus lessening the magnitude of the eventual hardship.





Perhaps the Club of Rome study should be called—  
THE HOLE EARTH CATALOG

Tom DeFanti (see p. DM 96.)

## READ IT AND WEEP

Donnella H. Meadows, Dennis L. Meadows, Jorgen Randers and William W. Behrens III, *The Limits to Growth: A Report for THE CLUB OF ROME'S Project on the Predicament of Mankind*. Universe Books, paper, \$2.75.

THE ORIGINAL ANALYSIS AND MODELS—AND EXCELLENT READING:

Jay W. Forrester, *World Dynamics*, Wright-Allen Press, 1971.

### TWO VIEWPOINTS

Aurelio Peccei, *The Chasm Ahead*, Macmillan (London), 1969.

J. Peter Vajk, *Doomsday Has Been Cancelled*.

### GOOD-FAITH NEGOTIATION SYSTEMS

More and more, the decisions of the future will be based on complex negotia-

tions involving the interplay of numerous issues—like movie deals, real estate deals, arms negotiations.

All parties will be bringing their own simulation program runs to the table.

Being able to intercompare simulations by different parties, in great detail, will thus be a vital aspect of tomorrow's negotiation systems.

I postulate that a vital new type of software will be the **good-faith negotiation system**: a thinkertoy (see p. DM 50) that allows users to make detailed intercomparisons of proposals, arguments and simulations, and draw together the agreed-upon details visually and conceptually, while working out the still-remaining issues. This demands the fullest possible development of visualizing tools for intercomparison (again, see p. DM 50).

The "good faith" part has to do with **really** showing the differences exactly, in every way people want to see them, from summary level to the finest details.

## Try It Yourself: The Club Of Rome On Your PC

Micro-DYNAMO Systems Dynamics Modeling Language. Addison-Wesley, \$200 (!) for the IBM PC.

You also need these two books:

Nancy Roberts et al.: *Introduction to Computer Simulation: A System Dynamics Modeling Approach*. Addison-Wesley.

George P. Richardson and Alexander L. Pugh III, *Introduction to System Dynamics Modeling with DYNAMO*, MIT Press.

The good news about this package is that it lets you experiment extensively with Forrester's world simulations, essentially those of the Club of Rome. Typical run: the quality of life peaks in 1950, population peaks in 2020, pollution peaks in 2050, and it's downhill from there.

This allows you to test, for example, the criticisms of J. Peter Vajk, who asserts that the Club-of-Rome models were not as robust as claimed, and that they do not converge to a simple and final end for mankind, regardless of the premises.

The bad news about this package: it is a terribly-set-up, miserable-to-use, technoid implementation of what should be a highly-interactive system. It is in the old style, with complex rules you must follow to put data into each "run."

The creators of this system have no conception of what interactivity means. The program is also badly constructed: two minutes of compiling time to produce each screen picture, and even then there is no way to scroll or change it without another compilation. There are lots of error messages (nine pages in alphabetical order) and FATAL ERRORS.

Batch mode comes to your PC! But until a better Club-of-Rome kit comes along, this will let you work over the equations.

(If DYNAMO is not yet visual or interactive enough, we may pardon Jay Forrester; for it was he that headed up Lincoln Lab in the early sixties, the days of TX-2 and Sketchpad; he is among those who brought us computer graphics in the first place.)

### THE HUNTINGTON II MODELS

Other Club-of-Rome models have been made by various groups, and perhaps some of them are available.

In particular, a world-dynamic set of models programmed by a university group in BASIC, the "Huntington II" models, has been officially distributed, but unfortunately has been seriously mishandled as far as getting it out to the public is concerned. Having it in BASIC is a definite advantage interactively over the Mini-DYNAMO system.

# ENDGAME

Now begins the winter of the world.  
We are poisoning everything.

With so little time left, we are of course expanding and accelerating every form of pollution and destruction.

We are killing the last of our beautiful brothers, the whales, just to provide marginal amortization of the whale-ships that are going to be scrapped anyway.

Item: supposedly the Sahara Desert was man-made. It is growing fast.

Set down upon this beautiful planet, a garden spot of the universe, we are turning it into a poisoned pigsty.

You and I may starve to death, dear reader. In some year fairly soon now, around the turn of the century, there will no longer be nearly enough food for the teeming billions.

That, anyway, is what the predictions say. The predictions are compelling, not because a computer made them—**anybody can make a computer predict anything**—but because the premises

from which the predictions grow were very well thought out.

It is now up to us to make the predictions come out wrong.

Not by killing the bearer of bad tidings, or by pretending they were not clearly stated—but by seeing what possible alternatives remain in the few moments of real choice we may yet have—scant years at best.

To haggle now about ideology is like arguing about who is driving while we are headed toward a brick wall with the gas pedal jammed to the floor.

The public thinks, "science will save us," a view at which many scientists snicker bitterly. Perhaps we will be shrunk to an inch's height, or fed on rocks, or given gills and super-kidneys to live in the ever-more-poisoned sea. Or perhaps we will do what science says others have done: die out.

This science-will-save-us ostrich position is nicely exemplified by Albert Rosenfeld, Science Editor of *Saturday Review/World*.

Since "science" has given us the Boeing 747 and the neutrino, neither of which could once, he thinks, have been imagined possible, obviously (to him) science

can do anything else we think is impossible! He fully imagines that science will come up with something to take care of geometrically increasing numbers of people. In perpetuity?

"Take a lesson from the neutrino," he says. "We *can* solve our problems." ("Look to the Neutrino, Thou Doom-sayer," *Saturday Review/World*, Feb 23, 1974, 47.)

## OTHER FUN

The growing diffusion of weapons and grudges, and the great vulnerability of almost everything, assure that terrorism and political extortion will increase dramatically for the foreseeable future. On the other hand, whole economic blocs and industries have lately mastered and demonstrated by example how to hold the country at bay in order to get their wishes; as everybody can see what's happening, and learn from it, the number of wrenching unpleasantnesses created by terrorists and industries and interest blocs will increase.

All these were essentially foreseen by Thomas C. Schelling in his masterly 1960 work, *The Strategy of Conflict*. Schelling

formalizes a theory of intimidation as part of his study of communicating adversaries. (His is a *structural* rather than a psychological study, examining the properties of situations whether or not they are psychologically perceived. Regrettably, perception of situations is improving all the time.)

But even if everything else were all right, the Breeder Reactors are sure to get us. I refer to those wonder machines that the electric companies are calling Clean Energy for the Future. What is not explained with such slogans is that breeder reactors not only create energy, they create atomic waste, breeding new fissionable material—including plutonium. Plutonium is well named for the god of hell. Chemically a poison and radioactively a horror, it does not go away; wherever we put it, it will get back to us.

The mere radiation from the atomic crap is hardly the problem. The radioactive poisons are getting into the oceans. They are getting into the clean waters of the land. (A December 1973 news report, for instance, revealed that a 1968 leak of radioactive chemicals was into the water



*The "environment," meaning mainly air, water, forests and non-human creatures, used to be big, used to surround us; human society was a spotty occupation force in a surrounding immensity of growth and animal life. (It is said that in the seventeenth century, a squirrel might hop from branch to branch all the way from Paris to Moscow.)*

*Now human infestation and effects are so great that the "environment," in these senses, is now small and fragile, and it is human settlement and causations that are the surrounding immensity. All other creatures now live at our sufferance. And so nature must now be managed, with all the political, policy and cost problems this implies.*

ALL THE  
WORLD'S  
A STAGE—  
and we'll  
get over it.

—Anonymous

Cousteau says the  
oceans are dying a lot  
faster than he anticipated—  
and gives mankind fifty  
years after life ends there.

## IBM Is Bullish On The Future



Lewis M. Branscomb, who has the awesome title of Chief Scientist of IBM, has been giving numerous talks recently that seem to be directed against pessimism resulting from the Club of Rome studies.

"On the shoulders of the information processing community rests the responsibility for convincing the public that we have the tools, if it has the will, to address the complex systems management problems of the future," Branscomb said.

"More than any other profession our community can restore the public's confidence that from the limited resources of the world can be fashioned a life of well-managed abundance for all," he concluded.

(Keynote speech, ACM '73, quoted in *Computerworld*, 5 Sept '73, p. 4.)



*Pretty soon they'll want to call Pollution—*  
ENVIRONMENTAL ENHANCEMENT

*The very notion of "cleaning up" toxic waste is bairy. Toxic waste, basically, is what can't be cleaned up, will not decay, and will do harm if it gets into the water, the air, our mouths or our skies. (Public native is so great. I know one woman who thought Bon Ami cleanser would be the answer.)*

*It's like that old joke about how to recan a spilled can of worms: "Get a bigger can." Where are we going to find a bigger can for all those drums and holding tanks, all those leaking pipes?*

*The cost of keeping this junk out of the water and our bodies is escalating like the national debt. And the great danger is that future "conservative" government will conveniently lose all the records of where these things are. That will solve the problem fast—but after that, where will it be safe to live?*

.....  
EVERYTHING  
IS DEEPLY  
INTERTWINGLED®

supply of Bloomfield, Colorado.) Now, atomic enthusiasts call it a Disposal Problem, like the question of where to bury the garbage. But it's a very different problem. Wherever we put it, it will come back. The sea? No, that'll be poisoned after the containers go. Deep wells? The mountains? But there is no place that cannot be guaranteed against earthquake and recycling. It will come back. Though dozens of generations might survive it, it will be waiting.

But the breeder reactors multiply this output. Perhaps we could survive the waste for a few hundred years, till it comes back out. But the other part of it is the fissionable material which can be made into backyard bombs.

That's the kicker. With more and more fissionable crud being disgorged, its availability for terrorists who want to build their own increases. Ralph Lapp pointed out last year that the stuff was

shipped in unguarded trucks, and one or two good hijackings would enable any bright kid to build his own dirty A-bomb. By the year 2000 it is not inconceivable that bootleg atomic weapons will be as widespread as handguns in Detroit—and as much used.

But now, with the breeder reactors—in lots of countries—pouring the stuff out, the era of atomic plenty is here. The smaller countries who want them are getting their atomic weapons—though holding back assembly of the parts, for various reasons. It is generally believed among bomb-watchers, for instance, that India and Israel have theirs anytime they want.

Add this to the great avalanche of missiles, tall and horny in their silos, ready to go on two, later three or four, sides. (The U.S. official arsenal now stands at the explosive equivalent of 5 billion tons of TNT, a ton of TNT for every human being. And that's just the explosive part,

*It is a safe prediction that the horrors of the future will be unprecedented, and will make the horrors of the twentieth century look modest.*

*What with the accelerating depredation and destructions of the earth's surface, groundwater and arable land, the snuffing of the wildlife and the great forests, it is fairly certain that starving Malthusian mobs will be vying for a torn-down, entangled, Bronxified and poisoned planet.*

*Whatever chance remains for the survival of anything good may be in the preservation and availability of information, the only commodity that will be cheaper and more convenient. See p. DM 141.*

Civilization, and the bulk of mankind, have about forty years to live, according to certain studies (see p. 172). The studies are depressingly good, although unfinished.

There are five possible things to do.

1. Ignore it.
2. Deny it.
3. Seek individual salvation somehow. Hide in a remote corner. Lay in stores.
4. The glorious flameout. Eat, drink and be merry, for tomorrow we die. Or apocalyptic occultism, or whatever.
5. Work starting now. In whatever directions might, just might, point or contribute to a way out.

not the fallout; a fraction of these bombs could destroy all life on earth by its seething residue.) And now, because of the SALT talks, we may expect a new and drastic increase of this Readiness Posture. Hoo boy. What is there to say.

So there it is, folks, merry times ahead. Humanity may end with a bang (thermonuclear exchanges, or just desultory firings until we're all poisoned or sterile), or a whimper (universal starvation), or, I would anticipate, some spastic combination of the two, and all within the (possible) lifetime of the average reader. This is, at any rate, what I think most likely.

Except of course we won't see it happen that way. We'll watch the starvations on TV (as we did Biafra, Bangladesh, now West Africa, what next... India?), and tsk about the poor foreigners who can't take care of themselves. And as the problems increase and move toward our heartland, it'll be blamed on environmentalists and on the news media, till bang.

Or maybe not. Just maybe.

But we've all got to get access to the Club of Rome models, and look for holes or strategies. If computer modelling systems doing this kind of work are made widely enough available, perhaps some precocious grade-schooler or owlish hobbyist will find some way out that the others haven't hit on...

We've got to think hard about everything.

*There's not much to add to this. We hear about Progress, yet there is everywhere an ever-growing pool of the hungry and dispossessed. There are now, in 1987, five billion people on the planet. The starvation continues (currently mostly in Africa as the Sahara grows). The principal imperative of the modern world seems to be: More Weapons. Where will you be when the music stops?*

## BIBLIOGRAPHY

- Frederick Pohl and C.M. Kornbluth, *The Space Merchants*. Ballantine, paper.  
 Thomas C. Schelling, *The Strategy of Conflict*. Paper.  
*The Great American Bomb Machine* (citation not handy). Paperback.  
 A book called *Cold Dawn* (citation not handy; originally published in the *New Yorker*) presents a most discouraging view of this country's actions in the SALT talks.

One Access Catalog, not to be named here, gives a recipe for an atomic bomb. Very funny, ha ha. "The U-235 we are using, (although Plutonium will work just as well) is a radioactive substance and deserves some care in handling. It is NOT radioactive enough to kill with limited exposure, but don't sleep with it or anything." And so on. Thanks a lot, fellas. Ralph Lapp had a piece in the *New York Times Magazine* last year, pointing out that plutonium is shipped in unguarded trucks and it's only a matter of time before punks get their hands on it...

A piece in a recent *Esquire*, "Did There Ever Come a Point in Time When There Were Forty-Three Different Theories about Watergate? Yes, to the Best of Our Recollection," is a very helpful general source, especially for those who suspect a connection between "Watergate" and the assassinations of the Kennedys, Malcolm X, Martin Luther King, etc. But for a real chill, see "Mae Brussell's Conspiracy Newsletter" in the March (?) 1974 *Realist*, as well as "Who Is Organized Crime and Why Are They Saying Such Awful Things About It," same issue.

Glen A. Love and Rhoda M. Love, *Ecological Crisis: Readings for Survival*. Harcourt, \$4 (paper). A quick way to catch up on some bad stuff. Four bucks well spent.

William Leiss, *The Domination of Nature*. Braziller, \$7.

For a dazzling, romantic and optimistic view of the future, see *Dimensions of Change* by Don Fabun (Glencoe Press, \$5 in paper).

*The Futurist* magazine goes out to members of the World Future Society, An Association for The Study of Alternative Futures, Post Office Box 30369, Bethesda Branch, Washington, DC 20014. The magazine used to be pretty sappy and optimistic, but seems to be acquiring sophistication. The Bulletin of the Atomic Scientists, a slick magazine, is actually very readable. It presents the current perils of the world as they appear to scientists.

Samuel Epstein, *Hazardous Waste in America*. Epstein is a very bright guy and serious in his warnings.

Hazardous Waste Investigation. A manual to help you find out what's going on in places that concern you.

## WHAT NEXT?

By browsing this book you may have more sense of what computers are doing, can do, should do.

What will you do now?

How do you think computers can help the world?

What are you waiting for?

Remember:

COMPUTER POWER TO THE PEOPLE!

DOWN WITH CYBERCRUD!

"Things  
are  
going to  
get  
worse  
and  
worse  
and  
never get  
any  
better  
again."

attributed to  
Kurt Vonnegut,  
Jr.

**"FOLKS DON'T NEED THESE LI'L SHMOOS!!—THEY ALREADY GOT ONE—TH' BIGGEST SHMOO OF ALL—TH' EARTH, ITSELF! JEST LIKE THESE LI'L SHMOOS, IT'S READY T'GIVE EV'RYBODY EV'RYTHING THEY NEED!! EF ONLY FOLKS STOPPED A-FIGHTIN', AN' A-GRAB-BIN'—THEY'D REE-LIZE THET THIS SHMOO—TH' EARTH—GOT PLENTY O'EVERYTHING—FO' EV'RYBODY!!"**

—Li'l Abner

(Al Capp, *The Life and Times of The Shmoo*, Pocket Books, 1949, pp. 121-122.)

## Beyond Ideology(2)

*Yesterday's ideological labels, like Communism and Capitalism, have little to do with the problems of tomorrow. We have an overwhelming and overriding commonality of interest now: peace, survival, preservation. The survival of humankind, and this our planet, is now the central issue.*

*Saving the earth's oxygen, its forests and wildlife, clean water we can continue to drink; coping with the proliferation of starving people; AIDS; beginning to reduce the nuclear arsenal, and cope with our new oceans of nuclear and chemical waste—these are the issues of tomorrow, issues to which the slogans of yesterday are totally irrelevant. Global solutions will take systems and negotiated programs on a scale never before seen.*

*New system ideologies will spring up, and perhaps become the fabric of a new politics; but that is a corner we cannot yet see around.*



# PIVOT

## DEDICATION

The first edition was dedicated to my grandfather. This second edition is dedicated to my son. With my apology that these toys were not ready when he was a boy, this book is dedicated

to: Erik  
Nelson  
to: Erik

Calligraphy by Scott Kim  
April 16, 1985

me from teaching duties, and everybody for their encouragement.  
Thanks to Commander Hugo McCauley, better known to you as Hugo's Book Service, for his yeoman performance in shipping out the books.  
Thanks also to my good friend Robert W. Fiddler, Esq., patent attorney and still an ex-philosophy professor at heart, for many delightful and witty conversations on the vagaries of intellectual property.  
I would like to thank everyone who furnished materials and permissions for the things herein.  
I am grateful to the many who have explained computers to me over the years; John R. Levine, Tom Barnard, Tom DeFanti.  
My thanks to the many others whose goodwill has kept me going, in particular my former wife and eternal friend, Deborah Stone Nelson.  
And finally thanks for his encouragement over the years, and hope that these tools will soon be useful

Thanks also to Bob Wallace for all his help with PC-Write and to Scott Kim for the calligraphy below.  
Thanks for great personal inspiration to Catherine Ikam and Timothy Leary.  
**ACKNOWLEDGEMENTS (From the First Edition)**  
This project could never have been completed without the efforts of my wise and warm friends Sheila McKenzie and Wade Freeman, who have my deepest gratitude.  
At Chicago Circle Campus, I am grateful not only for the encouragement and assistance of various individuals (especially Joseph I. Lipson, David C. Miller and Samuel Schrage), but for the atmosphere of support which has made this possible. My thanks to the Department of Art and the Office of Instructional Resources Development for freeing

phases of the project, and for keeping my computers alive.  
Especially thanks to Steve Ditlea for his great exertions at many places and times that made this book finally happen.  
Thanks also to the nice folks at Microsoft Press, especially Ron Lamb, Darcie Furlan, Becky Johnson, Jean Trenary, Tracy Smith, Anne Depue, Susan Lam-mers, Claudette Moore and Linda Pennington, for their encouragement and glad cooperation.  
My especial gratitude to Charles and Judith Harris for their encouragement over the years.

**THANKS (Second Edition)**  
My foremost and most special gratitude to wonderful Lauren Sarno, without whose extraordinary and punctilious endeavors this edition could not have been successfully prepared.  
Thanks to Sean Harmon for her hospitality and encouragement during these travails.  
Special thanks to the brilliant and mysterious Steve Wilham for help in various

# PIVOT

The occasional Oz illustrations are all by John R. Neill, from various out-of-copyright Oz books by L. Frank Baum, especially Ozma of Oz and Tik-Tok of Oz. Tik-Tok, the Machine Man, is the figure to whom occasional allegorical significance is attached here by juxtaposition.

people who knew he was in Auschwitz were frequently shocked by the way that Jack talked about being in the camps. "You know," he once told me, "it's hard to believe it really happened. But it can happen again. In America. Americans like to make rules, and that scares me. If you have too many rules you get locked in a system. It's the system that says this one dies and that one doesn't, not the people. That's why I don't hate the German people. Individuals, yes. Rules, yes. But not all Germans." He shrugged. "They just obeyed the rules. But that's why we need more Commodores. We need more mavericks, just so the rules don't take over."

Jack Tramtel, Auschwitz survivor, founder of Commodore, computer mogul, quoted in Michael S. Tomczyk, The Home Computer Wars: An Insider's Account of Commodore and Jack Tramtel. Computer Publications, 1984, p. 55-6.







# DREAM MACHINES

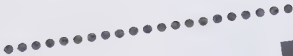
8



New Freedoms Through Computer Screens  
— a Minority Report

This is the flip side of Computer Lib.





Cover design adapted by Microsoft Press from original edition  
Interior design and art direction by Becky Geisler-Johnson  
Principal typography by Jean Trenary  
Additional production support by Michelle Lubow, Chris  
Larralde, and Bryce Holmes  
Artwork from original edition by Ted Nelson  
Text composition by Microsoft Press using the Magna composi-  
tion system and the Mergenthaler Linotron 202 digital  
phototypesetter.

# INTRODUCTION TO THE 1987 EDITION

*This is a hard book to summarize. Like Computer Lib, it is a lot of pieces presented from a point of view. I will try to explain the point of view here at the beginning, but browsing through this book—I do not particularly recommend trying to read it consecutively—will make it all much clearer.*

*This is a collection of materials and thoughts about presentational systems and media, now and in the future.*

*We live in media as fish live in water, seeing and dealing with them but not recognizing their effects on our lives and our thinking. Writing-paper, books, magazines, still photography, newspapers, lectures, live theater, schools and universities were essentially our media as the twentieth century began; but then came mass-marketed audio recordings, movies, the telephone and radio. Those shaped a different world, the world of the twenties, the thirties and World War Two.*

*Then came television in the late forties, which by being vivid, passive and narcotically habit-forming, shaped the fifties, sixties and seventies. Television also brought a decline in literacy, in vocabulary, in mental awareness, in the ability to read maps and diagrams; indeed, as McLuhan uncannily foresaw, the nation has re-entered a sort of "tribal" state (as he called it) in which the mind can no longer grasp specifics, rejects them like transplanted organs, and deals*

*with the world instead on the basis of finely-tuned intuitions—whose vocabulary is brilliantly played on by commercials and the so-called news.*

*Those of us who came from a generation of words and literacy think this is very bad, and hope the tide of dementalization can be reversed.*

*Now we are at a new time, and hundreds of new media are being impatiently prepared by lunatic dreamers and corporate development teams everywhere. I am*

## Author's Counterculture Credentials



*Designer of interactive software for personal computers since 1960. Independently invented word processing, outline processing 1960 (but after Engelbart). Coined terms hypertext, hypermedia.*

*Writer, showman, generalist, Gemini, moon in Libra, Gemini rising.*

*Onetime seventh-grade dropout. I have relatively little interest in improving the educational system within the existing framework.*

*Author of what may have been world's first rock musical, "Anything & Everything," Swarthmore College, November 1957 (with Richard L. Caplan).*

*Photographer for a year at Dr. Lilly's dolphin lab (Communication*

*Research Institute, Miami, Florida). Attendee of the Great Woodstock Festival (like many others), and it changed my life (as others have reported). What we are all looking for is not where we thought it was.*

*Lifelong media nut. Magazine collector; hung around TV studios as a child. Compulsive explainer. Gimmickist by disposition, computerman by accident.*

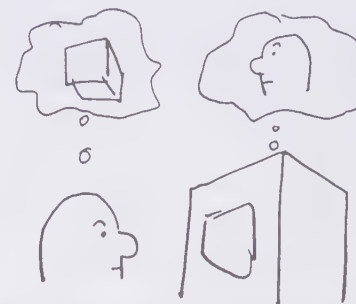
*Officer of the Future; Seeker After Truth; Renaissance Wacko.*

*This book (both sides) is based in part on my talks at or before the American Chemical Society, the American Documentation Institute, the American Management Association, the Associated Press, the Association for Computing Machinery, the Central Intelligence Agency, the Institute of Electrical and Electronics Engineers, the Printing and Publishing Association, the Rand Corporation, the Society for Information Display, the Society of Motion Picture and Television Engineers, TIME Incorporated, Union Theological Seminary (the Auburn lectures), Xerox Palo Alto Research Center, and various art schools, colleges, universities and Joint Computer Conferences.*

THE  
GESTALT,  
DEAR  
BRUTUS, IS  
NOT IN OUR  
STARS BUT  
IN  
OURSELVES.

I believe that an introduction to any subject can be humorous, occasionally profound, exciting, vivid, and appealing even to experts on their separate levels.

Perhaps someday I can prove it.



**NEW ARTS, NEW UNDERSTANDING: A Confusing but Titillating Summary of Dream Machines**

*Media are all the same but different. The principles of presentation are the same in all media; writing is like moviemaking is like drawing a diagram is like designing appliances is like designing video games is like designing media: a unified art and science, visualization and conceptual artistry, the communication of ideas with feeling. The art and film schools haven't caught on yet.*

*Chaos abounds. We must make things clearer and more open, not more mysterious and magical-seeming. STANDARDIZATION IS VITAL AND IT MATTERS WHAT WE CHOOSE. We stand at a crossroad between magnificence and crap. Others say complication is necessary. I say simplicity is necessary. The technicalities matter immensely.*

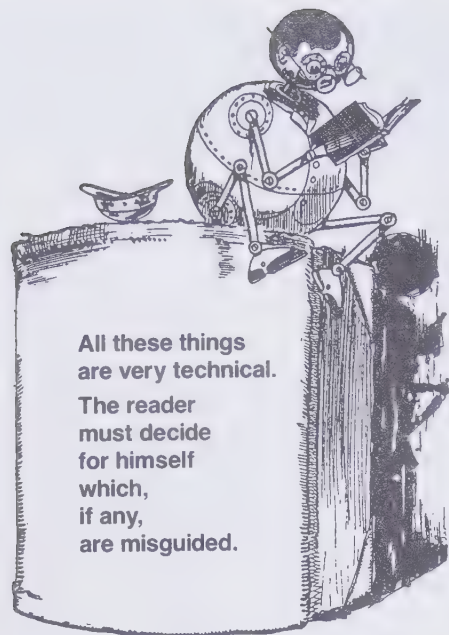
## HOW TO READ THIS BOOK.

*Paragraphs in italics are new; paragraphs in regular type are from the 1974 and 1975 editions; headlines vary. Page references without the "DM" are to the other side.*

No more  
pencils,  
no more  
books;  
No more  
teachers'  
dirty  
looks.

— Kid's jingle  
with new  
meaning.





But the point here is not to hassle the dreamers, just to sort out the dreams and put them on hangers so you can try them on, and maybe choose an ensemble for yourself.

**COME DREAM ALONG WITH ME:**  
THE BEST IS YET TO BE.

► **"S**ome of us started out in computers drawn by a vision of our particular art or society in general, and we realized that computers could be not only useful, but revolutionary, in exploring the visions we were compelled to pursue."

—Laurie Spiegel

► **"A**ll the wonders that we see at this conference are brought into the world by people listening to their imaginations."

—Loren Carpenter SIGGRAPH '85

► **"I**t's not the technology that lives. It's the dream that lives."

—Alan Kay

using the term "media" in the broadest sense, to include all the things in this book; each of which has a claim, or a claimant, for being a viable part of our new world that is to come.

Enthusiasts, in their minds embracing it all as if it were one, seem to think the new media will, with no involvement by the public, close over us like a convertible top, to make a beautiful new unified canopy of vivid and accessible information.

This is not so. The exact details of the new media matter enormously, down to the smallest iota; and we are indeed in danger that very bad ones will be adopted, enveloping us irrevocably in systems that will cripple our thinking and our access to information in even more ways than television has, cutting us off even further from analysis, from fact, from access to the past, from clarity and from intercomparison.

Yet there remains the very real hope of that unified canopy of vivid and accessible information, an environment which will enhance and nourish our minds and their capabilities, taking us not only to former levels of literacy but far beyond, to new levels of understanding and intelligence.

This is a difficult message to propound, because many may feel threatened by the idea that the next generation will be smarter—the same anticipatory resentment that often leads parents unconsciously to hold their children back.

But tomorrow needs all the human intelligence that can be created, through every possible mechanism and channel.

It is with that hope that this book and its ideas are offered up to you.

## BIBLIOGRAPHY

Marshall McLuhan, *Understanding Media* (various editions).

## INTRODUCTION TO THE 1974 EDITION

(Feel free to begin here. The other side is just if you want to know more about computers, which are changeable devices for twiddling symbols. Otherwise, skip it.)

(But if you change your mind, it might be fun to browse.)

In a sense, the other side has been a come-on for this side. But it's an honest come-on: I figure the more you know, the readier you'll be for what I'm saying here.

Not necessarily to agree or be "sold," but to think about it in the non-simple terms that are going to be necessary.

The material here has been chosen largely for its exhilarating and inspirational character. No matter what your background or technical knowledge, you'll be able to understand some of this, and not be able to understand some of the rest. That's partly from the hasty preparation of this book, and partly from the variety of interests I'm trying to comprise here. I want to present various dreams and their resulting dream machines, all legitimate.

If the computer is a projective system, or Rorschach inkblot, as alleged on the other side, the *real* projective systems—the ones with projectors in them—are all the more so. The things people try to do with movies, TV, and the more glamorous uses of the computer, whereby it makes pictures on screens—are strange inversions and foldovers of the rest of the mind and heart. That's the peculiar origami of the self.

Very well. This book—this side, *Dream Machines*—is meant to let you see the choice of dreams. Noting that every company and university seems to insist that *its* system is the wave of the future, I think it is more important than ever to have the alternatives spread out clearly.

But the "experts" are not going to be much help; they are part of the problem. On both sides, the academic and the industrial, they are being painfully pontifical and bombastic in the jarring new jargons. Little clarity is spread by this. Few things are funnier than the pretensions of those who profess to dignity, sobriety and professionalism of their expert predictions—especially when they, too, are pouring out their own personal views under the guise of technicality. Most people don't dream of what's going to hit the fan. And the computer and electronics people are like generals preparing for the last war.

Frankly, I think it's an outrage making it look as if there's any kind of scientific basis to these things; there is an under-level of technicality, but like the foundation of a cathedral, it services only to support what rises from it. **THE TECHNICALITIES MATTER A LOT, BUT THE UNIFYING VISION MATTERS MORE.**

This book has several simultaneous intentions: to orient the beginner in fields more complex and tied together than almost anybody realizes; nevertheless, to partially debunk several realms of expertise which I think deserve slightly less attention than they get; and to chart the *right* way, which I think uniquely continues the Western traditions of literature, scholarship, and freedom. In this respect, the book is much more old-fashioned than it may seem at the gee-whiz, very-now level.

The main ideas of this book I present not as my own, but as a curious specis of revealed truth. It has all been obvious to me for some time, and I believe it should be obvious as well to anyone who has not been blinded by education. If you understand the problems of creative thinking and organizing ideas, if you have seen the bad things school so often does to people, if you understand the sociology of the intellectual world, and have ever loved a machine, then this book says nothing you do not know already.

For every dream, many details and intricacies have to be whittled and interlocked. Their joint ramifications must be deeply understood by the person who is trying to create whatever-it-is. Each confabulation of possibilities turns out to have the most intricate and exactly detailed results. (This is why I am so irritated by those who think "electronic media" are all alike.)

And each possible combination you choose has different precise structures implicit in it, arrangements and units which flow from these ramified details. *Implicit* in Radio lurk the Time Slot and the Program. But many of these possibilities remain unnoticed or unseen, for a variety of social or economic reasons.

Why does it matter?

It matters because we live in media, as fish live in water. (Many people are prisoners of the media, many are manipulators, and many want to use them to communicate artistic visions.)

But today, at this moment, we can and must design the media, design the molecules of our new water, and I believe the details of this design matter very deeply. They will be with us for a very long time, perhaps as long as man has left; perhaps if they are as good as they can be, man may even buy more time—or the open-ended future most suppose remains. (See "Endgame," p. 175)

So, in these pages, I hope to orient you somewhat to various of the proposed dreams. This is meant also to record the efforts of a few Brewster McClouds, each tinkering toward some new flight of fancy in his own sensorium.

But bear in mind that hard-edged fantasy is the corner of tomorrow. The great American dream often becomes the great American novelty. After which it's a choice of style, size, and financing plan.

The most exciting things here are those that involve computers: notably, because computers will be embraced in every presentational medium and thoughtful medium very soon.

That's why this side is wedded to the other: if you *want* to understand computers, you can take the first step by turning the book over. I figure that the more you know about computers—especially about minicomputers and the way on-line systems can respond to our slightest acts—the better your imagination can flow between the technicalities, can slide the parts together, can discern the shapes of what *you* would have these things do. The computer is not a limitless partner, but it is deeply versatile; to work with it, we must understand what it can do, the options, and the costs.

My special concern, all too tightly framed here, is the use of computers to help people write, think, and show. But I think presentation by computer is a branch of show biz and writing, not of psychology, engineering, or pedagogy. This would be idle disputation if it did not have far-reaching consequences for the designs of the systems we are all going to have to live with. At worst, I fear these may lock us in; at best, I hope they can further the individualistic traditions of literature, film, and scholarship. But we must create our brave new worlds with art, zest, intelligence, and the highest possible ideals.

I have not mentioned the emotions. Movies and books, music, and even architecture have for all of us been part of important emotional moments. The same is going to happen with the new media. To work at a highly responsive computer display screen, for instance, can be deeply exciting, like flying an airplane through a canyon, or talking to somebody brilliant. This is as it should be. ("The reason is, and by rights ought to be, slave to the emotions." —Bertrand Russell.)

In the design of our future media and systems, we should not shrink from this emotional aspect as a legitimate part of our fantic (see p. DM 74) design. The substratum of technicalities and the mind-bending, gut-slammng effects they produce, are two sides of the same coin, and to understand the one is not necessarily to be alienated from the other.

Thus it is for the Wholiness of the human spirit that we must design.

## THE GREAT AMERICAN DREAMER

I already said on the other side that the computer is a Rorschach, and you make of it some wild reflection of what you are yourself. There is more to it than that.

America is the land where the machine is an intimate part of our fantasy life.

Germans are too literal, they can get off on well-oiled cogs. The French are too vague. (I've noticed that German science-fiction magazines had covers of machines and planets; French science-fiction magazines, of dragons and people with wings. Our science-fiction covers show people *with* machines. Intimately, emotionally.) German fantasy is icy and impersonal, French fantasy too personal, and American fantasy is splat in the middle, uniting both: man and machine, means and ends, emotion and details.

Men always longed to fly, but it was here that they first *did*. This is the land of the MOVIE, a fantasy fabricated with endless difficulty using various kinds of equipment.

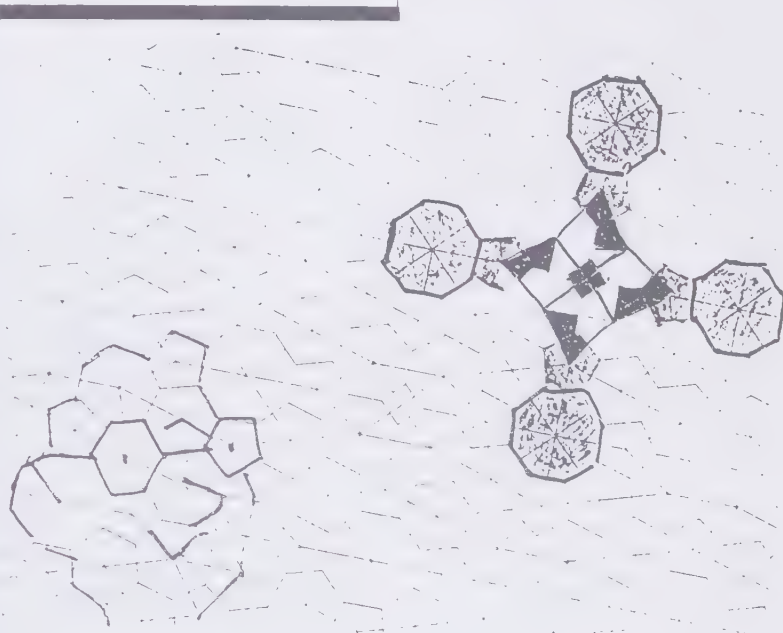
The mad tinkerer is a fabled character in our fiction.

This is a land of the kandy kolor hot rod, the Hell's Angel chopper, the drive-in movie. And the wild hot-rod, in fact, is just the flip side of the deep-carpeted Cadillac: *each* is a fantasy, an extension of its owner's image of himself in the world.

## DREAMS

"Technology is an expression of man's dreams. If man did not indulge his fantasies, his thoughts alone would inhibit the development of technology itself. Ancient visionaries spoke of distant times and places, where men flew around and about, and some could see each other at great distance. The technological realities of today are already obsolete and the future of technology is bound only by the limits of our dreams. Modern communications media, and in particular electronic media, are outgrowths and extensions of those senses which have become dominant in our social development."

—How Wachspress, "Hyper-Reality." © Auditac Ltd. 1973





☛ Thus it was not an historical accident, but utterly predetermined, that in the hands of Americans the computer would become a way of realizing every conceivable wild fantasy that was dear to them.

This is perfectly all right. This is as it should be. This is the best part of our culture. Not "Let a hundred flowers bloom," but "Let a hundred gizmos clank." This has sped immeasurably the imaginative development of many different things we might want. I try here fairly to explain a few differences among them.

There is just one problem with all this. Now that all these things exist, or come nearer to existing, which ones will other people want? What will it be possible for everyone to have? And how can we tie all these things together?

(Note: this thesis is being advanced only half-seriously. There have been a number of exactly-dreamful Frenchmen, and for this three-nationality split to be really true, they would all have to have come from Alsace, next to Germany: Jules Verne, Daguerre, the brothers Montgolfier, the brothers Lumiere, to name a few.)

*Computers are all essentially the same; yet they have inspired people to build artificial worlds, cloud-castles around them, each of which is its own creator's notion of "the computer" and "the way a computer should act." Unfortunately those who enter these castles believe in the substantiality of their walls, and do not see the hands of their architects: and as they sit entrapped in the dungeons of these castles, BLAME THEMSELVES, instead of saying, "HOW DARE THEY!"*

*While all computer systems represent the dreams of their makers, this flip side of Computer Lib is devoted to the dreams we all can share, dreams for a better world of the mind, the imagination and the understanding. Without much more understanding, humanity will not last long.*

*All the new media will involve computers in some way. But it is not the computer's involvement that matters in itself: computer setups can be hideously entangling and insidiously crippling, or they can be wonderfully liberating. What matters is the design and workability of the media we create and choose. This is an*

*enormous responsibility and an excitement.*

But folks, this all is the merest beginning. As it says on diametrically the other side, pp. 4,6,

COMPUTERS BELONG TO ALL  
MANKIND.

## APPARATUSES OF APPARITION

It seems different companies are all the time introducing wonderful new devices that will revolutionize, uh, whatever it is we do with, uh, information and stuff. Things you'll attach to your TV to get highbrow programs or dirty movies. Microfilm devices that will shrink the contents of the Vatican Library to a dot on your glasses. Goggles that show you holographic color movies. A pince-nez that lets you see the future. And so on.

Reading *Popular Mechanics* or the Saturday review of patents in the *New York Times*, you get the idea of Something Big, New and Wonderful About to Happen, so we'll all have access to anything, anytime, anywhere.

But it's been that way for decades, and with certain exceptions hasn't happened yet.

Here are some things that have caught on, and are mostly familiar to us all.

Book. Newspaper. Magazine. Radio (AM). Phonograph record (78). Tape recorder (1/4"). Black-and-white television. Radio (FM). Phonograph record (33).

Phonograph record (45). Color television. Tape cartridge (1/4"). Tape cassette (Philips, ca. 1/8"). Stereo records and tapes. Oh yeah, and movies: 35mm, 16mm, 8mm, Super 8mm. Carousel projectors. Viewmaster stereo viewers.

Here are some things in the process of catching on (and not assured of success): Quadrophonic sound. Dolby. Chromium dioxide tape emulsion. Super 16 movie format.

But for everything that *did* catch on, dozens didn't. Some examples: 12-inch 45 rpm records. 11.5 millimeter movies. RCA's 1/4-inch tape cartridge, which became a model for the much smaller Philips. Wire recorders.

Then there are the things that caught on for awhile and went away. Stereopticons (and their beautiful descendant, the Tru-View, which I loved as a kid). Cylindrical recordings. Piano rolls. And so on.

Then there are the video recording systems. CBS' EVR died before it got anywhere. RCA's SelectaVision isn't out yet. 2-inch quad is standard in the studios, 1/2-inch Porta-Pak is standard among the Video Freaks, and it looks like Sony's 3/4" cartridge will win as the main sales and storage medium. (The Philips system here looks as though it won't make it, and 1-inch is dubious.) But what's this we hear about video disks (twenty-five years after they announced Phonovision. Ah, well.)? ☛

## Great American Machine-Dreamers

D.W. GRIFFITH—took the movie-box and created the *photoplay*, no longer a twisted stage production.

WALT DISNEY—created a hypnotic pantheon of kindly and innocent semi-animals, sentimentally universal, generally acceptable.

JOHN W. CAMPBELL—as author and then editor of *Astounding*, turned American science-fiction from the Buck Rogers space opera to the *human story*, built around thought-out premises and structures.

### DREAMERS OF COMPUTER MEDIA: THE TITANS

IVAN SUTHERLAND—programmed and systematized a computer setup for helping people think and work with deeply-structured pictorial information. (See p. DM 96)

JOHN WHITNEY—first created movies by computer and the "computer movie" as art form; invented the motion-control camera and slit-scanner, foundations of today's science-fiction movies. But his unified art of digital harmony of music and animation is still AN UNFINISHED DREAM.

DOUG ENGELBART—foresaw the use of computer screens as a way of expanding the mind, and over the last decade and a half has brought about just that.

"Word processing" has taken over the world. But his is still AN UNFINISHED DREAM.

ALAN KAY—creator of today's graphical-computer paradigm, pioneer of screen icons and object-oriented computer languages; but his *Dynabook*, the easy supercomputer for children, is still AN UNFINISHED DREAM.

## Are We Materialistic?

Persons of sagacity have been saying for some time that we are materialistic.

In an important sense this is not so.

The machines, and toys, and involvements we buy into, are in but a small proportion of cases owned simply as scores, for their cost as consumption symbols.

Rather, we buy things that REPRESENT IDEALS, hoping ourselves to partake of some abstraction or image—the *Playboy* man, the Smart Businessman, the Clever Homemaker.

Each product tries to tell us it is the keystone of a way of life and then, at least at that moment of purchase, we step into, we embrace that way of life, covering ourselves with the feeling, the aura, the magic we saw in the commercial.

This is not materialism. It is wishful grasping at miasma. It's a seeking, not to possess, but to belong.

## Wonderful Systems That Were Gonna Be *Where Are The Shows Of Yesteryear?*

I once read a mind-blowing review article in *Films in Review*, early sixties I think, on schemes to make three-dimensional movies *before 1930*. There were dozens.

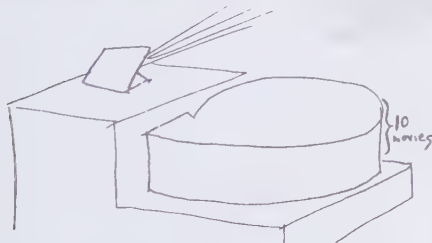
Then there was the multiscreen film *Napoleon*—a legend—done in the nineteen-twenties. (That one really existed.)

*Phonovision*, about 1947 or so, was going to store a half-hour movie on a 12-inch disk. Did they get the idea from the LP? Did they really think they could do it?

The German photo-gizmo, around 1950: a special camera that supposedly created a *sculpture* of what it was pointed at. (But how did it know what was behind things?)

A weird lens around 1950—I think it was depicted as having a blue center and a red periphery, like a fifties hoodlum tail-light—that was somehow going to find “residual traces” of color in black-and-white pictures, and make ‘em into color, zowie, just by copying them.

Then there was the Panacolor Cartridge. During the Days of Madness—1968, I think it was—a rather good little movie gadget was being pushed by a firm called Panacolor. It had ten parallel movie and audio tracks, I believe, on a 70mm strip. The prototypes were built by Zeiss.



Their idea was that this was a compact movie projector. I kept trying to persuade the company's president that they had inadvertently designed a splendid device for branching movies (see “Hyperfilms,” p. DM 64).

Exercise for the reader: map out properties of the branching and expository structures implicit in such a device. (It's one-directional. Gotta rewind when you get to the end. But you can jump between tracks when it seems appropriate.)

Anyway, it's gone now.

## Pay Careful Attention

when somebody shows you an electronic or other presentational system, device or whatever.

A certain kind of sleight-of-hand goes on. It's very easy to get fooled. They may show you one thing and persuade you you've seen another.

And if you're canny enough to ask about a feature you haven't seen they'll always say, “WE'RE WORKING ON IT.”

It's only dishonest if they say, “It'll be ready next month.”

“The  
Emperor  
has no  
clothes on!”  
—Small Boy  
(Name  
withheld)

## ● Children Of All Ages!

Ladies and gentlemen, the age of prestidigitative presentation and publishing is about to begin. Palpitating presentations, screen-scribbled, will dance to your desire, making manifest the many mysteries of winding wisdom. But if we are to rehumanize an increasingly brutal and disagreeable world, we must step up our efforts. And we must hurry. Hurry. Step right up.

Theodor H. Nelson, “Barnum-Tronics.” *Swarthmore College Alumni Bulletin*, Dec. 1970, p. 13-15.



## The Balkanization And Hope Of Media

Paper media, whatever their disadvantages, have at least been compatible: you could store the books, magazines and notes on the same shelf, compare them on the same desktop.

Not for the new media, whether electronic or optical or magnetic or computerized. Each one needs a separate device. You cannot presently make marginal notes on a videotape.

I say it will all have to come together again. We need a presentational and archival medium that can be as standard as paper, to reunify the present mess of separately beautiful and mutually unintelligible forms of storage, presentation and annotation.

The hope may be a shared-standard data structure (see p. 141).



The thing is, so many of these things seem to sound alike. They all mention "information retrieval," education, technology, possibly "the information explosion" and "the knowledge industry." Press releases or effusive newspaper articles may use phrases like "space-age," "futuristic," "McLuhanesque" or even "Orwellian" (though few people who use that word seem to know what Orwell stood for; see p. DM 151).

And the intimidating company names! Outfits with names like General

Learning, Inc., or Synergistic Cybernetics, Inc., or even Communications Research Machines, Inc. Surely such people must know what they are doing, to use such scientific-sounding phrases as these!

Then there are the business magazines. In the late sixties they were talking about "The Knowledge Industry" (a fiction, it turned out, of an economist's lumping a lot of things together oddly). Now they talk about the Cable TV outfits and the Video Cartridge outfits as though *they're* the cat's pajamas.

Having spent some considerable time around and among these areas, I have developed considerable cynicism and a bad case of the giggles. Originally it all seemed to fit together and to be leading somewhere, but talking to people at all levels, and either giving advice or trying to interpret the advice of others, I am convinced that what we have here in this whole audio-visual-presentational whizbang field is nothing less than a very high order of collective insanity. The strange way companies adopt and drop various product lines, and verbalize what they think they are doing, seem to me a combination of lemmingism and a willingness to follow any Authority in an expensive suit. I have talked to enough vice-presidents and presidents of computer companies, publishing companies, networks, media outfits and so on, to be totally certain that they have no special knowledge or unusual basis of information; yet these people's remarks, as amplified through the business reporters, send the whole nation a-dithering. There are times I think everybody in Media is either deluded, misguided, lying or crazy.

### THREE CRUCIAL POINTS.

#### 1. SYSTEMS "IN THE HOME."

The emphasis has changed from trying to sell snazzy systems to the schools (which don't have the money) to the home. This in turn has convinced most people that the new systems have to be very limited, like jimmied-up TV sets. (We easily lose track of the fact that you can have *anything* "in the home" if you want to pay for it; and an economy in which Marantzes and snowmobiles have caught on big indicates that *some* people are going to be willing to pay for really hot stuff.)

#### 2. CATCHING ON.

The key question is not how good a system is in the abstract, but whether it will *catch on*. (Obviously if we're public-spirited we want the *best* systems to catch on, of course.)

This matter of Catching On is a fickle and crucial business.

According to one anecdote, Mr. Bell couldn't interest anyone in his invention, which he was showing at some trade fair. Then who should come by but the Emperor of Brazil (!), who was about to leave

with his retinue of advisers. "What is that?" asked the Emperor of Brazil. "Nothing to bother with," they said, and tried to rush him by, but he stopped and loved it, and ordered the first pair of telephones sold. This made the headlines, and the sale of telephones began.

Another anecdote. It is legendary that inventors overvalue their own work. Yet after Thomas Edison had invented the kinematograph, or "moving picture," a device you looked into turning a crank, he *declined* to build a projector for it, saying that *the novelty would wear off*. Obviously he didn't quite see what "catching on" would mean here.

#### 3. STANDARDIZATION (from p. DM 51)

In order for something to Catch On, it has to be standardized. Unfortunately, there is motivation for different companies to make *their own little changes* in order to restrict users to its own products. The best example of how to avoid this: Philips patented its audio cartridge to the teeth, but then granted everybody *free use* of the patent provided they adhered to the exact standardization. The result has been the system's spectacular success, and Philips, rather than dominating a small market, has a *share* of a far larger market, and hence makes more money. That's a virtue-rewarded kind of story.

The other problem with standardization, though, is that we tend to standardize too soon. We standardized on AM radio, even though FM would probably have been better. (One Major Armstrong, a great figure in the development of radio, committed suicide when nobody would accept FM. If he could only have heard our FM of today, he might have said "Oh, nuts," and lived.)

Another example. When they designed the Touch-Tone phone pad, the Bell people evidently saw no reason to have it match the adding machine panel, so they put "1" in the upper left rather than the lower left. Now there are *lots* of people who use both arrangements, every day, and at least one of them curses the designers' lack of consideration.

Another interesting example of Catching On: during the early sixties, it was fun being at places where they were just getting Xerox copiers for the first time. Everyone would argue that *nobody needed* a copier. Then, grudgingly, one would be ordered. The first month's use invariably

In any medium, written, visual, filmic or whatever, you generate instantaneously an atmosphere, a patina, a miasma of style, involvement, personality (perhaps implicit), outlook, portent. Consider:

The complacency of the Sulzbergers' *New York Times*.

The cynicism and mischief of Krassner's *Realist*.

The perkiness and sense of freedom of "Sesame Street".

The personalized, focused foreboding of Orson Welles films; as distinct from the impersonalized, focused foreboding of Hitchcock.

Next to this matter of mood, all else pales: the actual constraints and structures of media, the expositions and complications of particular cognitive works and presentations within media, are as nothing.

## "Media" In The Classroom

Time after time, the educational establishment has thought some great revolution would come through getting new kinds of equipment into the classroom.

First it was movies. More recently it's been "audio-visual" stuff, teaching machines, film loops, and computer-assisted instruction.

In no cases have the enthusiasts for these systems seen how the equipment would fit into conventional education—or, more likely, screw the teacher up. Teachers are embarrassed and flustered when they have to monkey with equipment in addition to everything else, and fitting the available canned materials into their lesson plans doesn't work out well, either.

The only real possibilities for change lie in systems that will change the instructor's position from a manager to a helper. Many teachers will like this, many will not.

would exceed the estimate for the first year, and go up and up from there.

The worst aspect of the confusion among the corporations is that certain deficiencies and crudities of vision slip into the mix. Unless our new media and their exact ramifications and concomitants are planned with the greatest care, everybody stands to lose. We must understand the *detailed* properties of media. (The first question to ask, when somebody is showing you the Latest and Greatest, is: "What are the properties and qualities of the medium?" The followup questions come easily with experience: How often do you have to change it, what are the branching options, what part could somebody accidentally put in backwards, are there distracting complications? etc.)

I am unpersuaded by McLuhan. His insights are remarkable, yet suspicious: *he supposes that electronic media are all the same*. How can this be? Here we may

now decide *what* electronic media we want in the future—and this decision, I would say, is one of the most important we have to face.

The engineers seem to be quite the opposite of McLuhan: somehow to them it's always a multiple-choice, multi-engineering problem, different every time; "this technique is good for A, that technique is good for B." But the *net effect is the same*: "electronic media are generally the same." I would claim that they're *all different*, all ten million of them (TV being only one electronic medium out of the lot), and the differences matter very very much, and *only a few can catch on*. So it matters very much which. Some are great, some are lousy, some are subtly bad, having a locked-in information structure, built deep-down into the system. (Example: the fixed "query modes" built into some systems.)

## Hardware, Software And *Whatnot* (reprise)

Among the many odd things that have resulted from the collision of computer people with educators, publishers and others has been the respectful imitation of computer ways by those who didn't understand them. Again, the cargo cult.\*

The most dismal of these practices has been the adoption of the term "software" for any intellectual or artistic property. This wholly loses the distinction, made on the other side of the book, between:

hardware (programmable equipment)

software (programs, detailed plans of operation that the hardware carries out)

*contents or data* (material which is worked on by, moved in or presented by the hardware under control of the software)

In other words, hardware and software together make an *environment*; data or contents move and appear in that environment.

The publishing-and-picture folk have missed this distinction entirely. Not realizing that their productions are the *contents* (material, matter, data, stuff, message...) that come and go in the prefabricated hardware-software environments, they have mushed this together into a state of self-feeding confusion.

(The matter has not been helped by the computer-assisted instruction people—see p. DM 129—whose branching productions seemed to them enough like computer programs to be called "software.")

\*Primitives exposed to "civilized" man imitate his ways ridiculously in religious rituals, hoping for the shipments of canned goods etc., that his behavior seems to bring down from parts unknown. (See "Cybercrud," p. 27.)

## Various Media

IF WE HADN'T STANDARDIZED TV WHEN WE DID, WE'D HAVE A BETTER SYSTEM NOW. Let this be a lesson: standardize on the *best* system, not necessarily the first.

### THANATOPSYS

A self-employed repairman of mobile homes named Donald Wells has invented a solar-powered tombstone that can show movies and still pictures of the departed, along with appropriate organ music and any last words or eulogies selected by the deceased.

The device is activated by a remote control device carried by a visitor to the gravesite. The movies would be shown on a twelve-inch screen mounted next to the epitaph.

"You could also have pictures of Christ ascending to heaven or Christ on the cross, whatever you want," says Wells. "It adds a whole new dimension to going to the cemetery...."

*Cleveland Plain Dealer* (Quoted in *National Lampoon True Facts*, May '74, 10.)

Last year I actually heard a phone company lecturer say that in the future we will have "Instant Access to Anything, Anytime, Anywhere."

What they're pushing is Picturephone, which it seems to me is unnecessary, wasteful and generally unfeasible.

(See: Robert J. Robinson, "Picturephone—Who Needs It?", *Datamation* 15 Nov. '71, 152.)

Nonexistent Phone Numbers of Paris: in the French phone system, people calling the same nonexistent phone number can talk to each other; strange blindfolded encounters occur at The Number of the Day, spread by word-of-mouth; sometimes these result in people really getting together...

The High Rollers are saying they will be the information lords, and others will be the information peons, and they will nail on the world what systems they choose—for reading, writing, and the preservation of such heritage and freedom as they feel is justified by their market research and product planning. (A replay of IBM's attitude in the sixties.)

But I say that tomorrow's information and networking systems will be for the new generalists, and that means millions of people; that when tomorrow's real information systems appear, the explosion of public demand will be like the explosion of the personal computer. (And that is the need the Xanadu system has been designed to meet; see p. 141.)



One last point. Everybody only has a 24-hour day. Most people, if they increase consumption of one medium (like magazines or books) will cut down on another (like TV). This drastically reduces the sorts of growth some people have been expecting. *Except*, now, if we can begin to replace some of the inane paper-shuffling and paper-losing of the business world, and replace the creepy activities of the school (as now generally constituted) with a more golden use of time and mind. Read on.

**"When you're dealing with media, you're in show business, you know, whether you like it or not."**

**"Show business," he said. "Absolutely. We've gotta be in show business. We've gotta put together a team that will get us there."**

**I made a mental note to use the show business metaphor again, and continued. "IBM's real creative talent probably lies in other areas..."**

**Heywood Gould, *Corporation Freak* (Tower), p. 23.**

## Your Big Display Panels

All those scoreboards and wisecracking light-grids, now that they are computer-controlled, raise all kinds of possibilities for non-frame animation. The big ones cost in the millions; a small one for shopping centers costs a hundred grand (Millenium Info Systems, Santa Clara, CA).

Within a year or so, though, you ought to be able to get a nice, animated display-panel of some sort for the side of your van, assuming you've got the computer inside.

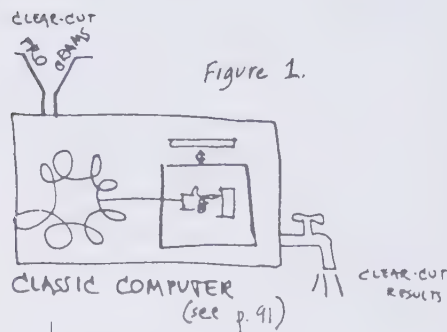
*A lot of people get the idea that they themselves will replace Gutenberg, superseding paper with the media they personally invent; and then come up with incredibly clunky designs like Picturephone, British videotext, Knight-Ridder videotext, CUBE (vote-at-home video), video conferencing, and PLATO (see p. DM 93).*

*In my opinion these have all been inane, clumsy, and (thank goodness) nonviable.*

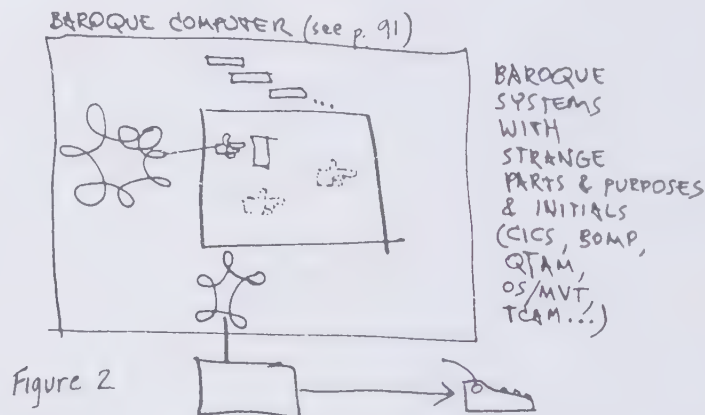
*Good design must be simple and clean. Radio, telephone, television and movies are well-designed media.*

## THE NEW ERA

A new era in computers is dawning. The first, or classic, computer era used straightforward equipment and worked on straightforward problems. (See Figure 1.)



The second, or Baroque, computer era used intricate equipment for hard-to-understand purposes, tied together with the greatest difficulty by computer professionals who couldn't or wouldn't explain very well what they were doing. (See Figure 2.)



But a change is coming. No one company or faction is bringing it about, although some may feel it is not in their interest. I would like to call it here the DIAPHANOUS age of the computer.

By "diaphanous" I refer both to the transparent, understandable character of the systems to come, and to the likelihood

that computers will be showing us everything (*dia-*, across everything, *phainein*, to show).

In the first place, COMPUTERS WILL DISAPPEAR CONCEPTUALLY, will become "transparent," in the sense of being parts of understandable wholes. Moreover, the "parts" of a computer system will have CLEAR CONCEPTUAL MEANING. In other words, COMPUTER SYSTEMS WILL BE UNDERSTANDABLE. Instead of things being complicated, they will become simple.

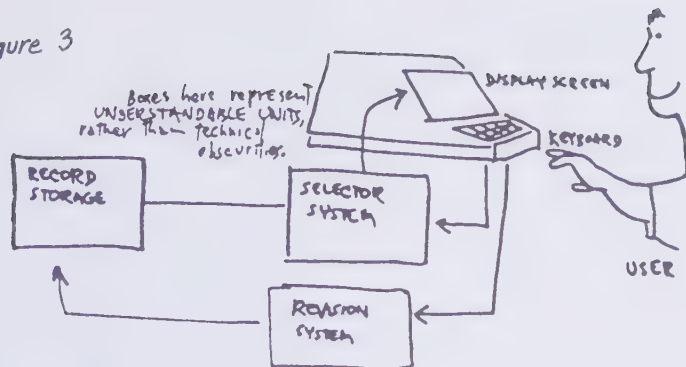
Now, many people think computers are by their nature incomprehensible and complicated—unfortunately, that's because they have been MADE TO BE. Usually this is unintentional, but I fear not always. EXAMPLE: Instead of being told, "this is the mysterious XYZ computer, it has to have things just so, you have to fill out these RMQ forms to go into the V34...", you will hear such surprisingly simple things as "This system is set up for keeping track of who owes what to the company. On the screen you can get lists of accounts and outstanding bills and who owes them; if you point at one with the light pen, the printing machine

over here will print a bill all set to go in the envelope."

In other words, systems will increasingly have UNDERSTANDABLE PARTS WITH UNDERSTANDABLE INTERCONNECTIONS. (See Figure 3.)

What is responsible for this remarkable change?

Figure 3



For one thing, smaller and smaller companies are buying computer services, and they won't stand for ridiculous complications. For another thing, a number of people in the computer field have gotten sick of systems that make things hard for

people. Finally, the price of computers, especially microprocessors (see p. 96) are coming down so fast that they can be tailored to fit people, rather than vice versa. But most of all, it's just high time, that's all.

## BIBLIOGRAPHY

- C.L. Freitas, "Making the Best Buy for the Small Business." *Computer Decisions*, March '73, p. 22-26. Compares the relative costs of mini-computers and time-sharing; concludes that minis are the best buy.
- Burton L. Katz, "Making Minicomputers Work in a Medium-sized Business." *Data Processing*, Winter 1971, p. 9-11. Stresses the point that well-designed computer systems can be used by existing personnel of a firm, without excessive complication.
- Frederic G. Withington, "Cosmetic Programming." *Datamation*, Mar. '70, p. 91-95. How to make systems friendly on the outside.

## INTERACTIVE SYSTEMS

Used to be that ordinary people had to deal with computers by filling out intricate forms, which were then translated into punch cards. The forms put things in weird categories (see "Coded-Down Data," p. 72).

No longer.

Anyway, no longer necessary.

Computer systems can now give you action, excitement—and explanations.

This is done through the magic of the TERMINAL. Terminals come in two conspicuous flavors (typewriter and screen or "boob tube") and also have two less-noticable divisions ("Teletype" or "industry" versus "IBM type").

## The Most Important Computer Terms For The 70's

Here are some phrases that will count in the new era of computing, when we will run into more and more computer systems set up for particular purposes.

### on-line

connected to a functioning computer. (Note that the computer may be in the typewriter or desk itself.)

(As distinct from *off-line*, setting things up for processing later.)

### interactive

not just connected, but responding to you. Interactive systems and programs can respond to your choices and requests, clarify what they want from you, etc.

### remote

referring to something far away, as distinct from *local*, right where you are. A computer can be either remote or local, e.g., on your desk.

### front end (n.), front-end (adj.)

whatever stands between you and a system. A front end can be the terminal in your office, for example. A *front-end program* is one which mediates between a user and some other system or program, perhaps collecting data for it by quizzing you.

### dedicated

set up for only one use. A big computer at a computing center has to have many uses; a little computer in your office can be dedicated. Dedicated computers are now hidden in all sorts of things: cash registers, for example (see "Microprocessors," p. 96).

### turnkey (adj.)

turned on with a key. Especially, *turnkey systems*, small computer systems that can just be turned on (key or not) and are fully set up, ready to run, programmed, etc.

### real-time

responding to events in the world as needed, without delays. Computer systems that control machinery, make airline reservations, predict the weather or respond to naive users are real-time. Systems that can catch up overnight are non-real-time.

### "intelligent terminal"

stupid term referring to any object that does more than act like a plain terminal. The term is stupid because it confuses distinctions. Some "intelligent terminals" have extra circuits for various purposes; others contain their own minicomputers; still others are ordinary terminals connected to front-end programs.

### user-oriented

set up for "users"—people who are not programmers or input typists, but who actually need something done.

### user level (n.), user-level (adj.)

"where the user is" mentally; his level of involvement. User-level system, system set up for people who are not thinking about computers but about the subject or activity the computer is supposed to help with.

### naive user (n.), naive-user (adj.)

person who doesn't know about computers but is going to use the system. *Naive-user systems* are those set up to make things easy and clear for such people. (We are all naive users at some time or other; it's nothing to be ashamed of. Though some computer people seem to think it is.)

### idiot-proof

not susceptible to being loused up by a naive user. The hostility in this term may in some cases be real. Computer people sometimes forget, or do not wish to tolerate, the degree of confusion that naive users bring to the keyboard. This attitude is not just their problem but everybody's, since they lay it on us.

### good-guy system

term to be used here for naive-user systems that are friendly, helpful, simple and clear.

### stand-alone system

system (regardless of purpose) which doesn't have to be attached to anything else. (May contain its own computer.)



Anyway, a terminal is something that allows a person and a computer to type at each other.

Now, computers are merely gadgets for twiddling information. They no more understand English, or human psychology, than puppies can read music. (See "Artificial Intelligence," p. DM 120.) But the computer's program can, for instance, direct the computer to type out a simple question, and compare the user's answer with a simple set of alternatives. For example, suppose the user is visiting a hospital. A computer can sign him in without the abrasiveness of a receiving nurse, and with far more patience. The following might be a sample dialogue. (Here the computer types what's in caps, and the user's replies are in lowercase.)

DO YOU HAVE AN ACUTE PAIN? (Y, N, DK)

dk

YOUR ANSWER IS: DK FOR "DON'T KNOW." DOES THAT MEAN YOU'RE NOT SURE WHAT 'ACUTE' MEANS? (ANSWER A) A PAIN COMES AND GOES? (ANSWER B)

YOU HAVE A PAIN SORT OF ON THE BORDER? (ANSWER C)

c

IS THIS PAIN IN AN EXACT PLACE YOU CAN IDENTIFY? (Y, N, DK)

y

An interactive system of this kind is called a *conversational* system, in that it "converses" with the user. The secret is that the alternatives in the computer program are few and carefully worked out beforehand: there are great pitfalls when there are too many alternatives, as when such conversational systems are used for teaching (see pp. DM 132, DM 124).

Here is a straightforward example: a system I wish I had for balancing a checkbook.

WHAT PROGRAM WOULD YOU LIKE TO RUN? ckbk

CHECKBOOK PROGRAM STARTS.

DO YOU WANT TO PROOF THE NEW STATEMENT FOR MARCH? y

PLEASE LIST THE CHECKS THAT HAVE COME IN.

231, 239, 240, 242, 244, 245.

SUM OF INCOMING CHECKS IS \$345.72. DO YOU WANT BREAKDOWN? n

PRESUMABLY BANK IS CHARGING YOU .60 FOR SIX CHECKS. ALSO MONTHLY CHARGE OF FIFTY CENTS (PLEASE CONFIRM). y

ARE THERE ANY OTHER BANK CHARGES THIS MONTH? n

ARE THE FOLLOWING DEPOSITS ON THIS STATEMENT —MARCH 1 SALARY, \$854.00? y GIFT FROM AUNT AGATHA, 14 MARCH, \$25.00? n

TOTAL ON STATEMENT SHOULD BE \$1753.21. PLEASE CONFIRM. y

YOUR CURRENT FLOAT IS \$656.75. DO YOU WANT BREAKDOWN? y

CURRENT FLOAT AS FOLLOWS—

NO. 241 IRVING'S RECORDS 7 MARCH \$6.75

NO. 243 SINISTER & MALADROIT (LEGAL FEES) 12 MARCH \$600.00

NO. 246 DOGGIE HAIRDRESSERS 12 MARCH \$20.00

NO. 247 SAM GRONK (REPAYMENT) 14 MARCH \$30.00

TOTAL \$656.75

ARE YOU DONE WITH CHECKBOOK PROGRAM? y

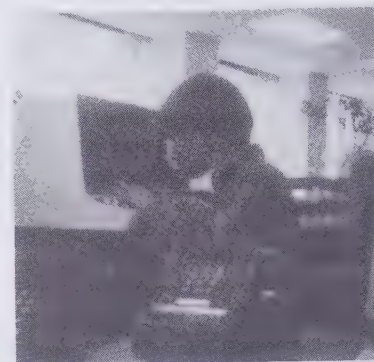
(The part shown above is easy. Thinking out the ways for the user to *correct* his records, and/or the bank, is the tough part.)

Yes, interactive systems have come in with a bang. Most of them operate on minicomputers (including desktop machines), but some are on mainframes as well. When you call for an airline reservation or to discuss your bill with a big company, the person you talk to is generally at a terminal of some sort.

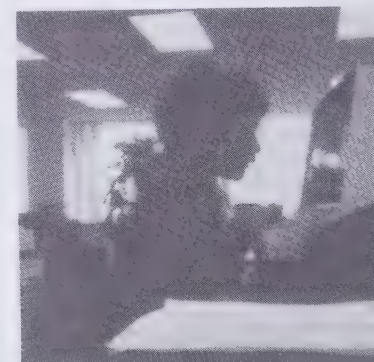
You can go to the bank after hours and withdraw money from an interactive computer terminal. Various professionals you deal with (accountant, real estate broker, even lawyer) may sit at their computer screens as they talk to you. The nurse at the hospital may record your progress on an interactive computer. Even cash registers are now interactive computers, restrained to act as if they were just plain old-fashioned *machines*.

## TERMINALS

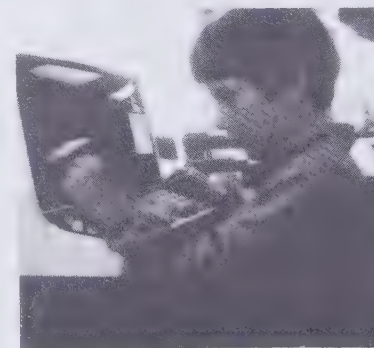
A terminal is simply any device by which a person and a computer can type at each other.



Kids love terminals. This one is a video terminal or keyscope. It allows the computer to present textual or numeric information, play games with you, quiz you for information in a good-guy system, or whatever—depending on the program, of course.



More expensive scopes (or computer displays) allow pictorial animation under the user's control (discussed throughout this side of the book). THE MAIN THING TO UNDERSTAND: what they do is decided by human beings, not "scientific principles." Human beings take note.



*"It is an insult to the human intellect to have to remember a number."*

—Donn Parker

A "GOOD-GUY SYSTEM" is a conversational computer system that is CLEAR, EASY TO USE, AND FRIENDLY.

In a conversational system, the computer can helpfully lead the user on.



Below: a "bull pen" of terminals, all hooked up to the main computer at the Chicago Circle Campus, University of Illinois. What each person does at his terminal is normally independent of what any other person does, through time-sharing of the main computer. Installations more suited to timesharing can have large numbers of terminals, all over a campus, a company or the world; see "Time-Sharing," p. 103.



**A CONSIDERATE LAYOUT**

**CARSON PIRIE SCOTT & CO.**

YOUR CREDIT LIMIT IS: \$ 500

YOUR CLOSING DATE IS: 09/06/73

\* BASED ON A "PERIODIC RATE" OF 13.5 % WHICH IS AN

\* ANNUAL PERCENTAGE RATE OF 13.7 % APPLIED TO

THE "PREVIOUS BALANCE" AFTER DEDUCTING PAYMENTS

AND CREDITS APPEARING ON THE STATEMENT.

PAY NEW BALANCE BY: ACCOUNT NUMBER

10/06/73 17-051-081-2

TO AVOID ADDITIONAL FINANCE CHARGE ACCOUNT TYPE

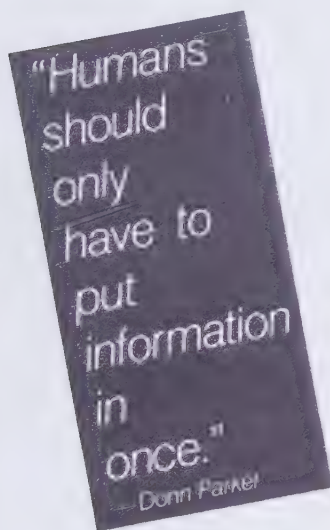
REGULAR-OPTION

**YOUR BILLING SUMMARY**

BEGINNING WITH YOUR PREVIOUS BALANCE	\$ 50.99
WE HAVE SUBTRACTED PAYMENTS AND CREDITS	\$ 30.00
BALANCE FOR FINANCE CHARGE	\$ 30.99
WE HAVE ADDED CHARGES	\$ .46
NEW BALANCE	\$ 31.44
MINIMUM PAYMENT DUE	\$ 10.00

NOTICE: SEE REVERSE SIDE FOR IMPORTANT INFORMATION AND OUR STORE LOCATIONS

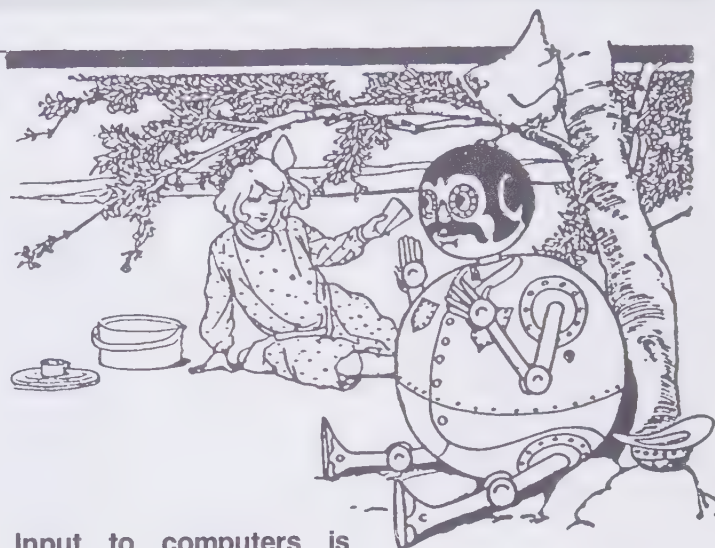
Thank you, Carson's.



## Your First Computer Contact

When you first sit at a computer terminal, the feeling is one of sheer terror. Sweat and chills, jumpiness and sudden clumsy nervous motions, lunatic absentmindedness and stammering fear and awkwardness interfere with your ability to function or understand the person who is helping you.

It's perfectly normal.

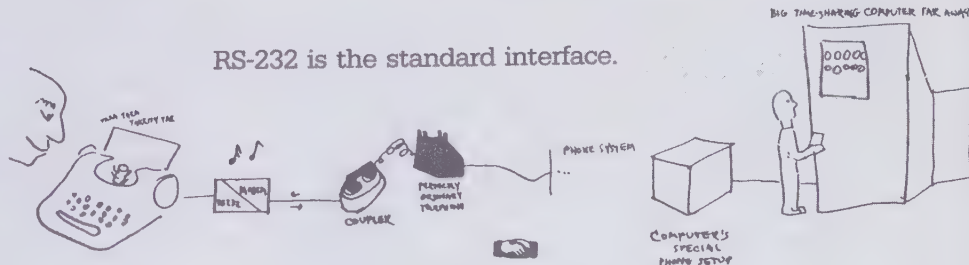


Input to computers is much easier from interactive terminals.

**THE MIRACLE OF OVER-THE-PHONE TERMINALS.** (Some people go ape just to see the typewriter going by itself.)

"Modem" takes the terminal's pulse code and warbles it into the phone as audible tones. The computer answers with similar warbles and tweedling; the modem converts that back into alphabetical characters.

RS-232 is the standard interface.





**ANY MAN OF COMMON SENSE  
CAN DESIGN A COMPUTER  
SYSTEM FOR A PURPOSE  
IMPORTANT TO HIM:**

the data structure, forms of information, general operations, record-keeping, and responses to on-line users.

*So can any woman of common sense.*

But for some reason this is generally kept a secret.

*It's not a secret anymore, and many people—perhaps millions—are doing it, now, on personal computers. This is one of the reasons for the great success of Lotus 1-2-3 and other software packages that allow a similar breadth of work.*

*However, it does take time, trouble and concentration; and that's why many people now buy software that's already set up in one way or another.*

It is because of my heartfelt belief in this kind of simplicity that I stress the creation of prefabricated environments, carefully tuned for easy use, rather than the creation of computer languages which must be learnt by the user, as do such people as Engelbart (see p. DM 16) and DeFanti (see p. DM 96). Now, their approach obviously has considerable merit for sophisticated users who want to tinker repeatedly with variant approaches. For people who want to work incessantly in an environment, and on other things—say writers—and are absent-minded and clumsy and nervous and

forgetful (like the present author), then the safe, prefabricated environment, with thoroughly fail-safe functions and utterly memorable structural and control inter-relationships, is the only approach.

The true problem that I think is emerging, though, is the problem of *system response and style*. Okay, so you're controlling widget assembly, or traffic light grids, at the CRT screen. The real question is, *how does the screen behave and respond?* This is not, darn it, a technical issue. It's psychological and then some. The design of screen activities which will enjoyably focus the user's mind on his proper concerns—no matter how personal these may be—is the new frontier of design, of art, and of architecture. But more of that later.

But the question is: What will the systems be *like*? How should they perform? What forms will information take? What conventions, diagrams, animations, ways to sign things, ways to view things... HOW SHALL IT BE?

I am afraid that as long as people are befuddled with technicalities, or confused by those who profess that these considerations are their specialty by right, we will never get it straight. Lacking time for the full discussion, I give a motto:

IF THE BUTTON IS NOT SHAPED LIKE  
THE THOUGHT, THE THOUGHT WILL  
END UP SHAPED LIKE THE BUTTON.

**Motto 1 for the new era:**

**USING A COMPUTER SHOULD ALWAYS BE  
EASIER THAN NOT USING A COMPUTER.**

**Motto 2 for the new era:**

**THE NEW FRONTIER IN COMPUTERS IS  
CONCEPTUAL SIMPLICITY AND CLARITY.**

**People who delight in intricacy are going  
to have to learn some new tricks. Internal  
intricacy is fine, as long as the user doesn't  
have to deal with it.**

**Motto 3 for the new era (to computer people):**

**MAKING THINGS EASY IS HARD.**

**Motto 4 for the new era:**

**ANY SYSTEM FOR A SPECIFIC PURPOSE  
SHOULD BE TEACHABLE IN TEN MINUTES  
OR LESS.**

**Anyone who has been taught the use of  
some fixed-purpose computer system, such  
as an airline reservation system, may doubt  
this. But perhaps this book will clarify  
things somewhat.**

Companies who want business systems  
built around minicomputers may want to in-  
vestigate companies that will put together  
whole business systems for them around  
minis.

(It is hoped that one contribution of this  
book will be to give the reader a better idea of  
what to ask for.)

Well, there are now thousands of such com-  
panies, even in small towns; some good, some  
bad. The cost of custom software, though, is  
like the cost of custom anything, so watch it.

**"Joe Turkey user"**

• A good friend of mine, Jordan Young, is a former  
• R.E.S.I.S.T.O.R. (see p. 110) and now a systems program-  
• mer (see p. 161) on the mighty Dartmouth time-sharing  
• system, DTSS. (see p. 104).

• Jordan tells me that one of the more important people  
• at Dartmouth is a mythical individual named Joe Turkey  
• User. This estimable personage knows hardly anything  
• about computers, makes a lot of mistakes, thinks he un-  
• derstands what you tell him when he doesn't, tends to hit  
• the wrong keys on the terminal, and in general tends to  
• screw up.

• But the motto up there is: "If it's not simple enough for  
• Joe Turkey User—it's too complicated."

• DTSS is a good-guy system.

# FROM INTERACTIVITY TO GRAPHICS

We have been discussing interactive computers and terminals. The most important form of interaction, though, uses screens (referred to in this book as *display screens*) and interactive graphics.

Almost everyone knows now what an interactive computer screen is, and interactive computer graphics. However, in 1974 this had to be explained.

## THE WONDER OF INTERACTIVE DISPLAY SYSTEMS

(For much more on interactive display systems, see "Computer Graphics," p. DM 80)

If you have not seen interactive computer display, you have not lived.

Except for a few people who can imagine it—and I'm trying to help you with that as hard as I can—most people just don't get it till they see it. They can't imagine what it's like to *manipulate a picture*. To have a *diagram respond to you*. To change one part of a picture, and *watch the rest adapt*. These are some of the things that can happen in interactive computer display—all depending, of course, on the program.

For some reason there are a lot of people who pooh-pooh computer display: They say it's "not necessary," or "not worth it," or that "you can get just as good results other ways."

Personally, I wouldn't think of trying to justify computer display on "practical" grounds. So what if it offers you faster access to information and pictures and maps and diagrams, the ability to simulate extremely complex things by modifying pictures, the ability to go through complex transactions with the system in very little time, the ability to create things in the world almost instantaneously (say, by creating fabric patterns which are then automatically woven, or design 3D objects which are then automatically milled by machines), and never mind that it enables the user, say, to control entire oil refineries by the flick of a light pen.

As far as I'm concerned, these matters aren't very important compared to changing the world: making education an

excitement rather than a prison; giving scholars total access to writings and notes, in new complex form; allowing people to play imaginatively, and raising human minds to the potentials they should have reached long ago; and helping people think at the deepest level about very heavy and complex alternatives—which confront us more ominously today than ever.

## WOMEN, CHILDREN AND MEN AT WORK

The computer display screen is the new frontier of our lives.

That such systems should (and will) be *fun* goes without saying. That they will also be a place to *work* may be less obvious from the tone of this publication, so I want to stress it from here.

Once you pass through the paradigm, to realize that computer screen systems will be the center of every kind of work, then the silly terminology that's been with us—terms like "microcomputer" and "word processing" and "CAD/CAM"—become seen for how inane they really are.

## DESKTOP ANYTHING

At desktop computer screens—be you professional, student or some other type—you can now write books and type-set them, design buildings, compose music, do experiments in chemistry and physics, and much more. Whatever can be done through paper, blackboard, loud-speaker or machine tool can be done, or nearly done, on some computer screen.

A popular term right now is "desktop publishing." But "desktop engineering" (presently called something else; see p. DM 83), desktop whatever, desktop *anything*, is here or on the way.

The question is not whether it's "as good as" or "not as good as" what was previously done. The question is how to make it great.

(Another question is, why does the screen have to take up desktop space? But that's a separate issue.)

The computer display screen will be mankind's new home.

IF COMPUTERS ARE THE WAVE OF THE FUTURE, DISPLAYS ARE THE SURFBOARDS.

## Another Viewpoint

(from handout, 1974 Natl. Joint Comp. Conf.)

John B. Macdonald  
Research Leader  
Computer Applications: Graphics  
Western Electric Company  
Engineering Research Center

## PROBLEMS, PERILS, AND PROMISES OF COMPUTER GRAPHICS

I would begin with some definitions which may be obvious but bear repeating.

1. Engineering is the application of science for (\$) profit,
2. Computer graphics does not make possible anything that was previously impossible: it can only improve the throughput of an existing process,
3. A successful application of computer graphics is when over a period of five years the cost savings from improved process throughput exceed the costs of hardware, software, maintenance and integration into an existing process flow.

## Saving Energy With Computer Display

A timely criticism of computer display is that it needs electricity. But (as mentioned elsewhere) it saves paper, and importantly, it bodes to save energy as well.

IF WE SWITCH TO COMPUTER SCREENS FROM PAPER, PEOPLE WON'T HAVE TO TRAVEL AS MUCH. Instead of commuting to offices in the center of town, people can set up their offices in the suburbs, and share the documentary structure of the work situation *through the screens*.

This view has been propounded, indeed, by Peter Goldmark, former director of research for CBS Labs, the man who brought you the LP record.



Making pictures with the GE halftone system (see pp. DM 100).



# Display Terminals

Some computer displays have to be deeply attached to a computer and some don't. These latter we call *display terminals*.

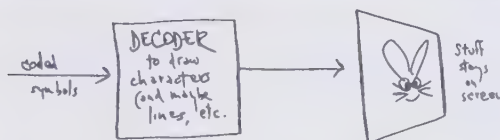
A display terminal is like an ordinary computer terminal (see pp. DM 10-11): that is, fundamentally a *device by which a computer and a person can type at each other*. However, display terminals have screens.

Now, some display terminals only show *text*, just like ordinary printing terminals (described on the other side), but manufacturers are free to add any other features, and so different manufacturers make it possible to do various kinds of picture-making with their particular display terminals, if appropriate programs are running in the computer that controls them.

Some devices are sold as display *terminals* but actually, to further confuse the issue, contain complete minicomputers. (The fact that the manufacturer may not stress this is simply a marketing angle he has chosen.) Similarly, certain terminals contain microprocessors (see p. 96), which means they can be programmed to behave like various other terminals, but ordinarily they cannot be programmed to do much else by themselves.

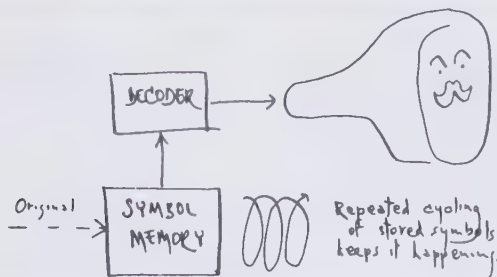
Without getting into it deeply, there are two main types of display terminal: those that are *refreshed* and those that are not. A refreshed display is one whose viewing surface fades and must be continually re-filled; a non-refreshed display somehow stores the presentation in the viewing surface itself.

Non-refreshed displays simply take the symbols from the computer, blam them onto the screen, and that's it until the screen is erased (by either the computer or the user).



Two major types are the *storage tube* and the *panel*. These in turn have separate subtypes, etc.

Refreshed displays have to have some other kind of symbolic (digital) memory, whose contents repeatedly go to the screen:



## Repeated cycling of stored symbols keeps it happening.

Most refreshed displays use an actual television screen—that is, a CRT (see pp. DM 84-86) whose entire area is repeatedly repainted by the electron beam.

Since computers send text out to terminals as individual alphabetic and punctuation codes, each terminal must contain circuitry to change the character code to a visible alphabetical character on the screen. Such a piece of circuitry is called a *character generator*. There are various kinds, they go at various speeds, some offer more different characters than others.

Display terminals generally have a little marker, or *cursor*, that the user or the computer can move around the screen. The computer can sense what the user is pointing at by the motion codes it gets, telling where the user has moved the cursor.

Note that the term *video terminal* is often used, incorrectly, for any display terminal. The term "video" should only be used when the screen is refreshed by an actual video raster. (See "Lightning in a Bottle," pp. DM 84-86.)

*Text terminals* (also called alphabetic terminals, character terminals or key-scopes) simply show written text, put in either by the computer or the user. (Some terminals, called *transaction terminals*, can be divided up into specific areas that the user may and may not type into—for banking and stuff. However, whether that form of terminal is necessary may also be a matter of taste in the program design.)

Text terminals range in price from, say, \$1500 on up to \$6500. (This last is the price of a remarkable *color* text terminal demonstrated by Tec, Inc., at the 1974 National Computer Conference. Each alphabetic position could contain a letter and/or a bright color; altogether the screen could hold big colorful pictures made up of these bright spaces. Ostensibly just a text terminal, actually the device could be regarded as an Instant Movie Generator for television animation. But it may take Tec, Inc. awhile to realize what they have created.)

*Graphic terminals* offer some kind of pictures on their screens. These come in a great variety: line-drawing, some without, some with levels of gray. Of interest to the beginner are:

"The Tektronix." (Also called "the greenie," or "the green screen.") Tektronix, Inc., makes a display based on a pale green storage tube they make. (So does Computer Displays, Inc.) Such displays allow you to put more and more text and pictures on a screen, crowding it all up—but you can't take the lines or words off individually.

"The PEP." Excellent (but very expensive) display that comes out to a video screen from a high-resolution storage tube. Permits grey scales and selective erase. Princeton Electronic Products.

**REFRESHED HIGH-RESOLUTION COLOR SYSTEMS.** A number of companies manufacture computer displays allowing complex grey-scale pictures, including color. They are expensive but very nice. Indeed, if you buy them in clusters, these fancy-picture scopes can cost as little as text terminals.

*GOING FOR BAROQUE: the one they built for the PLATO system* (see p. DM 15).

THAT EXTRAORDINARY TERMINAL—THE CAVE OF PLATO.

**IF PLATO JUST USED THE PLASMA SCREEN, it could just HANG ON THE WALL LIKE A PICTURE. But the plan to have all those projected pictures, too, makes the PLATO terminal one weird object.**

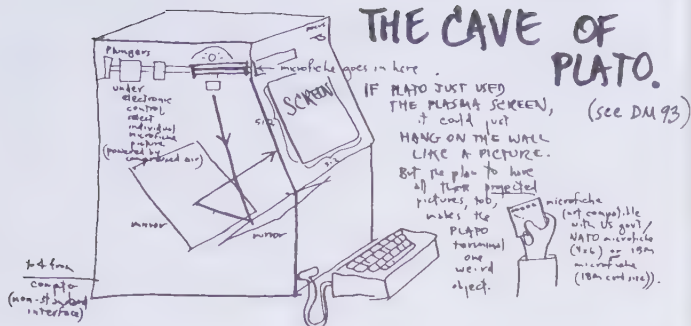
Microfiche (not compatible with US govt NATO microfiche (4 x 6) or 3M microfiche (IBM card size)).

You can read the standard-size lettering off the screen at SIX FEET—even though it's NO BIGGER THAN PICA TYPE. Fantastic.

The internal circuitry that draws on the screen is highly capable. Receiving a 20-bit code, the terminal itself deciphers it as

A LINE ON THE SCREEN, or

TWO STANDARD CHARACTERS ON THE SCREEN from its FIXED character memory, or



TWO SPECIAL CHARACTERS ON THE SCREEN from its CHANGEABLE character memory (which can be loaded with Russian, Armenian, Katakana, Cherokee, or whatever, even *little pictures*, at the start of the lesson), or

A COMMAND TO THE MICROFICHE PROJECTOR, or

A COMMAND TO THE AUDIO PLAYER, or

A COMMAND TO WHATEVER'S IN THE GENERAL JACK.

Note that all lines and characters for the plasma screen can be turned on (orange on black) or off (black on orange).

#### WHERE ARE THEY NOW?

The PLATO terminal, tied to the original system, is a bit of a dinosaur. There are PLATO installations here and there, but it wasn't the big hit its developers foresaw. See "Plato's Retreat," p. DM 95.

The thing about display screens—especially the high-performance, subroutining kind—is that the screen can become a place from which to control events in the outside world.

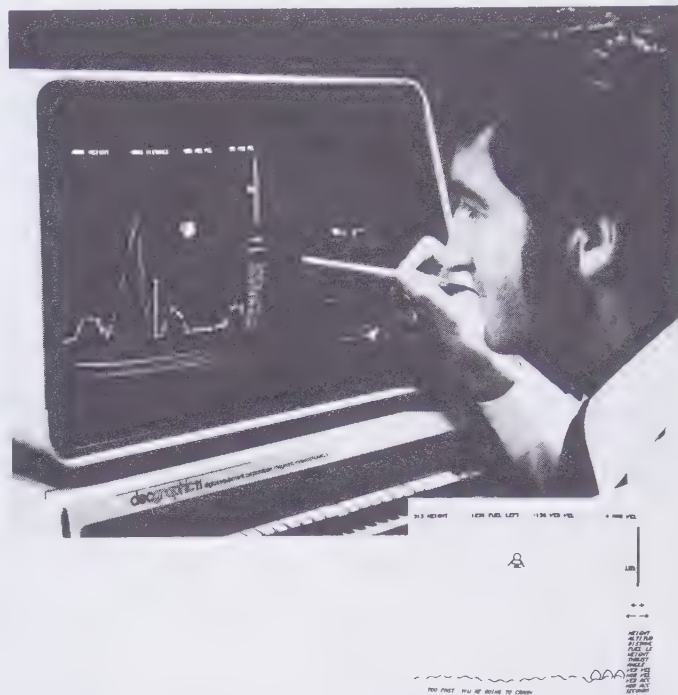
Example: I believe a town in New York state has its electrical system hooked up to an IDIOM subroutining display (made by Information Displays, Inc., and coupled to a Varian 620 minicomputer). Instead of having a wall with a big painted map having switches set into it, like many such control centers, the switches are linked directly to the minicomputer, and a program in the minicomputer connects these circuits to the pictures on the screen. Thus to throw a switch in the real world, the operator *points with his lightpen at the picture of the switch*, and the minicomputer throws the switch.

There are oil refineries that work the same way. The operator can control flows among pipes and tanks by pointing at their *pictures*, or at symbols connected with them, and bingo, it happens Out There.

In another case, a person designing something at a screen can look across the room and see a machine producing what he just finished designing a few minutes ago. I wish I could say more about that particular setup.

This honey is the GT-40 from DEC (\$12,000, including computer—the thing with teeth, below). It's a subroutining display (see p. DM 88).

Man is playing Moonlander game: controlling screen action with light-pen. Computer simulates real moon lander.



There are now thousands of brands of graphical computer; at least half are based on the IBM PC or its clones (see pp. 48-49). Incredibly opulent graphic machines are now available.

Note that "Moonlander" was one of the first computer games, developed by Dave Ahl at DEC. Dave went on to found Creative Computing, which lasted a little over ten years, and helped whip up enthusiasm for personal computers. "Moonlander" came out in an arcade version.

#### GENETIC WORKSTATIONS

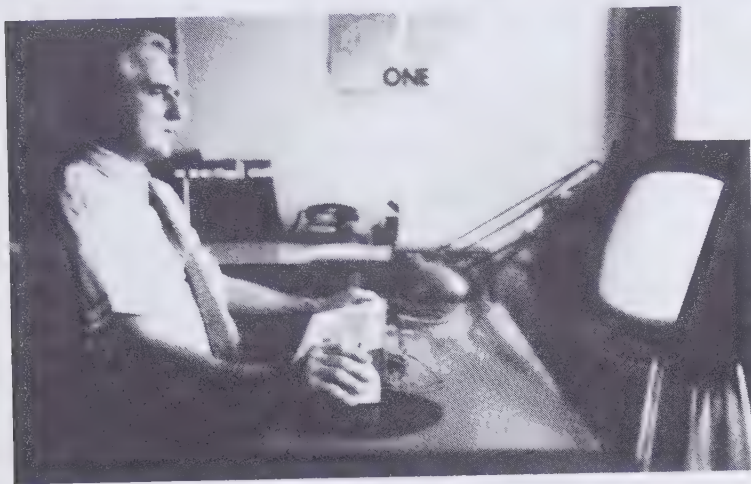
used to be just barns and beds. Today, though, the operator can actually have the workstation create a DNA chain with an arbitrary sequence. From the screen. Scary?

There are workstations now set up for almost every profession, every occupation, every and every price. You can easily pay a hundred thousand dollars for a hot workstation, and get an excellent personal computer as well (see p. DM 113).

Display terminals aren't so common now, because the display is typically built right into the computer. A desktop machine with screen is often called a workstation now.



# DOUG ENGELBART AND "THE AUGMENTATION OF INTELLECT"



Perhaps the largest interactive use of computers now is for text: so-called "word processing," as well as various activities by which text is shared, such as "electronic mail" and "networks" (see p. DM 39).

One man started all this, and pursued it with a unique and powerful vision. He invented what we now call "word processing," "outline processing," the mouse, dividing the screen into windows, sharing material between different users' screens and workstations. This individual is well-known to computer people, but nowhere near as well appreciated by the public as he should be. His name and his chosen mission should be known to all:

Douglas Engelbart is a saintly man at Stanford Research Institute whose dream has been to make people smarter and bring them together. His system, on which millions of dollars have been spent, is a wonder and a glory.

He began as an engineer of CRTs (see "Lightning in a Bottle," p. DM 84); but his driving thought was, quite correctly, that these remarkable objects could be used to expand man's mind and improve each shining hour.

Doug Engelbart's vision has never been restricted to narrow technical issues. From the beginning his concern was not merely to plank people down at display consoles, but in the most profound sense to expand man's mind. "The Augmentation of Human Intellect," he calls it, by which he means making minds

work better by giving them better tools to work with.

An obvious example is writing: before people could write things down, men could only learn what they experienced or were told by others in person; writing changed all that. Within the computer-screen fraternity, the next step is obvious; screens can double and redouble our intellectual capacities. But this is not obvious to everybody. Engelbart, patiently instructing those outside, came up with a beautiful example. To show what he meant by the Augmentation of Intellect, Engelbart tied a pencil to a brick. Then he actually made someone write with it. The result, which was of course dreadful, Engelbart solemnly put into a published report. Not yet being able to demonstrate the augmentation of intellect, since he had as yet no system to show off, he had masterfully demonstrated the *dis*augmentation of intellect: what happens if you make man's tools for working out his thoughts *worse* instead of better. As this poor guy was with his brickified pencil, explained Engelbart, so are we all among our bothersome, inflexible systems of paper.

Starting small, Engelbart programmed up a small version of what most fans call "The Engelbart System" some ten years ago. One version has it that when it came to looking for grants, management thought he acted too kooky, and so assigned a Front Man to make the presentation. But, as the story goes, the man

from ARPA (see "Military . . .", p. 151) pointed at Engelbart and said, "We want to back *him*."

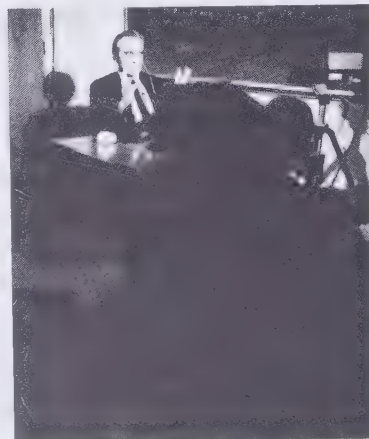
A small but dedicated group at SRI has built up a system from scratch. First they used little CDC 1700 minicomputers; then, various grants later, they were able to set up their own PDP-10, in which the system now resides, and from which it reaches out across the country.

Doug calls his system NLS, or "oN-Line System." Basically it is a highly responsive, deeply-structured text system, feeding out to display terminals. From a terminal you may read anything you or others have written, and write with as yet unmatched flexibility.

The display terminals are all over. The project has gone national, though at great expense: through the ARPA net of computers, you can in principle become a user of NLS for something like \$50,000 a year.

## THE "KNOWLEDGE WORKSHOP"

For a lucky fifty or so people, Engelbart's system is Home. Wherever they are—at Stanford Research Institute or far away on the ARPAnet—a whole world of secretarial and communication services is at their fingertips. The user has but to call up through his display terminal and log on. At that point all his written files, and numerous files shared among the users, are at his fingertips. He may



Engelbart meets with someone far away, as others watch.

read, write, annotate the cross-link. (Engelbart's system has a provision for collateral structuring: see "Thinkertoys," p. DM 50.) He may send messages to others in the Workshop. He may open certain of his files to other people, and read those that have been opened to him.

This all has a certain vagueness if you do not understand how bound you are today by paper—the problems of finding it, sorting it, looking things up. (If you *write*, that is, write a *lot*, you know all too well how intractable is paper, what a damned nuisance.) With a system like Engelbart's, now, whatever is written is instantly *there*. Whatever you want to look up is instantly *there*, simultaneously interconnected to everything else it should be connected to—source materials, footnotes, comments and so on. A document is completed the moment it is written; no human being has to retype it. (It need not be typed on paper at all, if it's just for the workshop members; a printout is only needed if it has to go to someone outside the system.)

In many ways, Engelbart's system is a prototype for the world of the future, I hope. ALL HANDLING OF PAPER IS ELIMINATED. Whatever you write, you write on the screens with keyboard and pointer. (No more backs of envelopes, yellow pads, file cards, typewriters.) Whatever you transmit to fellow users of the system you simply 'release'—no physical paper changes hands.

The group has also worked out some remarkable techniques for collaborative endeavor. Two people—say, one in California and one in New York—can work together through their screens, plus a phone link; it's as if they were side-by-side at a magic table. Each sees on his screen what the other sees; each controls a moving dot (or "cursor") that shows where he's pointing. The effect is somewhere between a blackboard and a desk; both may call up documents, point things out in them, change them, and anything else two people might do when working on something together.

## THE SYSTEM ITSELF

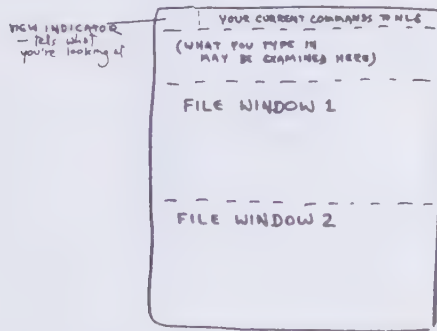
Basically the system is a large-scale setup for the storage, bringing forth,



viewing and revision of documents and connections among them.

The documents are stored (of course) in alphabetical codes. Connections among them, or other relations within them, are signalled by the presence of other codes *within them*; these are ordinarily not displayed, however, except as directed by a particular display program and display programs can of course vary.

There are various programs for display, in large part depending on what sort of screen system the individual user has. (NLS is used with everything from high-resolution line-drawing screens converted to 1000-line television, down to inexpensive Delta Data terminals—a brand, incidentally, that allows text motion, which most don't.) Engelbart's system is *extremely general*, allowing the creation of files having all kinds of structures, and display programs in all kinds of styles. (I hope that this side of the present book conveys a sense of how many styles that can be.) However, most users are devoted to certain standardized styles of working that have been well worked out and permit the easy sharing of material and of operating practices. Here, for instance is their standard screen layout:



Two separate panels of text appear, and links may be shown on them. (Thus it's a thinkertoy—see p. DM 50.) Two little windows at the top remind you of what you're seeing and what you're asking for. We can't get into the rest of it here.

## THE COMMAND LANGUAGE

NLS has a command language which all users must learn. While it is a streamlined and straightforward command language, nevertheless it requires the user to type in a specific sequence of alphabetical characters every time he wants something done. (This is acceptable to computer-oriented people; I suspect it would not be satisfactory, say, for philosophers

and novelists. For designs oriented to such users, see JOT (p. DM 71), the basic Hypertext system (p. DM 29) and Parallel Textface (p. DM 41).

Incidentally, NLS users may also employ a cute little keyboard, something like a kalimba, that allows you to type with *one hand*. You simply type the six significant ASCII bits (see p. 75) in one "chord"—it sounds hard but is easy to learn.

Sample commands: I (insert), D (delete), M (move or rearrange). Then you point with the mouse.

### MOUSE?

The Engelbart Folks have built a pointing device, for telling the system where you're pointing on the screen, that is considerably faster and handier than a lightpen. (Unfortunately, I don't believe it's commercially available.) It's called The Mouse.

## WHERE ARE THEY NOW? —THE MOUSE

*The mouse was adopted wholesale at Xerox PARC, from whence it was assimilated to the Macintosh, the first mouse really to go public. Now mice come with the Mac, the Amiga, the Atari ST, and a*

*number of professional workstations; they are available for any computer (Kurta even sells a hamster—that is, no tail; works by infrared). But the problem is always whether your software on (say) the PC will accept its squeaks as input.*

The Engelbart Mouse is a little box with hidden wheels underneath and a cable to the terminal. As you roll it, the wheel's turns are signalled to the computer and the computer moves the cursor on the screen. It's *fast and accurate*, and in fact beats a lightpen hands down in working speed.

Through the command language, NLS allows users to create programs that respond in all sorts of ways; thus the fact

that certain text-handling styles are standard (as in illustration of screen layout) results more from tradition than necessity.

The same apparently is true of the data structure. I used to be somewhat disturbed at the way Engelbart's text systems seem to be rigorously hierarchical. This in fact is the case, in the sense that having multiple discrete levels is built deep into the system. But it turns out to be harmless. The stored text is divided by the storage techniques into multiple levels, corresponding to a Harvard outline. Think of it as something like this:

### 1. HIERARCHICAL FORMAT

- A. STORAGE
- B. DISPLAY
- C. LANGUAGE

But let's expand this example a little:

### 1. HIERARCHICAL FORMAT

- A. STORAGE
  - A1. Everything in NLS is stored with hierarchical codes.
  - A2. Their effect depends on the display.
- B. DISPLAY
  - B1. The hierarchical codes of NLS have no consequences in particular.
  - B2. The hierarchical codes for NLS can splay the material out into a variety of display arrangements,
    - B2A. They can be displayed in outline form.
    - B2B. They can be displayed in normal text form.

B2C. These datted numbers can even be made to disappear.

### C. LANGUAGE

C1. The command language determines what the display shows of the hierarchical structure.

C2. *What is shown can be determined by a program in the command language.* (For instance, "how many levels down" it is being shown.)

C2A. This is four levels down. (The earlier example wasn't.)

C3. The display format all depends on what display program you use, in the NLS command language.

That's enough of that. I can't help remarking that I still don't like that sort of structuring, but it is deep in NLS, and if you don't like it either (poor deprived lucky user of NLS) you can program it to disappear, so it's hardly in your way.

## WHERE ARE THEY NOW?

*When Stanford Research Institute went private, the AUGMENT project was sold to a firm called Tym-Share, which proceeded to offer AUGMENT to corporate users on a time-sharing basis. Then it all became a part of McDonnell-Douglas, makers of DC-series aircraft and much else.*

*Doug went along for the ride. At McDonnell-Douglas he meets with prospective users of Augment and works on extensions to the design. When he gets a chance he gives inspirational talks at conferences.*

By The Beard  
Of The  
Prophet!

Engelbart in German means Angelbeard; Doug Engelbart is indeed on the side of the angels. In building his mighty system he points a new way for humanity. The sooner the better. Any history of the twentieth century will certainly hold him high. Few great men are also such nice guys.

AUGMENT can be used from Tymnet (now a service of McDonnell-Douglas) for \$10 to \$15 an hour. Unfortunately this adds up rapidly, especially considering that you want everybody in your office attached to it all the time. Unfortunately the alternative way to get AUGMENT is still to acquire a PDP-10 or workalike (now rare) and run it with the same enhanced operating system they use.

AUGMENT requires weeks of learning, which has held it back. But the system is very powerful. Its uses include document control, collaborative work, office team coordination.

AUGMENT can be given different front ends in different styles, provided they send and receive its standard protocol.



## BIBLIOGRAPHY

Douglas C. Engelbart, Richard W. Watson & James C. Norton, "The Augmented Knowledge Workshop." Proc NCC 73, pp. 9-21.

Charles H. Irby, "Display Techniques for Interactive Text Manipulation." Proc NCC 74, pp. 247-255.

When I asked him recently, Doug recommended these as the best summary of his work:

D. C. Engelbart, "Toward Integrated, Evolutionary Office Automation Systems." Proceedings of the Joint Engineering Management Conference, Denver CO, 16-18 Oct 78, pp. 63-68.

D. C. Engelbart, "Evolving the Organization of the Future: A Point of View." In Robert Landau, James Bair and Jeanne Siegman (eds.), Proceedings of the Stanford International Symposium on Office Automation, March 23-25 1980. Ablex Publications Corporation, Norwood, NJ.

D. C. Engelbart, "Toward High-Performance Knowledge Workers." Proceedings of the AFIPS Office Automation Conference, San Francisco, April 5-7 1982, pp. 279-290.

D. C. Engelbart, "Collaboration Support Provisions in AUGMENT," Proceedings of the 1984 AFIPS Office Automation Conference, Los Angeles CA, February 20-22 1984, pp. 51-58.

D. C. Engelbart, "Authorship Provisions in AUGMENT." Proceedings of the 1984 COMPCON Conference, San Francisco CA, February 27-March 1 1984, pp. 465-472.

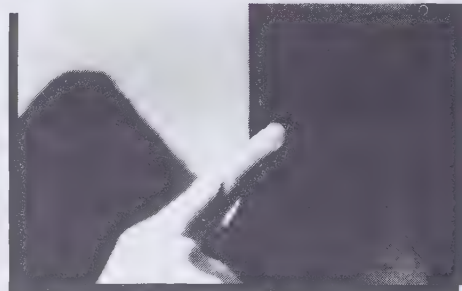
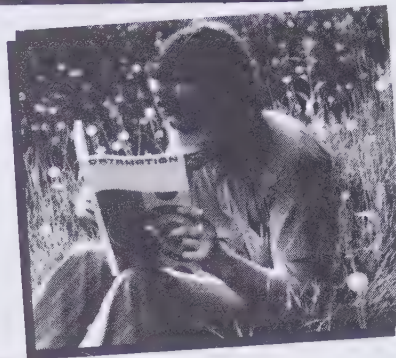
D. C. Engelbart, "The Augmented Knowledge Workshop," ACM Conference Proceedings on the History of Personal Workstations, Palo Alto CA, January 9-10, 1986.

HISTORICAL: "A Research Center for Augmenting Human Intellect," Proceedings FJCC 1968, pp. 395-410.

There is a videotape kicking around of Doug's great presentation at one of the Joint Computer Conferences in the sixties, where he shared a screen with a colleague on the other side of the country. That highly inspirational tape may or may not become available.

## THE AUGMENTATION OF INTELLECT.

Infamous Ape Sequence from my slide show.



Actually it needs the "2001" music.

## "THE OFFICE OF THE FUTURE"

Everybody talks about the office of the future—always meaning at least three years from now—but nobody agrees on what it will be. The office of the future is claimed as a special preserve by psychologists, chair designers, spreadsheeters, video people and anthropologists.

Old-fashioned territoriality at work; in today's world of abstractions, any empty space is immediately claimed by every adjacent field, which is a lot of fields.

But when it comes to territoriality, we must note the claims of the biggest animal:

"That the office of the future will be highly automated, more complex, and based upon information-processing systems is now fairly obvious, as is the fact that IBM will be the dominant force in this

You can't read the screen here. It says:

COGITO ERGO SUM

$E = M * C^{**2}$

Call me Ishmael.



It really needs the music.

evolving field." Robert Sobel, IBM: Colossus In Transition. Times Books, 1981; quote from Bantam edition, p. 329.

Ha ha ha.

The office of the future will not be more complex but more simple.

It will not be based upon "information-processing systems" of any traditional kind, but around shared memory with version control.

And IBM's ability to make things simple is still highly dubious.

The office of the future, in the opinion of the author, will have nothing to do with the silly complexities of automatic typing. It will have screens, and keyboards, and possibly a printer for outgoing letters, but possibly not. All your business information will be callable to the screen instantly. An all-embracing data structure will hold every form of information—numerical and textual—in a cat's cradle of linkages; and you, the user, whatever your job title, may quickly rove your screen through the entire information-space you are entitled to see.

(See "Xanadu," p. DM 141.) And you will be able to work on these objects with great new simplicity and clarity.



There are many lessons to be learned from video games that should affect the design of office systems and desktop working environments of every kind.

One is that user performance, given truly responsive systems, can far exceed the greatest performance you expect. But commercial companies don't get this yet. I know of a company where engineers foolishly set a maximum typing speed in the equipment!

Instead, controls should be as fast, flexible and subtle as possible; designers should study in particular the handlike animated pointers in Tempest, and the extraordinary smoothness of their response.

Animated symbols (like the crawling and flipping menaces in Tempest), can present fast, powerful visualization tokens, far more useful to tomorrow's power user than static icons.

And simultaneous maps giving larger views (for example, your current position in a large document) can be a powerful parallel visualization.

The higher levels of play that are ramifications waiting to be discovered will be a vital part of tomorrow's software. Here study Pac-Man, Ms. Pac-Man, and Qbert.

For a feeling of tomorrow's most powerful office systems, I would recommend the study of Tempest, or (less attractively and powerfully) the upper speeds of Ms. Pac-Man.

# GAMES

Games we have had from time immemorial.

Well, **computer games** were off and running in the sixties, if only as an elitist occupation. There was DARWIN at Bell Labs (see "Survival of the Fittest," p. 114), Dave Ahl's "Lunar Lander," and the graphically exciting "Spacewar," developed at MIT and then played at the Shy Company (actually Xerox PARC), described by Stewart Brand in the early seventies (see p. 115). And the PLATO games of the early seventies (see p. DM 20) gave a hint of things to come.

## PONG

Then came "Pong," and the advent of video games.

At first (Pong and the next few), the games were just wired up individually; but soon the new cheap microprocessors made possible the great video-game explosion.

The hit of '77 was the dread-inspiring "Space Invaders," aliens sinking toward you with a deadly thump-thump-thump. You, and the world, always lost.

Soon bright and jingly video games were everywhere. Created mostly in Japan, for some peculiar reason they were all tried out in one parlor in Chicago, and only if they succeeded there would they be imported in quantity.

These were of course computer games, just as much as Spacewar or DARWIN had been, but they did not come out of the

same academic tradition. They were shoot-em-ups, out of the old shooting-gallery tradition, involving simple war-to-the-death between you and whatever else. (And the word "computer" still whispered to the public of great locomotives in airconditioned rooms, so someone hit on the diabolically silly term "video game" instead.)

Players rose to levels of performance undreamt of by the games' designers. (As I recall it, Space Invaders had **too few digits in the scoring mechanism** to accommodate the scores encountered.)

1982 was the high time. Video-game parlors were everywhere, with males from young to middle age dumping every possible quarter to die every possible death in every possible screen incarnation. In that year alone, if memory serves, the game of Pac-Man, all by itself, grossed **five hundred million dollars** in the United States, equal to a substantial fraction of the entire film industry.

And then, just as suddenly, video games died. The parlors emptied out. Other pursuits, perhaps more social, called the customers away.

## THE SCORE

There was an extremely creative side to video games, if we ignore the puerile and violent aspects. That was the artistic and fanciful side—the wonderful colors and responding animations, the jingle jungle of sound (all digitally stored also; see p. DM 60); and, best of all, the tricks of presentation and control. Little maps and alternative views that showed, in parallel, where you were on a larger setup (as

A lot of teaching games are being marketed. Various productions use Sesame Street characters. There are programs with titles like "Dungeon of the Alricula much."

Some military training, e.g. of tank drivers, is now done at actual cheapo video-game-type simulators. Battlefield commanders are trained in much more powerful simulations. (See "Simulation," p. 149).



in various driving games, *Wizard of Wor*.) Extraordinarily sensitive and clever controls (*Tempest*). Scrolling playfields (*Baby Pac-Man*).

Some of these games, in their particular ways, were truly works of art. Foremost in beauty was Atari's *Tempest*, abstract and stunning (and intensely responsive to a degree, and in a style, which designers of all interactive systems would be well-advised to study in detail). Others of particular visual and interactive quality were Donna Bailey's *Centipede*; *Qbert*, which had a hilarious surprise sight gag at about the fourth round; *Pac-Man* and especially his bride *Ms. Pac-Man*, one of the most playable games and the one that most appealed to women;

*Crystal Castles*; and of course the milestone *Space Invaders*. On home machines, I would claim Art Status for *Star Raiders* and *Ballblazer* (discussed on p. DM 69) for the Atari 800; and on the Apple II, for a hauntingly beautiful game called *Pollywog*, by Alan Wootton, where hundreds of tadpoles hatch—each is actually a game-of-life figure (see p. 115)—and swim in schools, this way and that, through beautiful colors, losing members constantly to predators as we hear jingly algorithmic music. (The player guides the whole school.) Attract mode is wonderful to leave running. A stunner.

So far these games have influenced of-  
fice and general software design all too lit-  
tle. But there's a new generation coming.

In the mid-seventies, Scott Guthery started Computer Recreations, Inc., to provide a time-shared game service across the country. It would have gone over big, I think, like PLATO games. But he could find few investors. Ahead of his time?

At Oberlin College, they did a "human video game." Students suspended a TV camera from the gymnasium ceiling, ran it to the official Game Screen. Students assigned to be the characters put on brightly-colored coolie hats with team insignia and got into position. Students assigned to be the players—since there was no electronic way for them to control the characters—shouted out their instructions. Merry chaos ensued, and a good time was had by all.

## PLATO GAMES

They work hard and they play hard on the mighty PLATO system. (See p. DM 93.)

When the Author gets tired of Authoring, or the Student of Stewing, just around the corner, a few keystrokes away, are diversions and games to boggle the imagination.

You can go to a program ("lesson rose") and look at "the great roses"—elaborate curlicues generated by mathematical patterns that appealed to the authors of that program; or find, also tucked in rose, Conway's Game of Life (see write-up, p. 115, and picture series nearby.)

Then there are games you can play against the system, like *racetrack* and *blackjack*. (These games let you win astronomical sums of money—play money, forgotten when you sign off.) Remember, of course, that you're not really playing against a computer but against a specific program, with its quirks and shortcuts and blind spots.

Then there are games you play by yourself—actually responding resources (see p. DM 133), which entice you into trying things out. Tenczar himself has created two elegant, gem-like lessons, *man* and *picto*, which teach you computer programming without ever saying so. These two programs present the user with a little picture of a man on the screen, and show him how the little man may be moved around and made to pick up pictures of balls. From there on, the

student may have his way, and is never told that he's learning to program in a true computer language. (Though it is a quite restricted one, dealing exclusively with little men and their excursions among balls and falling sticks.)

Another charming game, I don't know by whom, is called *candy factory*. Here too, the user may control the animation of the picture by what he types. Machines are seen to manufacture candy, box it, and ship it—depending on what buttons you press.

Some games are played between people who sit together before a single PLATO terminal, often with teaching intent. Such games include the *hop game*, where Bunny (you) and Frog (your friend) add their way along a board with numbered squares. Older children can dig *How the West Was (1 + 2)x3*, which involves grouping the numbers you get by chance to try to get ahead of the other stagecoach.

## THE "BIG BOARD" GAMES

Still another category of games, though, awaits the adult who craves real excitement. Because PLATO has so many terminals, all over, there is a curious combination of anonymity and intimacy between users (much like the curious Nonexistent Phone Numbers of Paris).

Anyway, the Big Board games of PLATO have exactly that: a shared list, or "Big Board," showing *who* is playing the specific game.



Moonwar on a Saturday in Urbana. It's man-to-man among the craters; then a quick kill of the unknown adversary.



And our doughty warrior looks to the Big Board for more challengers. Kids love PLATO games.

But you don't have to use your right name.

In this jaunty society of shadows, you pick your own *nom de guerre*, or fighting name. This has numerous advantages: the most obvious is that as you improve at play, you can shed the identity in which you have been humiliated.

The main games with Big Boards are that old standby, *spacewar* (rocketships wheeling and firing at each other and sliding around on the screen); *dogfight* (biplanes wheeling and firing at each other and sliding around on the screen), *moonwar* (shooting at the other guy by specified angles as you stand among craters). In addition, PLATO offers (not during working hours) what must be two of the most baroque space-war games anywhere, *empire* (eight races—the Vulcanians, Klingons, etc.—seek to control the galaxy) and *nova* (simulated navigation among millions of different stars and solar systems, all of which may be revisited, all of which are different...).

People who only play PLATO games occasionally have to sign on by typing their names into the big board. (They often get slaughtered by the regulars.) The regulars—hah. When they're signed into the system, they have merely to jump to a specific game for their fightin' names to be posted on the big board. A mighty rollcall they make, too—such great warriors as von Dave, zot, fright pilot, AL 9000, simpson, doc, THE RED BARON, The Red Sweater, The Giant Pud, Fodzilla, tigress, enema salad, Conan, Siddhartha, wonder pig!!!!, and EXORCIST.

(As those insiders who have automatic sign-on to Big Boards write *programs* to do the sign-on, their arrival in a Big Board game is often an *animated* sign-on. The cutest trick is THE RED BARON's; it looks like this:

THE RED BARON 🐉 (plane falling in flames)

It works like this. For *dogfight*, the terminal already has stored in its temporary memory, as "characters," the little pictures of airplanes that are going to buzz around the screen. So the Baron just follows his name with the code for that special character.)

One last point. No longer can you sign on with an obscenity: a little obscenity-checking program looks for the usual expletives, in case visitors or other priggish folk might be looking. But of course this is easy to circumvent by putting periods between the letters of your nasty word, or something similarly deceptive to the poor program.

### FANTASY ROLE-PLAYING GAMES

A remarkable type of game is the fantasy role-playing game; best known is the

"Dungeons and Dragons" series of Gary Gygax, sold by TSR in Lake Geneva, WI.

These games, typically played by middle-class American males of age 13 to 35, take place in a story-telling setting, with a map and a storyteller/game director called the "dungeonmaster." Each player owns one or more "characters." The characters have certain powers and possessions—like magic swords, magic spells, money and good looks—which are gained and lost as play progresses.

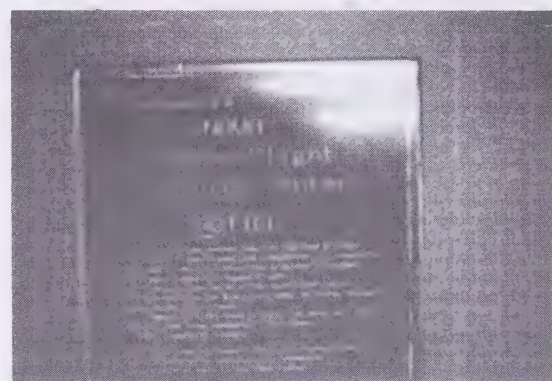
As the characters move about the board, which may be partly or wholly in-

visible, they encounter situations of danger and opportunity which are told, like stories, by the dungeonmaster. Each character separately decides, in the situation of each turn, whether to fight, flee, cast spells, climb, jump, etc. The dungeonmaster, by rolls of complex dice and surreptitious personal decision, determines the outcome.

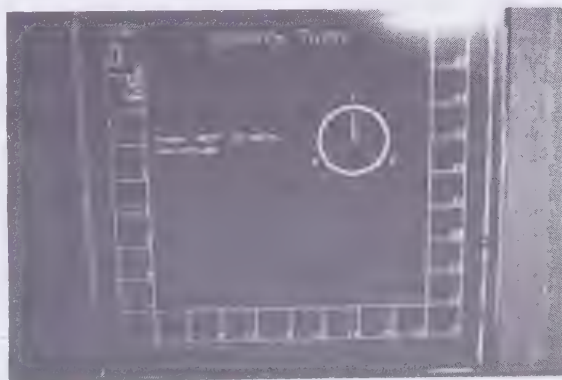
The players feel as one with their characters. The characters' victories, close calls and deaths are deeply felt by the real person. 🐉



Welcome to the Hop Game. PLATO often uses animated opening titles.



The Navigation part of nova is already working. To get around you need instruction; here we are at the Training Center.



Here it is Bunny's turn. Screen instructs you personally: "Press-NEXT-to spin, ermintrude".



View from your Nova spaceship includes perspective view of where you are among billions of stars; and your various controls.



☛ These experiences are powerful and gripping, far more intense than anything else suburbia has to offer. Suicides have been reported over these games. Allegations of devil-worship have been heard. But there's nothing magical about the games themselves, really; these are just the only outlets for **companionate fantasy** available today.

There have been various one-person interactive games based on the fantasy role-playing games, but next to the **real** fantasy role-playing games they are as nothing. The real power of the fantasy role games is in the companionship, the crises together and the imagining together. Replacing the dungeonmaster with a machine throws it away.

### COMPUTERIZED ADVENTURE GAMES

Actually, one mechanized variant of adventure gaming, with a certain limited

appeal, has been very successful.

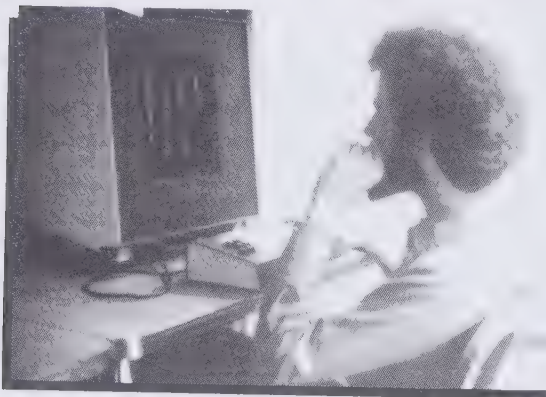
The text adventure game, beginning with what is now called *Original Adventure* at MIT (now available from Software Arts, as part of an album called *Les Crane's Golden Oldies*), began by telling you, textually, that you were in a strange land beside a stream. You then had to experiment with typed commands, such as go north and south, knock on doors, pick things up, eat, fight, and so on. Some people, especially programmer types, seem to find this exhaustive exploration an exciting challenge (sometimes nothing happens until you knock on something a **THIRD** time); others find it a bureaucratic bore.

There are many such adventure games on the market, some with illustrations. *ZORK* is the most popular. I think the market is limited to very specific personality types, but I don't know that anyone has formally researched this aspect.

Student programmer Alan McNeil, an art major, ponders something or other. It may be the program for the Nova space-game he and Pete Rowell are building.

Alan also made a film showing what may have been the motions of the continents, shooting straight off the PLATO screen.

Some PLATO purists point out that this is not exactly what PLATO was originally intended for. So?



**WHERE ARE THEY NOW?**  
Alan created the hit arcade video game "Berzerk." He programs in FORTH, and (last I heard) lives in the country.

## TOMORROW'S SYSTEMS

Those who have happened recently into the computer world may see today's screens and think, *How puny!* Computer screens show too little to be of real use.

That is true today, friend, **today**.

For instance, the standard screen of the IBM PC and its clones is 80 characters wide, 25 lines high. The standard screen of its yuppie-craftsy rival, the Macintosh, is 800 dots wide. Each of these is only about half a sheet of paper, and you can turn pages by hand much faster than flipping documents on their screens.

"Too limited," you may say.

But here a little historical perspective is called for. Just ten years ago personal computers had **no** screens, and then we were able to buy screens 24 characters wide, and **that** was a revolution. And then eighty characters wide, which is the present size.

But the next step up is ready. The hotter screens, those of the "professional workstations" (computers costing, say, \$10,000 and up), go to 132 characters wide, and that can probably double in the next five or six years. (And Mac screens are now available with a million dots; and **that** number will be multiplied soon.

How much can eventually be put on the screen of a CRT we still don't know. But

it's not advisable to bet on an upper limit (see p. DM 85).

How fast the screen can **change** is another matter; it has to do with the speed of memories and processors, also getting faster.

Tom DeFanti (see p. DM 96) has given us an excellent watchword for all this, inspired by Wizzywig (see p. DM 28). His motto is **WIGGYPOOF**, the way he acronymizes his motto: "What You Get Is What You Pay For." And that's the real issue.

Every year or so, more powerful computers and screens come into the Professional market, and the formerly "professional" screens come into the Personal market. Pricewise. So we can plan on getting this year's best as hand-me-downs a few years from now.

### NO UPPER LIMIT

We are safest betting that there is no upper limit on how good screens and computers will get. When will the screens rival 70mm Dolby? 1995? 2005? Stick around.

Then there's the question of sheer processing power, raw crunch.

A class of machines they now call "Crayolas"—miniature Craylike computers (see p. 90)—will soon heave toward the personal price range. As may

## Super TV As A Possible Standard

*The high-resolution TV systems now in their experimental phases may offer the perfect resolution (pun intended) between text, sound, animation and playstation (see p. DM 23) monitor. The hi-res TV can in principle be broken into windows, and could combine text, computer graphics and video feed into a simple composite, especially with something like the DVI chips managing the video (see p. DM 99). This could be an ideal setting for many of the possibilities hoped for in this book.*

See "Technically Better TV," p. DM 59.

such parallel machines as the Transputer (see p. 99), or even the extraordinary Connection Machine of Danny Hillis, with the power of thousands of regular computers all at once.

## WINDOWS

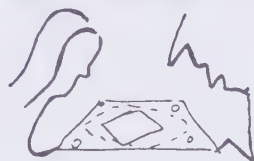
But there will also be stylistic changes. Software for dividing the screen into "windows" will come soon: Microsoft Windows and Quarterdeck Desq for the PC, and competing windowing software for the professional workstations—vying presently are Xwindows, Sun windows, PostScript windows. (But see also "Qframes," p. DM 49.)

## FLIGHT-SIMULATOR-CLASS MACHINES

However, I think that what is really coming for the personal market is another kind of interactive screen. The "flight simulator," of a class with the Evans and Sutherland and IRIS machines (see pp. DM 110, DM 112-113, DM 24, DM 113), but at breakthrough prices.

Given the ever-falling prices of current chips, a thousand-dollar IRIS-type machine might be with us in 1992, perhaps even sooner. And *that* will make a whole new world. For we must redesign education at once toward the time that every pupil can have one, attached to a common feeder network. (One avenue toward that cheap flight-simulator-class machine may be RCA's DVI chip; see p. DM 97). ■

## Quadrapong,



a swell video game now in bars, probably controls the four-player ping pong on the screen with a minicomputer or microprocessor.

Especially exciting is the social possibility of horizontal screens for other fun interpersonal stuff. As well as collaborative work. (But boy, let's hope the radiation shielding is good.)

*There are various Interactive Graphics Standards being worked out by professional committees, standards with names like PHIGS and GKS, but such standards are usually decided by the marketplace. (There are de facto graphics standards in the personal computer market: on the PC it's write-to-screen on the color graphics adapter, and on the Macintosh it's called PICT format.)*

A monitor, or operating system, is a master program that always runs a computer, even when other programs appear to be in control. (Compare the monitor to a stage manager who controls the curtain, while the audience sees only the comedian onstage. See p. 99.)

Our conventional monitor programs have assumed that a user is in charge of the computer and makes things happen by selecting programs to run. Thus the programs start when the user demands, and no outside events need to be synchronized.

Tomorrow's screen machines, however, will be playing in a new fashion: they will combine visuals and animations created inside the user's computer with visuals and animations arriving from various forms of storage or other feed. All these need to be synchronized to make one overall, coordinated presentation. Not only that, but data structures and new dimensions and details of data will have to be added and removed during smooth new animated performances for the user. (The media designer will synchronize these with a symphony-style notation.)

Flat-panel displays, which can in principle be made much larger than CRTs, may yet be perfected.

## Chaos!

### The Local Area Network Situation

So you want several connected computers that are able to share files and printers; that brings us to the problem of

### LOCAL AREA NETS

As soon as you have two computers, you find you have to keep swapping disks all the time. Or putting serial cables between them and writing messy programs to transfer stuff.

As with most problems, there are many different ways to do it. And as with computers, the best advice is GO WITH THE STANDARD. Except there isn't any.

It's chaos out there.

Datapoint's ARCNET, first and best of the lot, blew the chance to be standard because Datapoint would not explain what it actually **was**. (I remember trying to find out at the NCC booth when it came out in the late 70s. The salesman acted as if he wasn't telling, but it was fairly obvious that he actually didn't know.) Xerox' Ethernet is expensive and has a lot of problems. General Motors, Intel and AT&T each offer their own, and IBM has announced at least three.)

However, in typically ibmish fashion, IBM has basically settled on something called the "token ring" network, thought by many to be technically the worst imaginable.

"One ring to rule them all,  
one ring to bind them  
In the land of Mordor  
where the shadows lie."

—J.R.R. Tolkien, Lord of the Rings

The same for sound. Sound and music created inside the user's computer will have to be synchronized with sound coming from something outside the computer.

I take this to be a new and necessary kind of software environment for tomorrow's educational and office systems. Just to give it a name, I call it the Playstation monitor. We need it now.

If you want an overall conjecture on the personal computer system of the future, here goes:

A flight-simulator-class graphical computer with 3D zooming and maneuvering display and real-time 3D image synthesis (see p. DM 111)—the IRIS will do (see pp. DM 24, DM 113); running a Playstation monitor for various real-time feeds with on-board sound and video and data structure swapping; symphonically coordinated by media designers; controlled by a user through controls such as the Walking Net™ and ZigZag™ systems, which are almost instantly learnable (see pp. DM 51, DM 47); and fed from a xanalogical network (see p. DM 144) permitting the linkage of all data structures.



## WEARABLE COMPUTERS

It is still traditional for computers and their display monitors to clutter up desktops (though smaller "lap portables" have established a market position). Very well: what about simply **wearing** our computers?

There is nothing silly about this: **pilots** wear **their** displays, which superimpose upon the visible world animated schematic views of obstacles, targets, enemy aircraft, navigational zones; these helmets adapt the view to the pilot's head position, making it seem as if these glowing features are part of the outside world (see p. DM 91 and picture below).

Okay, yesterday that was luxury; but tomorrow it's **ours**, us'n's, the people's. And for prices in the thousands we'll be able to read and write on faceplates that are always before our eyes, and type on

one-handed keypads, or perhaps on suspended waist-level desktops like the trays of the 1930s cigarette girl.

Not for everybody. But for those of us deep in the media-intensive life, wedded to sweeping search and analysis of things to be known, this is the only direction to pursue.

### TOGETHERNESS

This brings us to the matter of sharing ideas and work.

Most screen systems are currently set up for one person, but we need much more. We need the ability to call forth text, numbers, visualizations, and show all of them instantly to others (and not just today's punk business graphics, but the **real** stuff, animated and in 3D).

What will this be like?

I prefer to see it as a sharable cloud above each person's head as we talk.

This image conveys the **spirit** of the idea. Whether we will have projection gear pointing out of our helmets, or perhaps by then some sort of Aladdin display that makes an apparition in thin air, I don't know; but we will have **something** we can visually share, to replace the blackboard and the paper napkin.

One further thought. The displays we share will have to have two sections: the

shared section, which we show others, and the private section, in which we check and assemble what we are about to show. Think of it as the difference between what we actually **say**, and what goes through our minds before we say it.

But the most important thing is to convey much more of our thinking than we could before.

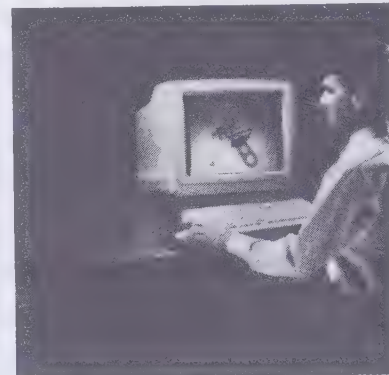
### You saw it here first:

#### THE PERSONAL COMPUTER OF 1993 OR SO

**will have a flight-simulator-class display screen that will show (and swoop amongst) simulated airplanes, landscapes and mountains and clouds, buildings, people, comic characters, mighty molecules and whatever other shapes your program can imagine. Fast, in 3D perspective, among thousands of solid-looking objects. As on this IRIS; see p. DM 112-113. (Similar things can be done in software on today's popular personal computers, but only slowly, and with only a few objects on the screen at once.)**

**You can buy an IRIS right now, as a fine personal computer, but for fifty grand or so. Sometime in the 1990s a comparable system should get down to a thousand dollars.**

Photo courtesy of Silicon Graphics Computer Systems.



### CONVERSATIONS OF THE FUTURE with MINIAIR DISPLAYS



**Porta-Xan. (Mockup by Tom Barnard.) Faceplate reflects CRT to user while he's abroad in the world. One-hand typewriting and pointing device frees the other. Can be built with available ruggedized components. See Xanadu, p. DM 141.**



Artist's impression of the original wearable CRT, Ivan Sutherland's responding stereo helmet and helmet-world (described on p. DM 91).

# PROBLEMS OF SOFTWARE

In the previous section, we considered the lyrical dream that home and office systems could be easier. And in fact this dream has come partly true, sometimes and some places. But it is not generally true as yet.

**USING A COMPUTER SHOULD ALWAYS BE EASIER THAN NOT USING A COMPUTER.** If it isn't, you (or your company, or your state) may have been sold a bill of goods.

OR, they may have decided your inconvenience is less important than something else.

In any case, you have a right to ask sharp questions.

## GETTING TRAPPED IN THE MENU, LOSING THE CURSOR, AND OTHER PROBLEMS OF INTERACTIVE LIFE

Just because you've gotten a program up and running doesn't mean you're out of the woods. Just as you're writing on a roll, or just when you're up against a report deadline (typical points at which software locks up) things can stall, freeze, suddenly foul up. Suggesting that programs are telepathic in the same way delicate equipment is.

You can get trapped in the menu, meaning you've selected what you

wanted, or decided you didn't want anything thanks, and want to get back to what you were doing please, and somehow NOTHING IS HAPPENING.

Or for some reason the cursor has disappeared, or shrivelled to some new symbol, sitting there surly and unresponding.

**YOU CAN'T TELL A BUG FROM A FEATURE.** Have you just stumbled on some glaring flaw that would shame the programmer? Or is the problem that you don't understand the Inner Logic, the thought behind it all, the secret meaning? It of course makes PERFECT SENSE to the guy that designed the program, or the uncomplaining clerk who's been using it for six months, but how in the world were you supposed to know?

In a popular word processing program from a popular Seattle manufacturer, there is a section called the "Gallery," where they have stuck features they couldn't fit anywhere else. I was never

able successfully to get out of the Gallery, even with extensive reference to the manual, and had to reboot the system each time. I now use PC-Write, also made in Seattle.

## THE JAMES JOYCE PRINCIPLE

Opaque wordsmith James Joyce is quoted as saying: "to understand my work you must devote your life to it."

Of course, any artist or writer can make what demands he or she wishes upon the world. The world in turn need pay no attention.

But the software designer is a different kind of artist, uniquely empowered to imprison the victims. Not only can the designer pinion the user in a painful cage, but users have had to pay for the privilege in time, money and effort.

**LIFE IS SHORT, AND WE HAVE MANY OTHER COMMITMENTS.** To be

Some of the worst possible designs, though they may have been whomped up in a hurry by back-room grunts, are often put forward as if they had extraordinary virtues. That's the art of selling.

*Motto of Electronic Arts, a software company:*

*"Software should be simple, hot and deep."*

**SIMPLE:** the user can get into it easily.

**HOT:** it should be excitingly interactive.

**DEEP:** you'll be able to use it for years; it will have "new folds" to discover, and thus a long shelf life.

## FEATURITIS, A Disease of Software

"Featuritis" is what happens when programmers keep adding features (instead of letting them be ramifications of deeper principles).

If you keep adding unrelated features, you have to give them all separate control keys. But the keyboard is only so big.

This book has featuritis, but the different thingies tacked into it don't make it harder to use or get around in. Which is the problem in software.

HELP messages, when not seeming crudely translated from a foreign language, generally repeat some phrase from the manual telling you what you already know (instead of what you want to know) about the specifics of what you're looking at rather than how it relates to anything else, which is usually what you need to know.

The problem is not software "friendliness." It is conceptual clarity. A globe does not say "good morning." It is simple and clear, not "friendly."

## CACOPHONES: The New Frontier Of Bad Design

*There are all kinds of things that telephones can be made to do: include other parties in the conversation; hold your calls; transfer your calls elsewhere; switch to answering-machine mode; record dictation; and so on.*

*The problem is how to make it clear and sensible.*

*Engineers love to design the new controls, and time after time the result is something incomprehensible.*

*One firm had a training film on how to use its telephone system, intended for the customer's secretaries (the ultimate users), with a nice lady explaining the key-sequences. But that did not impart any sense or reason to the design.*

*See "ESS," p. 92*



able to understand, appreciate and use things in a short time-frame is increasingly important.

Your new car and your new camera require half an hour or so before you can use them comfortably.

Software ought to be even less effort. Good software should **free** our time, not encumber it. There is no longer any excuse for not explaining things clearly, and there is no longer any excuse for creating computer systems that cannot be explained clearly. And indeed the artistry of software design is to make that software

as clear and easy as possible.

Software is not made to be casually intelligible, let alone understood by flustered, frantic or distracted users.

Probably no one makes computer systems complicated on purpose. But the problem is that everybody's idea of how a computer system should work is different, and each of these ideas spins out its own web of ramifications. Each creates a world—a virtuality (see p. DM 68-74) with its own system of orientation—of complication, confusion, and mystery if you don't understand it.

## Tuttlefields

There was a legendary maintenance person, some say at Dartmouth, named Tuttle (for whom perhaps the character Tuttle, the fugitive maintenance man in Terry Gilliam's film "Brazil," was named), who supposedly formulated the following Conservation Law:

Tuttle's Law.  
THE AMOUNT OF FUNCTIONING EQUIPMENT IN THE  
UNIVERSE IS A CONSTANT.

This has grim ramifications. (Note that it is also closely related to the doctrine of the Conservation of Immortal Souls, present in some theologies.)

1. The more equipment you own, the less will work.
2. If you fix one thing, another breaks.
3. If one thing breaks, another mysteriously resumes working.

Now, why is it that if your equipment breaks, it is equipment nearby that gets better, rather than equipment on Alpha Centauri?

That is because Tuttle's Law is local as well as global: LOCAL SUBSETS OF TUTTLEFIELDS ALSO CONSERVE BROKENNESS.

Thus research has been urged to ascertain the value of the Tuttle radius, that distance governing what collection of equipment is Tuttlebound.

"WordStar is  
the second  
hardest video  
game ever  
made."

—Anon.

INCOMPREHENSIBLE SOFTWARE  
NOW MAKES IT DIFFICULT OR IM-  
POSSIBLE TO USE—

Alarm clocks,  
answering machines,  
cameras,  
car heaters,  
tape recorders,  
typewriters,  
telephones,  
VCRs,  
wristwatches.

When will they learn? (The de-  
signers, not the gadgets.)

## THE MYTH OF FUNCTIONALITY: IT'S THE INTANGIBLES THAT MATTER

Laymen get so confused when they go to buy computer stuff. They concentrate on all the wrong things: the tangible hardware and its intricate specifications, rather than the all-important **intangibles**, what they really want to do and what style they want to do it in.

People talk as if functionality—what something does—is generic, as if all the programs that do a certain thing are alike, and as if their relative style or ease of their use is of little concern. (Even megapundit Adam Osborne has asserted this.) I call it the myth of functionality.

### THE MYTH OF FUNCTIONALITY FOR LAYMEN

In the movie **Police Academy**, a rookie cop is asked to get a cat out of a tree. He shoots it.

Moral: something can fit the description and not be what you want.

Same for programs. "What a program does"—according to some official description—may be less important than **how it does it**.

Someone who wants the relatively simple text editing of Bank Street Writer, for instance, should not be inflicted with the tangles of Wordstar. Yet many people talk

as if programs of a given type are all alike. They're not.

STYLE, EASE, CLARITY, COMPATIBILITY, AND HAVING IT YOUR WAY—these are what matter.

Unfortunately the myth of functionality is even worse among the technical guys.

### THE MYTH OF FUNCTIONALITY FOR TECHNOIDS

Engineers and other technoids think "software" is **engineering**, and mess up totally. They make a list of the parts they think are needed, program up the separate items, and consider it all done. Thinking that just to make the program's **functions** work is enough.

But software is art, and what counts now most of all are the effects on the user's mind: conceptual **clarity** (especially) for office systems, **feelings** (especially) for games and artworks. And making component pieces that work is only the middle part. The virtuality of the whole is vital.

What we have to do first is **design the ideas** of the program, and its **presentations and interactions** (see p. DM 69). Then we're ready to make the individual pieces that work. And then come the smoothing and integration, as we work the pieces into a consistent whole whose effects all operate together clearly.

What so many people wanted a computer to do is still possible.

Modern life is a mess of accounts and mandated record-keeping. To be middle-class means you have to be an office. Right now, computers are as likely to bring complication as to clear anything up. But within a few years you'll see a turnkey software package that we may call a

### LIFE MANAGEMENT SYSTEM, or LMS

It will be a single unified program, utterly clear, with sprawling octopus routines that easily handle financial planning, checkbook, taxes, bills, insurance, subscriptions, warranties, routine auto maintenance, purchasing, typical correspondence sequences, correspondence filing and summaries, scheduling (with PERT as needed), and tickler tracking of all that must be carried out.

It will have various life sub-scenarios (traffic accident, incorrect billing, wedding) on call through a network, so as to respond effortlessly in those directions as needed, bringing in appropriate form letters, pointing out parts in the letters you are most likely to want to change, and keeping track of when you sent them. All the typing you'll have to do (in these stereotyped sequences) is input of your transactions, changes to form letters, and control selections.

Most importantly, the record-keeping will be fully integrated, with all the relationships maintained. And you'll be able to reconfigure the system quickly and restyle its interaction.

All this without the Fifth Generation or Artificial Intelligence. But it will go far beyond today's integrated software, which is still concerned with producing general, compact tools and fitting in core.

# BEYOND PAPER

*It's very amusing. People sit at computer screens and think they're working with paper. This paper paradigm, this paperdigm, permeates today's computer world. But the notion of paper—a two-dimensional, sequential relation of facts and ideas—is one of the things most holding back software design and human progress.*

*Today's computer systems simulate paper in stupefying detail, enshrining it as if tomorrow would never come, as if today's frozen methods and concepts were permanent.*

*Paper is a crutch, an old-fashioned idea that is holding us back. When screen methods get good enough—and they are far from it now—there will be no more hard copy. And it is toward that day we must design—toward finer, more flexible, all-seeing methods that we could never have in these last paper days.*

*Eventually we must drop the ancient ways and frozen methods, we must throw away this security blanket of paper, we must let go of the side of the pool, and find those new virtualities of tomorrow's hypermedia that will free our minds and unchain our work.*

## TYPE RIGHTER: The Magic Typewriters

The Moving Finger writes; and, having writ,

Moves on: nor all your Piety nor Wit  
Shall lure it back to cancel half a Line,  
Nor all your Tears wash out a Word of it.

—Khayyam/Fitzgerald

A number of different systems are coming on the market to aid you in error-free typing.

IBM would have you call these “word processing systems,” since that makes them sound of-a-piece with their dictation equipment. Actually they're text regurgitation systems, but let's just call them Magic Typewriters.

Prices of these things tend to run between \$100 and \$250 a month.

Generally these are being sold as secretarial aids, partly because they tend to be too ungainly for use by writers themselves. A principal use has been in large law offices, where contracts, wills and such are stored as “boilerplate” (standard sections of Document) and then modified slightly by the lawyer to justify the legal fees.

Such systems all basically consist of three things:

A typewriter, connected to some sort of *magnetic memory*, such as a tape, coated card or disk, and *editing circuitry*, which responds to various acts by the user.

**WHAT THEY DO:** allow you to type stuff in, which is both typed on the paper and at the same time *stored* on the magnetic whatever. Small errors you correct as you type along, generally by backspacing.

When you want a clean copy—Presto Wait-o! Put in clean paper, start the magnetic whatever at the beginning, and the typewriter retypes it without a mistake.

If you're lucky.

Unfortunately some of these systems are quite badly thought out. In one or two cases I am not sure whether they are designed as they are *accidentally* or *on purpose*. Neither interpretation is flattering to the manufacturer.

I have had extensive experience with two of these systems, the IBM Mag Tape Selectric and the IBM Mag Card Executive. Suffice it to say that if I believed that these systems were as cumbersome as they are by accident, then the sections in this book on IBM and its products might have a very different slant. As it is, these systems require a training period of (say) a week, and require such continuous at-

tention to their curious mechanics that the user is given little opportunity to think of anything else. In both cases, in my opinion, the superficial plausibility of the initial design premises knots into tangled ramifications which verge on the preposterous. Much of this book was written on a Mag Card Executive—and I'm damned sorry I bothered.

## THINGS COMPUTERS CURRENTLY DO WITH TEXT

*“Word processing” is an odd term because it refers generically to handling text by computer as though it were a well-defined bundle of activity.*

*The term as usually used means just typing stuff in and getting it out again, typed on paper. Thus there are various conventions for arranging it on paper. There are also various conventions for what happens on the screen—the cursor moves as if between Scrabble tiles; there is no recognition by the computer of words or sentences.*

*Now, “word processors” are only expected to arrange text very simply on the page; if they can handle multiple columns, or make space for illustrations, they're called **desktop publishing programs**. On the other hand, if the program allows you a shrunk view, showing only the beginnings of things, then it's an **outline processor**. These distinctions are idi-*

*Present-day iconic office systems represent different machines and activities with little paper-related pictures. This line of reasoning suggests that to prepare an electronic letter you should see a picture of a letter slide into a picture of an envelope. Then, to send it, you should lick the screen.*

IBM has come out with a “Document Content Architecture.” Sounds imposing. Turns out it's an intercommunication standard for tying together all their word-processing products. Adds complication and won't even support all features.

There are still no serious computer aids to structure and organization: “outline processing” only deals with sequence, and that is still within the paperdigm. Collateral linked viewing of alternative versions is an example of a viewing-structure that cuts much deeper. (See p. DM 50.)

## A Nice Distinction

“Prose” is text which may be reformatted automatically with no loss of value—i.e., sentences and paragraphs.

“Poetry” is text whose format may not be changed automatically. This includes title pages, spreadsheets and so on, as well as actual poetry.

Most text systems today are built around

**THE VPOP**  
(Virtual Piece of Paper)

and shower you with

**COMPLOP**  
(big piles of printout).

JOKE  
(requires filling in name of ill-regarded person)  
Q. How do you know if  
has been using the word processor?  
A. By the white-out on the screen.



## Now It Can Be Told: *The Dataflash Project*

In 1981 and 1982, Datapoint Corporation's Advanced Products Division set out to investigate high-power text systems. Initially called the Conehead project, it was later officially christened Dataflash. On the project were alumni from Project Xanadu (the author, Mark S. Miller and Stuart Greene), as well as Datapoint veteran Kim Riley. In charge was Klavs Landberg, who has given permission to print this.

We wanted a fast text system that would handle very large documents with high performance, permit links between documents, and automatically format both screen and printout in columns and around graphics.

The data structure we chose was from my 1965 ACM paper (citation p. DM 33), which breaks everything into zippered lists or zips (sequential strips of data which can link sideways to other zips). In the Dataflash versions these were a cumulative zip (ZQ) of all characters typed in, in chronological order, and a control zip (ZK) which pointed to the pieces that made up a given document. Links were to be between spans of ZK in different documents. (This data structure had actually been dropped by the Xanadu project in 1970.)

A format zip (ZF), pointing to ZK, was a precooked list of line-lengths and starting points on the screen.

This permitted really neat things. In addition to the obvious editing features (all editing was done to ZK), you could scroll through text on the screen at seeming lightning speed, watching paragraphs and indentations whiz by much faster than the actual contents could be retrieved. This was done by fast scrolling of the screen format pointers, ZF, mapped to the screen as random characters. It looked like very fast scrolling (we called it "Thunderscroll"). And when you wanted to stop, the correct characters were to be thrown on the screen in random order to give the impression of coming into focus, rather than filling the screen sequentially: it took the same length of time but looked niftier. This was called "Popcorn." A rapid jump into a document was to be shown by a zooming trapezoidal effect. This was "Hyperhop."

The Thunderscroll-Hyperhop-Popcorn visuals were implemented in a demo version by Steve Witham. We called this "Thunder Hop'n'Pop." It looked great. This was the high point of the project.

I put such matters under the heading of screen theatrics and hold them to be vitally important. (Klavs used the more dubious term "operator entertainment.")

Printed output formatting was to be handled by a system involving various pourable formats (which we called "pourmats"); the text would play through invisible boxes constrained on the page by invisible rubber straps.

ANYway, Datapoint decided to take a more evolutionary approach, upgrading the system they already had.

**Early  
horseless  
carriages  
were built  
with a  
buggy-whip  
socket.  
Today's  
simulated  
paper is  
comparable.**

**"The  
paperless  
office is as  
much a myth  
as the  
paperless  
toilet."**

**—Industry  
Pundit**

**Anyone who  
equates an  
office with a  
toilet should  
not be  
designing  
software.**

otic. All these legitimate functions should be packaged together, even **within** the paperdigm.

It's also sad and silly that people regard these activities—"word processing," "outline processing," "desktop publishing"—as well-defined and well-bounded forms of endeavor, whereas they're just conventional packaging for things that shouldn't be separate, and had no reason to be separated in the first place. (And weren't, in Engelbart's original NLS; see p. DM 16.)

But the worst aspect of this premature and insane division of function is that it enshrines **paper** as the destination, the final repository of what is being worked on. And this is the insidious and retrograde aspect of such systems.

### **The Grotesque Prudery of WIZZYWIG**

"Wizzywig" is how you pronounce the popular acronym WYSIWYG, which stands for What You See Is What You Get—now a retrograde and unfortunate doctrine that holds us back.

A lot of early text programs had ugly and stupid special characters and codes on the screen. Programmers could deal with these, secretaries and writers could not.

However, at places like Brown University and Xerox PARC, they began to put text on screens in ways that more resembled a final printed output.

To the best of my knowledge, the first visual word processor that was easy for beginners (using menus and the like) was actually something called "The Brown University Hypertext System" (see p. DM 35), which I was involved with in the sixties. There were people putting two different directions into its design: those who wanted to emphasize printout features for conventional paper text, and those who wanted to put the main resources on techniques for leaving paper behind (actually only the author).

The result: a basic hypertext system that showed people what screen editing for paper printout should be like. To the best of my knowledge most of the present generation of word-processing programs derived from the Brown System—but lost the hypertext part.

And people took these to heart, and called it Wizzywig, a term that became popular around 1980.

But what is sinister about the term "Wizzywig" is its **homage to paper**. In the

all-screen worlds that are coming, what you **get** is what you get: paper will be irrelevant, hard copy will be forgotten, the screen will be all there need be. (Today you still need hard copy because interaction and viewing methods are so bad.)

Not only that, but Wizzywig is also advocated by those who think **there should be only one view of a thing**. "What you see" is all you see and can't be any other way (because then it would no longer be what you "get"! ). Other views are not allowed; uptight people think they would be **confusing**.

And this is the great mistake.

### **THE LOWERED VISION**

Perhaps the most tragic aspect of software today is its shallowness. Because the Mundanes have called us "crazy" and "Buck Rogers," computer people have answered by abandoning the most important possibilities of the interactive computer screen. People are **simulating paper** on the screen, putting out software that caters to the yokel mentality, and castrating the real possibilities of high interaction. Designers are sticking to systems where printout is imagined to be the final proper version—and even these systems rarely get out of first gear.

The paperdigm is absurd. Screens should set us **free** from the dimensions and topology of paper; we should find truer visualizations of the conceptual structure of what we do; and we should have many views, being able to switch quickly among them.

Computer screens offer us unprecedented variety and flexibility in looking at, and understanding, **everything**; but only if they can bring us, rapidly and interactively and comprehensively, hundreds of different ways of seeing things. To resist seeing things in different ways is to keep your head in a bucket.

Most people just **don't get** that our screen worlds of tomorrow will be paperless. They cannot walk out of the paperdigm, lift their eyes from the paperscape, imagine a life without hard copy.

In the sixties I found it very frustrating to try to tell people what it would be like to read and write on screens, to draw on screens.

I still feel the same frustration in trying to explain about tomorrow's **real** text systems. Not to mention multidimensional screen graphics.

# !GREBNETUG

Now, in our time, we are turning Gutenberg around. The technology of movable type created certain structures and practices around the written word. Now the technology of computer screen displays makes possible almost *any structures and practices you can imagine* for the written word.

So now what?

For new forms of written communication among people who know each other, jump to the "Engelbart" piece, p. DM 16.

To learn about new forms of multi-dimensional documents for computer screens, jump to "Hypertext," below.

Or just feel free to browse.

## HYPERTEXT

By "hypertext" I mean non-sequential writing.

Ordinary writing is sequential for two reasons. First, it grew out of speech and speech-making, which have to be sequential; and second, because books are not convenient to read except in a sequence.

But the *structures of ideas* are not sequential. They tie together every which-way. And when we write, we are always trying to tie things together in non-sequential ways (see p. DM 29). The *foot-note* is a break from sequence; but it cannot really be extended (though some, like Will Cuppy, have toyed with the technique).

I have run into perhaps a dozen people who understood this instantly when I talked to them about it. Most people, however, act more bemused, thinking I'm trying to tell them something technical or pointlessly philosophical. It's not pointless at all: the point is, writers do better if they don't have to *write* in sequence (but may create multiple structures, branches and alternatives), and readers do better if they don't have to *read* in sequence, but may establish impressions, jump around, and try different pathways until they find the ones they want to study most closely.

(The astute reader, and anybody who's gotten to this point must be, will have noticed that this book is in "magazine" layout, organized visually by ideas and meanings, for that precise reason. I will be interested to hear whether that has worked.)

And the pity of it is that (like the man in the French play who was surprised to learn that he had been "speaking prose all

his life and never known it"), we've been speaking *hypertext* all our lives and never known it.

Now, many writers have *tried* to break away from sequence. I think of Nabokov's *Pale Fire*, of *Tristram Shandy* and an odd novel of Lazaro Cortazar called *Hopscotch*, made up of sections ending with numbers telling you where you can branch to. There are many more; and large books generally use many tricks to get around the problem of indexing and reviewing what has and hasn't been said or done already.

However, in my view, a new day is dawning. Computer storage and screen display mean that we no longer *have* to have things in sequence; totally arbitrary structures are possible, and I think that after we've tried them enough people will see how desirable they are.

*I still don't know why it took so long. I first published the term "hypertext" in 1965, and for the last twenty years have been giving speeches and writing articles about how we need hypertext for education, scholarship, archiving and "the office of the future"—not to mention poetry.*

*But suddenly in 1986 the idea caught on. It may have been the NoteCards program from Xerox—programmed by Frank Halasz and Randy Trygg—released the year before; it may have been Gary Kildall's opening speech at the 1986 CD-ROM conference, where he spoke in praise of hypertext; it may have been the appearance of the GUIDE hypertext program from Owl in Seattle. And a "hypertext abstract machine" (HAM) is being developed at Tektronix.*

I believe in calling a spade a spade—not a personalized earth-moving equipment module; and a multi-dimensional spade, by gum, a hyperspade—not a personalized earth-moving equipment module with augmented dirt access, retrieval and display capability under individualized control.



**More on the word-picture continuum:**

**CONFUCIUS WAS RIGHT**

**A thousand words comes out to a little under 10K of computer storage. So does a modest sort of picture.**

## Procrustes the Giant

The Greeks told of a giant named Procrustes (rhymes with Rusty's) who was very hospitable to passing travelers. He would invite, indeed compel, them to sleep in his bed. Unfortunately, because it was a very odd bed, he had to cut them up first...

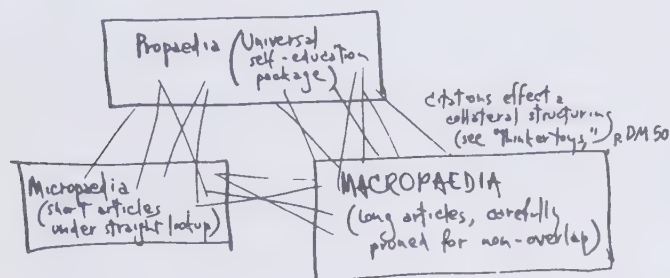
Procrustes has haunted conversations ever since; and any time we are forced to use categories that don't properly fit a subject, it seems like an invitation to the Procrustean bed.

Hypertext systems at last offer total freedom from arbitrary categorizing and chopping; but in *some* systems for storing and presenting information, I can't help hearing the whisk of Procrustes' knife—



Mortimer J. Adler, the man who reduced all of Western Culture to a few Great Books plus an index under his own categories, has now Added the *Encyclopaedia Britannica*.

Since 1965 he has been creating *Britannica 3*, the venturesome and innovative new version, now on sale for about half a thou.



*Britannica 3* is basically a 3-level hypertext, made to fit on printed pages by the strictures of Adler's editing (according to *Newsweek*, some 200 authors withdrew their work rather than submit to the kind of restrictions he was imposing).

The idea may be basically good, even though the sesquipaedian titles may impaied the raeder.

## "Interactive Fiction"

Interactive fiction is an interesting new genre, first promulgated (I believe) by Bob Lafore, and later by computer software houses such as Infocom. (There are also "multiple-choice romances," on paper, available at any local grocery.)

The reader of interactive fiction makes various choices as she or he goes along, and the story line thus selected becomes the ongoing reality of the tale.

This is a form of hypertext, since hypertext simply means nonsequential writing. However, the sorts of overviews you want in a non-fiction hypertext might detract from the fun where a story with surprises is involved.

(A program for writing interactive fiction on the Apple II is StoryTree from Scholastic WizWare (I), New York City.)

Anyway, now the idea is everywhere; there are hypertext sessions at various conferences, and hypertext conferences coming up (at last organized by other people).

Lots of different pieces of software could be used to present hypertext of one sort or another; even MacDraw (the reader could point and delete to uncover pieces he or she chose). But dealing with the complexities of the hypertext problem such as alternative versions, is quite another matter, and requires a level of hypertext consciousness most have not yet reached (see "problems of hypertext" in my book *Literary Machines*).

**WHY HYPertext?**  
**WHY INTERACTION?**  
**WHY EXPLORABILITY?**

The notion of sequential presentation is deep in our thinking. The TV show, the movie, even the fireside tale, are sequential. The report to management is sequential.

Yet there are many reasons for wanting to create non-sequential explorable media:

TO REPRESENT THE TRUE STRUCTURE OF THINGS. Your body is

not sequentially interconnected, but neither is everything connected to everything else. There are many specific places and forms and aspects of interconnection, each of which needs to be understood—by doctors if not necessarily by you. Why not have a way to represent, and to present, this true structure in a way that someone can explore it—for learning, for ready reference.

Often there are simultaneous and interpenetrating structures which need to be understood separately and together.

TO FOLLOW PREFERENCES. Some people want to study one connection first, some another. (Or, in fiction, some people want one alternative, some another.)

MOTIVATION. When you are free to explore, rather than being pushed to do something in a certain sequence, you have a far greater sense of autonomy—and much greater motivation.

LESS COST. It is difficult enough to force writings into sequence on simple subjects; but on complex, interpenetrating subjects it is so difficult as to be pointless, when the true structure itself can be represented instead. ☛

## Couldn't Have Hypertext Novels, You Say?

Consider the hypertext character of—

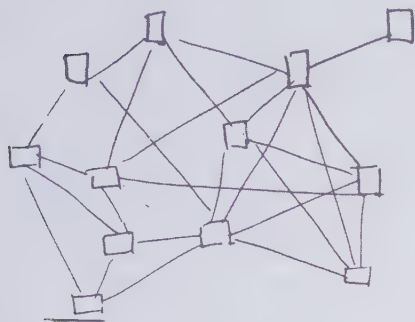
- *Tristram Shandy*, by Sterne.
- *Spoon River Anthology*, by Masters.
- *Hopscotch*, by Cortazar.
- *Pale Fire*, by Nabokov.
- *Remembrance of Things Past*, by Proust.

And, surprisingly, hypertext actually FIGURES IN *Giles Goat-Boy*, by Barth.

Arthur C. Clarke wrote a book entitled *The Lost Worlds of 2001* (Signet, 1972), about the variants and alternatives of that story that did not find their way to the screen.

In a hypertext version, we could look at them all in context, in collateral views, and see the related variants—with annotations.

Hierarchical and sequential structures, especially popular since Gutenberg, are usually forced and artificial. Intertwining is not generally acknowledged—people keep pretending they can make things hierarchical, categorizable and sequential when they can't.



EVERYTHING IS DEEPLY INTERTWINGLED.

In an important sense there are no "subjects" at all; there is only all knowledge, since the cross-connections among the myriad topics of this world simply cannot be divided up neatly.

Hypertext at last offers the possibility of representing and exploring it all without carving it up destructively.

## PRESENTATIONAL SEQUENCES ARE ARBITRARY

BIRDS → BEES → FLOWERS

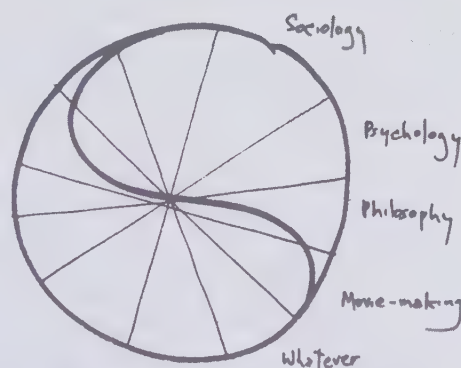
FLOWERS → BIRDS → BEES

FLOWERS → PEOPLE → BIRDS

## HIERARCHIES ARE TYPICALLY SPURIOUS

LANGUAGE  
TRUTH  
LOGIC

GOD  
MAN  
YALE



## BOUNDARIES OF FIELDS ARE ARBITRARY

## COMPARTMENTALIZED AND STRATIFIED TEACHING PRODUCES COMPARTMENTALIZED AND STRATIFIED MINDS.

Hypertext is non-sequential text. If we write a hypertext on something, it will be most appropriate if we give it the general interconnective structure of the field. In other words, the interconnective structures chosen for the textual parts are likely to have the same connective structure.

For another kind of hypertext, the anthological hypertext built up of lots of other writings, it is also reasonable to

expect the connective structures to cluster to the same general form.

In other words, the very same field of knowledge that others represent as an explorable, formalized whole, I am out to represent as an explorable *informalized* whole, with anecdotes, jokes, cartoons, "enrichment materials," and anything else people might dig.

# Hypertext For Learning

A principal point is that the student is in control and may use his initiative dynamically; the subject is *not* artificially processed into a presentational sequence. Moreover, the arbitrary interconnections of the subject, which are no respecters of the printed page, are recognized as the fundamental structures the student must deal with and come to understand.

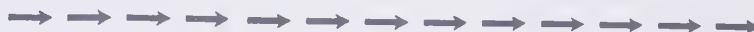
Compare Alice, when she gets to Wonderland

("Deary me! Curiouser and curiouser!")

with Dorothy Gale, transported to Oz

("How do I get back to Kansas?!!!")

*The sense of initiative makes all the difference, and hypertext promotes that.*



In hypertext systems, there is a good reason to make the tools and access privileges of all users the same: the reader's tool can be the same as the author's tool. Thus hypertext may be intrinsically an egalitarian medium. (Provided, especially that pre-existing materials may be freely edited and incorporated in new objects without damaging the old. See "Xanalogical Storage," p. DM 144.) And this is different from the lord/peon model of information exchange.

The hypertext approach does not confer omniscience or omnipotence upon the author. The writing, like all writing, represents a point of view.

As with paper, the author cannot imprison you or make insuperable requirements.

Any nonsequential text, through a hypertext portal, confers on you this freedom.



## TYPES OF HYPERTEXT

Let's assume that you have a high-power display—and storage displays *won't do*, because you have to see things *move* in order to understand where they come from and what they mean. (Especially text.) So it has to be a refreshed CRT.

*Basic or chunk style hypertext* offers choices, either as footnote-markers (like asterisks) or labels at the end of a chunk. Whatever you point at then comes to the screen.

*Collateral hypertext* means compound annotations or parallel text (see p. DM 50).

*Stretchtext* changes *continuously*. This requires very unusual techniques (see p. DM 134), but exemplifies how "continuous" hypertext might work.

Ideally, chunk and continuous and collateral hypertext could all be combined (and in turn collaterally linked; see "Thinkertoys," p. DM 50).

A "fresh" or "specific" hypertext—I don't have a better term at the moment—would consist of material especially written for some purpose. An *anthological* hypertext, however, would consist of materials brought together from all over, like an anthological book.

A *grand hypertext*, then, folks, would be a hypertext consisting of "everything" written about a subject, or vaguely relevant to it, tied together by editors (and NOT by "programmers," dammit), in which you may read *in all the directions you wish to pursue*. There can be alternative pathways for people who think different ways. People who *have* to have one thing explained to them at a time—many have insisted to me that this is normal, although I contend that it is a pathological condition—may have that; others, learning like true human beings, may gather and sift impressions until the ideas become clear.

And then, of course, you see the real dream.

### THE DREAM OF GRAND HYPERTEXT

The real dream is for "everything" to be in the hypertext.

Everything you read, you read from the screen (and can always get back to right away); everything you write, you write at the screen (and can cross-link to whatever you read; see "Canons," p. DM 149).

Paper moulders. Microfilm is inconvenient. In the best libraries it takes at least minutes to get a particular thing. But as to

linking them together—footnoting Aeschylus with Marcus Aurelius, linking genetic data to 15th-century accounts of Indian tribes—well, you can only do it on paper by writing something *new* that ties them together. Isn't that ridiculous? When you could do it all electronically in seconds?

Now that we *have* all these wonderful devices, it should be the goal of society to put them in the service of truth and learning. And this is the way I propose. *Not* through obscure forms of "information retrieval"; *not* through newly oppressive forms of "computer-assisted instruction"; and *not* through a purported science of "artificial intelligence" that will create new personalisms to irk us. All these obstructive oddities, I think, have developed as separate ideals because of the grand preposterousness of Professionalism that has created a world-wide cult of mutual incomprehensibility and disconnected special goals. Now we need to get everybody together again. We want to go back to the roots of our civilization—the ability, which we once had, for everybody who could read to be able to read everything. We *must* once again become a community of common access to a shared heritage.

This was of course what Vannevar Bush said in 1945 (see nearby), in an article everybody cites but nobody reads.

The hypertext solution in many ways obviates some of these other approaches, and in addition retains and puts back together the great traditions of literature and scholarship, traditions based on the fact that dividing things up arbitrarily just generally doesn't work.

### EVERYTHING IS DEEPLY INTERTWINGLED.

(The only way in which my views differ with those of Engelbart and Pask, I think, is in the matter of structure and hierarchy. Both men generally assume that whatever natural hierarchy may exist in particular subjects needs to be accentuated; I hold that all structures must be treated as totally arbitrary, and any hierarchies we find are interesting accidents.)

### CAN IT BE DONE?

I dunno.

Licklider, one of computerdom's Great Men, estimated in 1965 that to handle all text by computer, and bring it out to screens, would cost no more than what we pay for all text handling *now*. (But of course there is the problem of what to do

Weapons manuals are estimated to cost a thousand dollars a page.  
The cost of producing manuals for the B-1 bomber was estimated at 1.2 billion dollars (New York Times, 1 April 1985; no April Fool).  
This is an area where the need for hypertext is felt most insistently; which is why airplane maintenance manuals are already being distributed in hypertext form.

## The Burning Bush

In fact hypertexts were foreseen very clearly in 1945 by Vannevar Bush, Roosevelt's science advisor. When the war was in the bag, he published a little article on various groovy things that had become possible by that time.

"As We May Think" (*Atlantic Monthly*, July 1945) is most notable for its clear description of various hypertext techniques—that is, linkages between documents which may be brought rapidly to the screen according to their linkages. (So what if he thought they'd be on microfilm.)

How characteristic of Professionalism. Bush's article has been taken as the starting point for the field of Information Retrieval (see p. DM 127), but its actual contents have been ignored by acclamation. Information Retrieval folk have mostly done very different things, yet thought they were in the tradition.

Now people are "rediscovering" the article. If there's another edition of this book I hope I can run it in entirety.

*It is reprinted in full in the 1987 edition of Literary Machines.*

with the people whose lives are built around paper; that can't be taken up here.)

The people who make big computers say that to get the big disk storage to hold great amounts of text, you have to get their biggest computers. Which is a laugh and a half. One IBM-style computer person pompously told me that for large-scale text handling the only appropriate machine was an IBM 360/67 (a shamefully large computer). Such people seem not to understand about minicomputers or the potential of minicomputer networks—using, of course, big disks.

There are of course questions of reliability, of “big brother” (see “Canons,” p. DM 149), and so on. But I think these matters can be handled.

The key is that people will pay for it. I am sure that if we can bring the cost down to two dollars an hour—one for the local machine (more than a “terminal”), one for the material (including storage, transmission and copyrights)—there's a big, big market. (And that's what the Xanadu network is about; see p. DM 141.) My assumption is that the way to do this is *not* through big business (since all these corporations can see is other corporations); *not* through government (hypertext is not committee-oriented, but individualis-

tic—and grants can only be gotten through sesquipedalian and obfuscatory pompizzazz); but through the byways of the private enterprise system. I think the same spirit that gave us McDonald's and kandy kolor hot rod accessories may pull us through here. (See “Xanadu Network,” p. DM 141.)

Obviously, putting man's entire heritage into a hypertext is going to take awhile. But it can and should be done.

#### BIBLIOGRAPHY

Theodor H. Nelson, “The Hypertext.” Proc. World Documentation Federation, 1965.

Theodor H. Nelson, “A File Structure for the Complex, the Changing and the Indeterminate,” *Proceedings ACM* 1965.

Theodor H. Nelson, “A New Home for the Mind,” *Datamation*, March 1982.

Theodor H. Nelson, “Replacing the Printed Word.” In S.H. Lavington (ed.), *Information Processing 80*, North Holland Publishing, IFIP 1980.

Theodor H. Nelson, *Literary Machines*. (Citation p. DM 148.)

J.C.R. Licklider, *Libraries of the Future*, (Citation p. DM 128).

## NONSEQUENTIAL WORK PRODUCT A Sociological Problem

Recently, a hypertext project was turned in at a major research institute. The guy who had created the system wanted to write a hypertext to report on the system.

Management demurred. A project results in a paper report, the programmer was told.

BOSSES HATE THIS: when management got the idea that the system was meant to replace paper reports with a whole new way of doing things, it was Not Pleased.

When I have explained to people that hypertext will allow the true uncertainties of thought to be accurately recorded and shown, they have tended not to like this idea. It seems to violate a dearly-held notion of closure and completeness.



The past is like the receding view out the back of an automobile: the most recent is more conspicuous, and everything seems eventually to be lost.

We know we should save things, but what? Those with the job of saving things—the libraries and museums—save so many of the wrong things, the fashionable and expensive and high-toned things esteemed by a given time, and most of the rest slips past. Each generation seems to ridicule the things held in esteem by times before, but of course this can never be a guide to what *should* be saved. And there is so much to save: music, writing, sinking Venice, vanishing species.

But why should things be saved? Everything is deeply intertwined. We save for knowledge and nostalgia, but what we thought was knowledge often turns to nostalgia, and nostalgia

often brings us deeper insights that cut across our lives and very selves.

Computers offer an interesting daydream: that we may be able to store things *digitally* instead of *physically*. In other words, turn the libraries to digital storage; digitize paintings and photographs (see “Picture Processing,” p. DM 81); even digitize the genetic codes of animals, so that species can be restored at future dates (see “The Mitiest Computer,” p. 94).

Digital storage possesses several special advantages. Digitally stored materials may be copied by automatic means; corrective measures are possible, to prevent errors from creeping in—i.e., “no deterioration” in principle; and they could be kept in various places, lessening mankind's dependence on its eggs being all in one basket (like the Library at Alexandria, whose burning was one of the greatest losses in human history). See “Archives,” pp. 166, 167.

But this would of course require far more compact and reliable forms of digital storage than exist right now.

Nevertheless, we better start thinking about it. Those who fear a coming holocaust (see p. 172) had best think about pulling some part of mankind through, with some part of what he used to have.

*The possibility of using a hypertext network as a universal archive is a dramatic development. See p. DM 141.*



# RASHOMONICS AND THE PERPLEX

Uneducated people typically think of education as the learning of a lot of facts and skills. While facts and skills certainly have their merits, "higher education" is also largely concerned with tying ideas together, and especially alternative structures of such tying-together: with showing you the vast uncertainties of things.

A wonderful Japanese film of the fifties was called *Rasho-Mon*. It depicted a specific event—a rape—as told by five different people. As the audience watches the five separate stories, they must try to judge what *really* happened.

The Rasho-Mon Principle: everything is like that. The complete truth about something is never known.

Nobody tells the complete truth, though some try. Nobody knows the complete truth. Nowhere may we find *printed* the complete truth. There are only different views, assertions, supposed facts that support one view or another but are disputed by disbelievers in the particular views; and so on. There are "agreed-on facts," but their *meaning* is often in doubt.

## THE TISSUE OF THOUGHT

The great compromise of the western world is that we go by the rule: *assume* that we never know the final truth about anything. There are continuing Issues, Mysteries, Continuing Dialogues. What about flying saucers, "why Rome fell," was there a Passover Plot, and did Roosevelt know Pearl Harbor would be attacked?

A brilliant piece of academic humor in the mid-sixties was Frederick C. Crews' *The Pooh Perplex*. In it, half a dozen weighty academics propounded different theories of the inner meaning of A.A. Milne's Winnie-The-Pooh, each in the language and *Weltanschauung* of his particular academic specialty: a psychoanalyst, a literary critic, and so on. They were all Crews, of course.

But this book made a contribution to understanding, I think, and that is the word *perplex*. Crews used it to mean the interpenetrating, intertwined whole of the controversy he presented, a precisely

delineated muddle. I think this is an excellent word for it.

In every area of scholarship, we need precise systems of intercomparison that will enable us to study and annotate the perplexes we must deal with.

Outsiders find the intellectual world pompous, vague in its undecided issues, stuffy in its quotes and citations. But in a way these are the sounds of battle. The clash of theories is what many find exhilarating about the intellectual world. The Scholarly Arena is simply a Circus Maximus in which these battles are scheduled.

Many people think "science" is free from all this. These are people who do not know much about science. More and more is scientifically known, true; but it is repeatedly discovered that some scientific "knowledge" is *untrue*, and this problem is built into the system. The important thing about science is not that everything will be known, or that everything unanimously believed by scientists is necessarily true, but that science contains a *system* for seeking untruth and purging it.

This is the great tradition of western civilization. The Western World is, in an important sense, a continuing dialogue among people who have thought different things. "Scholarship" is the tradition of trying to improve, collate and resolve uncertainties. The fundamental ground rules are that no issue is ever closed, no interconnection is impossible. It all comes down to *what is written*, because the thoughts and minds themselves, of course, do not last. (The apparatus of citation and footnote are simply a combination of hat-tipping, go-look-if-you-don't-believe-me, and you-might-want-to-read-this-yourself.)

"Knowledge," then—and indeed most of our civilization and what remains of those previous—is a vast cross-tangle of ideas and evidential materials, not a pyramid of truth. So that preserving its structure, and improving its accessibility, is important to us all.

Which is one reason we need hypertexts and thinkertoys.

WE OFTEN WANT TO SAVE ALTERNATIVES.

28

And closely to my <sup>side</sup> [heart] she press'd,  
And closer still with hushful art,  
[And ask'd me with her swimming eyes]  
That I <sup>might</sup> [would] rather feel than see  
The swelling of her Heave  
[Her gentle Bosom rise.—]

29

[And now serene, serene & chaste,] I calm'd her fears; & she was calm  
[But soon in calm and solemn tone]  
And [She] told her love with maiden pride;  
And so I won my Genevieve,  
My <sup>dear</sup> [bride] & lovely Bride.]

From Coleridge's Poems: A Facsimile  
Reproduction of the Proofs  
and MSS. of some of the Poems.  
(Folcroft, 1972.)

We might also think of thinkertoys as systems  
for THE MANAGEMENT OF LOOSE ENDS.

## GLINDA'S MAGIC BOOK

Glinda the Good, gentle sorceress of the southern quadrant of the land of Oz—not the flaphead portrayed by Billie Burke in the Goldwynized film—has a Magic Book in which Everything That Happens is written.

The question, of course, is how it's chosen.

You can only watch news tickers for a short time before getting very bored.



# A VERY BASIC HYPERTEXT SYSTEM

Hypertext is non-sequential writing. It's no good to us, though, unless we can go instantly in a choice of directions from a given point.

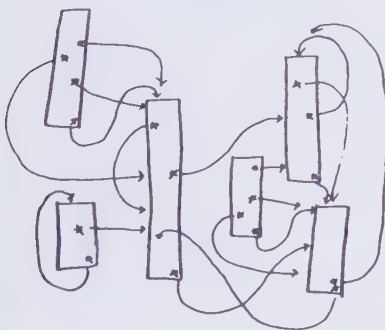
This of course can only mean on computer display screens.

Engelbart's system, now, was mainly designed for people who wanted to immerse themselves in it and learn its conventions. Indeed, it might be said to have been designed for a community of people in close contact, a sort of system of blackboards and collaborative talking papers.

A more elemental system, with a different slant, was put together at Brown U. on IBM equipment. We'll refer to it here as "Carmody's System," after the young programmer whose name came first on the writeup.

Carmody's system runs on an IBM 360 with 2250 display. While the 2250 is a fine piece of equipment, the quirks of the 360's operating system (see p. 102) often delay the user by making him wait, e.g., for someone else's cards to get punched before it responds to his more immediate uses; this is like making ice-skaters wait for ox carts.

Anyway, the system essentially imposes no structure on the material; it may consist of text segments of any length and ties and links between them. An asterisk appearing anywhere in one piece of text signals a possible jump, but the reader doesn't necessarily know where to; zapping the asterisk with the lightpen takes you there, however.



This is stark and simple. It could also get you good and lost. However, a simple

technique took care of that: every time the user jumped, the address of his previous location was saved on a *stack* (see "The Magic of the Stack," p. 78). The user also had a RETURN button: when he wanted to go back to where he had last jumped from, the system would pop the last address off the top of the stack, and take him there. (This feature was adapted from my 1967 Stretchtext paper, and turned out to work out quite well in practice.)

The system also had handy features for light-pen text editing, and various nice printout techniques. All told, it was a clean and powerful design. While it lacked higher-level visualization facilities, like Engelbart's display of Levels (see "outline" in Engelbart article) or collateral display (see "Thinkertoys," p. DM 50), it was in some ways suited for naive users; that is, it was eventually fairly safe to use, and could in large part be taught to rank beginners in a couple of hours—provided they didn't have to know about JCL cards.

It is left for the reader to figure out interesting uses for it. How would you do collateral structures? How could you signal to a reader which of several pieces of text a jump was to?

(At least one real hypertext was actually written on this system. It tied together a lot of patents for multilayer electrodes. Readers agreed that they could learn more from it about multilayer electrodes than they had imagined wanting to know.)

Development of hypertext software and approaches has been underway at Brown University from 1968 to the present, under the leadership of Andries van Dam. It is continuing now under Norm Meyrowitz, with interface design (on the Mac) by Nicole Yankelovich.

## BIBLIOGRAPHY

Steven Carmody, Walter Gross, Theodor H. Nelson, David Rice and Andries van Dam, "A Hypertext Editing System for the 360." In Faiman and Nievergelt (eds.), *Pertinent Concepts in Computer Graphics* (U. Ill. Press '69), pp. 291-330.

# IDEA TRANSFER

"Idea transfer" is a clumsy term, but I want to talk here about a very important problem that is usually covered under other headings that obscure certain main issues.

By "idea transfer" I simply mean **getting ideas across**. Now, some of this is often called "teaching," but that immediately calls up images of Rydell High, Ed School and the Teachers' Union.

Another part of idea transfer is **writing**. But that connotes (for different eras) the quill pen, the Fedora hat and the bottle of bourbon, and lately the computer screen.

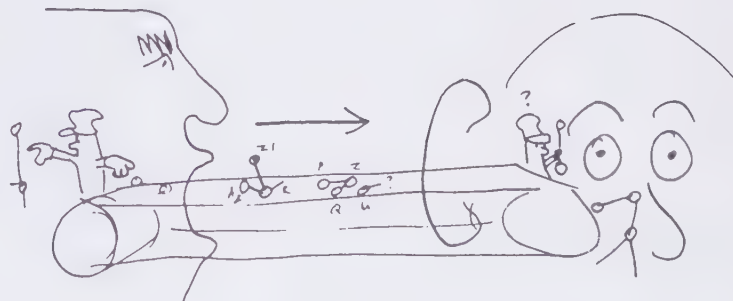
But writing is not these externals. It is an idea process. The skillful writer weaves many simultaneous aspects and presentations, tuning emphasis and shading, hinting about what is to come, letting thoughts dangle for later reactivation, and maintaining atmospherics, all at the same time.

You may say that this is not true of "technical writing." I would simply point out that technical writing is usually an enforced system of verbal enumeration under very restrictive rules. It may assure (at best) that certain things are to be found in certain findable places, but the problem of

↓ ↓ ↓

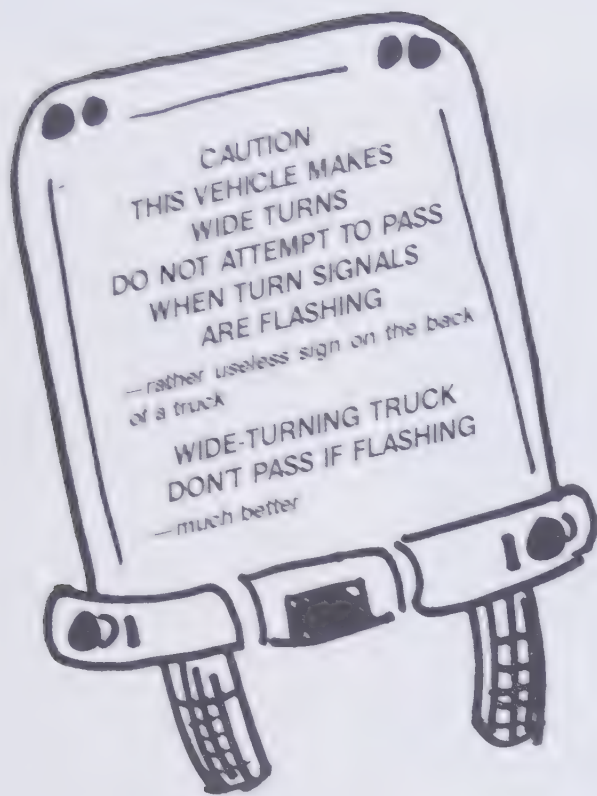
WRITING IS A SPECIAL CASE OF DESIGN. And one of the most arbitrary and indeterminate.

↓ ↓ ↓



**Verbal communication—whether written or spoken—is the disassembly of the Tinkertoy of thought into pieces, and placing it on a conveyor belt to its place of reassembly.**





Perhaps the best computer manual ever written was the original manual for the original video game, "Pong":

AVOID MISSING BALL FOR HIGH SCORE.

(This is a ball!)

Missing "I"

THE BIGGER THE SOFTWARE MANUAL, the less chance that people will read it or understand the system; therefore, THE WORSE THE DESIGN OF THE SYSTEM.

A good design is one that can be communicated easily.

When we write the manual for the IBM PC version, we'll put in longer words."

—Alan Boyd, referring to his company's GUIDE hypertext program for the Macintosh

estimating the reader's overall orientation, which is the hardest part, is not improved by any system of rules I know of. And that is the real problem.

Idea transfer may use words, pictures, animations, interactive screen diagrams, theatrical plays and movies, pop-up books, elegant and fancy typography—even perfumed paper. There's no one way and there are no good rules. Except to strive for simplicity and clarity.

And among these modalities of presentation, walking among them as a writer, is the basic idea of *fancies* (see p. DM 74; see also the remark on modalities of apprehension by Bottom the Weaver, p. DM 149), of which writing, design and moviemaking are specific practices.

### THE SOCIAL COMPLEXITIES OF COMMUNICATING IDEAS

Between people—whether in conversation between friends or between student and teacher—there are many levels of interplay: issues of mutual status claim and support (or denial); flirtation and teasing (including "Ah, but that's what we'll be covering next week!"); admirations, dislikes and resentments, as well as echoes and references to ongoing issues and struggles between the parties. These have nothing to do with the subject, but wind around and hide behind the subject, frequently obscuring motivation, but also frequently in the direction of actually getting things across.

The socially neutral presentation does not have these complications. This is a special advantage of socially neutral en-

tities, like maps and some texts. (Many texts are not socially neutral because the author is strongly present as a personality.) Yet many enthusiasts of artificial intelligence and computer-assisted instruction want to create socially-active entities. (See pp. DM 120 and DM 129.)

Hypermedia can be socially neutral or not. An open hypermedia network should combine the best features of both (see p. DM 141).

### "ON WRITING,"

a paradigm of the creative process being an examination of some very Complex Matters which Nobody Seems to Understand; and whose Generality of Relevance may be Gradually Apprehended. (Eventually I hope to develop a somewhat more formal treatment of "ideas," as distinct from propositions, sentence kernels, etc. But there is certainly no room for that here. Logicians: show me the truth-table of "BUT.")

The process of writing is poorly understood in most quarters. Many working writers despair of being "systematic," getting things done as best they can. On the other hand, people who think they might be able to contribute—particularly the symbolic logicians and transformational linguists—being immersed in their own formalisms, simply don't see what's going on—at least, when I've tried to talk to them.

Writing is not simple. As with vision or speech or riding a bicycle, an immensely complex process is being unconsciously pursued.

I have in hand a collector's item—a pamphlet called HOW TO SURVIVE A HOTEL FIRE from the National Safety Council, Chicago, IL. To imagine that anyone would read its seventeen fine-print "step-by-step" instructions transcends imbecility, reaches a remote plane of rarefied foolishness. However, an excellent exercise for writer-designers would be to reduce its advice to a useful, single-glance card that could be glued on the back of a hotel door.

Some people think you make an outline and follow it, filling out the details of the outline until the piece is finished. This is absurd. (True, *some* people can do this, but that is simply a shortcutting of the real process.) Basically writing is

THE TRY-AND-TRY-AGAIN INTERPLAY OF PARTS AND DETAILS against OVERALL AND UNIFYING IDEAS WHICH KEEP CHANGING.

In fact a number of things are happening, often simultaneously. We can separate them into three:

1. Provisional development of ideas and points:

A) forming overall organizing ideas; B) selecting tentative points; C) inductively finding overall organization among them; D) finding relations of interest between points.

2. Complex sifting and adjustment among collections of points, overall ideas.



3. Fine splicing within developed sequences.

A) transition and juxtaposition managements, B) cross-citations, C) smoothing.

Regrettably, there's no room or time to pursue this here. (The article I had intended to write would take a whole spread.) For people who really care about the matter, I will make some points in very abbreviated form.

The interesting structures in written material include:

"Points"—pieces, sentences, phrases, examples, plot events, and expository "points."

Organizing principles and structures (which we will call here *arches*)—final ironies, things to be led up to, themes, plots, concepts, principles, expository structures, organizing titles, overconcepts. These may be either local or global, over the entire work. (Note: arches may not be hierarchical relative to one another.)

Now, we may think of *points* and *arches* as individual objects which have individual relations to one another. Between two points there may be a good *transition*; a specific point may link well to a specific arch.

The problem in writing, then, is that overall structures you choose (systems of arches) may not link well to the points that have to be included among them; and that transitions between points don't work out the way you want them to. Good transitions can't be worked out for the sequence of points you want to make, or, alternatively, there are too *many* good

transitions within a specific structure of points, and picking among them involves difficult choices—especially when you have to devise appropriate arches on the basis of the final sequence of points.

There are a number of other important structures in written material. They include accordances, juxtapositions, cross-citations, connotations, nuances and rhythms.

The only ones we will discuss here are *accordances*.

The term "accordance," as I shall use it here, is simply a vaguely formal way of talking about whether things *match* or *fit together*. Two items are *in accord* if they

match or fit well, or *in discord* if they match or fit badly. Thus a good transition between points (as mentioned early) represents an accord, and a good link between a point and an arch is also an accord.

Now, it happens that a great deal of writing is concerned with notes to the reader about accordances in the material. In fact, quite a few words are exclusively concerned with subtly pointing out to the reader the accords and discords within the expository structure of what he is reading. We may call these *accordance-connectives* or *accordance-notes*.

Two of the most basic terms are *indeed* and *but*.

The word *indeed* has an interesting function.

The word *indeed* (in its main use, at the beginning of a sentence) indicates an accord between what has just been said and what is to follow. In other words, it functions as a positive transition, impetus or gas pedal, indicating a continuation of the flow in the direction already indicated. So do the words *thus*, *then*, *therefore*, *moreover*, *so* and *furthermore*. These are *infix accords*, that is, notes of accord that go between two items. We also see *prefix accords*, such as *since*, *inasmuch as*, *insofar as*; these have to be followed by two clauses, the second of which is in accord with the first.

The word *but* is exactly the opposite. It indicates a discord or contradistinction, a negative transition, "brakes" in the flow. Other such *infix discords* include *nevertheless*, *despite this*, *on the other hand*, *even so*, and "Actually, . . ." Similarly, there are *prefix discords*: *while*, *despite*, *though*, . . . , *notwithstanding*.

I find this topic of inquiry very interesting. These sorts of terms have been used since time immemorial by writers adjusting their transitions for smooth flow (note such antiquity variants as *haply*, *howbeit*, *withal*, *forasmuch* and *howsomever*), but the importance and structure of this service has not, I think, been generally understood.

(Note also that there are more intricate *accordance-connectives*: I wish we could go here into the structure of *in fact*, . . . , *at least*, . . . *if not*, . . . , *otherwise*, . . . , *Anyway*, . . . , and *Now*, . . . )

(Note: the try-and-try-again revision and reconsideration process, tinkering with structural interconnections, is a uni-

versal component of the creative process in everything from movie editing to machine design. There ought to be a name for it. I can't think of a satisfactory one, although I would commend to your attention *grandesigning*, *piece-whole diddlework*, *grand fuddling*, *metamogrification*, and that most exalted possibility, *tagnebulopsis* (the visualization of structure in clouds).

## DIAGRAMMATICS

The ability to read diagrams—of which maps are just the simplest case—appears to have been on the decline for some years. It is not taught separately but only incidentally as part of science or "social studies" courses. (But so, I think, has the ability declined to *make* creative diagrams. Great biologists like Agassiz and Simpson, physicists like John Wheeler have possessed this ability to a remarkable extent.)

The Scientific American is the bulwark of diagrammatics in the present age. Its diagrams range from those that almost any ten-year-old can understand up to things only a nuclear physicist can grasp.

In the nineteen-forties, TIME and FORTUNE magazines had superb collage diagrams which merged pictorial material with numbers and relationships. These were mostly the work of Herbert Bayer. Surprisingly, this excellent tradition has been revived in the newspaper USA TODAY.

One aspect of diagrammatics—the most violated—is how to show numerical relationships. One sin is making symbols misleading sizes (as they do on TV news, where the stock market's rise and fall is shown by an immense arrow whether the overall motion is great or tiny). Another is leaving off the bottom of a chart, so that minor fluctuations appear Himalayan. These matters have been splendidly treated in Edward R. Tufte's The Visual Display of Quantitative Information (Graphics Press, 1983).

Unfortunately, the much larger issue of the design of multipart diagrams, merging pictures and numerics and concepts, has not to my knowledge been covered anywhere; let alone adding responding animation; let alone embracing these in hypertext. All of which together will be a principal medium of the future. ■



A lot of people are afraid to ask questions because they're afraid of looking dumb. But the dumb thing is not asking questions.

"The exhibit should be equally interesting to a four-year-old or a person with an advanced degree."

—Spokesman for San Francisco's Exploratorium, the world's finest science museum.

ARE YOUR LIGHTS ON?

—Sign at the end of a tunnel that takes care of all four cases (courtesy of Don C. Gause and Gerald E. Weinberg, authors of the book by the same name, Little, 1982)

## HOW TO LEARN ANYTHING

As far as I can tell, these are the techniques used by bright people who want to learn something other than by taking courses in it. It's the way Ph.D.'s pick up a second field, it's the way journalists and "geniuses" operate; it brings the general understandings of a field that children of eminent people in that field get as a birth-right; it's the way anybody can learn anything, if he has the nerve.

1. DECIDE WHAT YOU WANT TO LEARN. But you can't know *exactly*, because of course you don't know exactly how any field is structured until you know all about it.

2. READ EVERYTHING YOU CAN ON IT, especially what you enjoy, since that way you can read more of it and faster.

3. GRAB FOR INSIGHTS. Regardless of points others are trying to make, when you recognize an insight that has meaning for you, make it your own. It may have to do with the shape of molecules, or the personality of a specific emperor, or the quirks of a Great Man in the Field. Its importance is not how central it is, but how clear and interesting and memorable to you. REMEMBER IT. Then go for another.

4. TIE INSIGHTS TOGETHER. Soon you will have your *own* string of insights in a field, like the string of lights around a Christmas tree.

5. CONCENTRATE ON MAGAZINES, NOT BOOKS. Magazines have far more insights per inch of text, and can be read much faster. But when a book really speaks to you, lavish attention on it.

6. FIND YOUR OWN SPECIAL TOPICS, AND PURSUE THEM.

7. GO TO CONVENTIONS. For some reason, conventions are a splendid concentrated way to learn things; talking

to people helps. Don't think you have to be anybody special to go to a convention; just plunk down your money. But you have to have a handle. Calling yourself a Consultant is good; "Student" is perfectly honorable.

8. "FIND YOUR MAN." Somewhere in the world is someone who will answer your questions extraordinarily well. If you find him, dog him. He may be a janitor or a teenage kid; no matter. Follow him with your begging-bowl, if that's what he wants, or take him to expensive restaurants, or whatever.

9. KEEP IMPROVING YOUR QUESTIONS. Probably in your head there are questions that don't seem to line up with what you're hearing. Don't assume that you don't understand; keep adjusting the questions till you can get an answer that relates to what you wanted.

10. YOUR FIELD IS BOUNDED WHERE YOU WANT IT TO BE. Just because others group and stereotype things in conventional ways does not mean they are necessarily right. Intellectual subjects are connected every which-way; your field is what you think it is. (Again, this is one of the things that will give you insights and keep you motivated; but it will get you into trouble if you try to go for degrees.)

There are limitations. This doesn't give you lab experience, and you will continually have to be making up for gaps. But for alertness and the ability to use his mind, give me the man who's learned this way, rather than been blinkered and cliched to death within the educational system.

## BIBLIOGRAPHY

Wilmar Shiras, *Children of the Atom*.

Science-Fiction about what a school could be like where kids *really* used their minds. I've always been sure it was possible; the R.E.S.I.S.T.O.R.S. (see p. 110) made me surer.

# TEXT SERVICES AND "NETWORKING"

A part of today's computer scene that almost nobody foresaw is the development of "networks" and networking. Great vitality drives the movement. And all you need is a computer (or a terminal) and a modem.

While the term "network" has many terms in engineering-and electronics, to computer users it seems to mean **services you can put text into and get text out of.**

Example: The Source and Compu-Serve are popular time-sharing systems (see p. 106), but one of their largest uses is for storing and retrieving text by users.

You could spend all your time on this stuff, much as you could spend your life reading want ads or wandering through department stores. But the sense of direct and inexpensive contact with many people is invigorating and addictive.

All kinds of special-interest groups meet through networks. There is even an on-line witch's coven.

You may think of the world of computer text network services as a lavish assortment of wonderfully different possibilities, with overlapping functions and purposes, or as a seething chaos, a Babel of unnecessarily different activities, mutually inaccessible, with overlapping functions and purposes. Both views are correct. Text networks are all different, and they're all complicated, and you have to learn lots of codes.

There are various kinds of text networks, with different names, but they all have approximately the same basic structure: they are **forms of storage** that you put stuff into and get stuff out of.

There are a number of major types; but they differ more by ownership and style of operation than by any fundamental characteristics.

We may distinguish five:

1. **ELECTRONIC MAIL** consists usually of stored messages passed from computer to computer till they land in a filing location called your Mailbox on a specific computer. You may then read these messages at your leisure. (Some systems will go on holding your old mail indefinitely, in which case you may consider it "filed," sort of.)

Members of the general public may establish mailboxes on services like MCI, The Source, Compu-Serve.

**Internetwork mail.** It is easy to send mail between the main networks, by naming the addressee and the network he or she is on. This is a lot handier and faster than conventional mail.

2. **TELECONFERENCING SYSTEMS** are a peculiar new form of intercourse pioneered by such individuals as Karl Zinn at the University of Michigan and Murray Turoff at the New Jersey Institute of Technology.

A "conference" is like a long scroll of text. Individuals may add to the end, and eventually the beginning is destroyed. Readers, at their home and office screens, may search through the available messages by topic. A contributor may include links to previous items, so that a conversation can skip along the scroll. Different conferences have different strange rules as to who may read and who may contribute, and how the conference manager, or editor, or gatekeeper, is to treat the materials entrusted to her or him.

## KEY-TO-FOOT MAIL

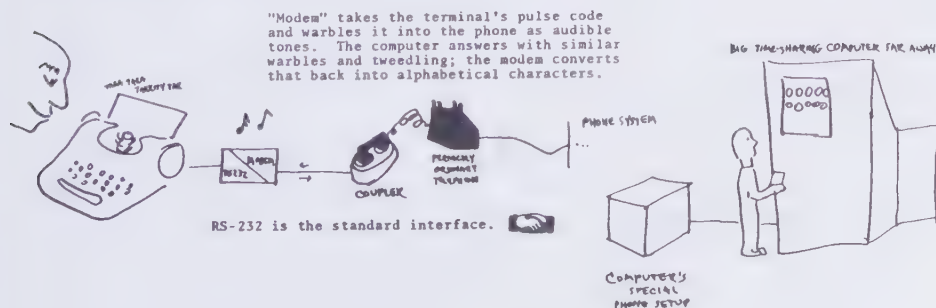
There are various ways you can send mail from your computer keyboard into the conventional mailstream of foot-delivered government service, using services like MCI.

**THE NETWORK DREAM:** instant access to ideas, facts, opinions, personal opportunities, advice, methods and tools, supplied by everyone on a volunteer basis.

**THE REALITY:** some of that, but enough complication to place a pretty firm limit on usage.

## THE MIRACLE OF OVER-THE-PHONE TERMINALS

(Some people go as far as to see the typewriter going by itself)



"S"

SourceVoid Dave" is the network name of a retired military man, a network evangelist, who runs a bulletin board in Colorado. One of the things Dave likes best about networking is that you get to chat with all kinds of people without knowing who they are. Many prominent people hide their identity for these conversations.

Cynics say these are no different from conversations in bars, but they cost about the same, are healthier, and don't lead to traffic accidents.

There have as yet been no publicized cases of fraudulent electronic mail or counterfeit documents on commercial networks. But such problems will no doubt occur.



Knowing your way around lots of networks is like knowing your way around the different branches of the New York subway system—an interesting skill whose full refinement virtually precludes useful activity.

"Networks are great for teenagers," says a friend of mine. "They love mastering obscure and complex things that adults find difficult, and they can communicate freely without dealing with a social situation."

**A recent piece in the Atlantic describes the daily ordeal of an expatriate American in Kuala Lumpur, Malaysia, hooking up his portable Radio Shack computer to MCI Mail from the phones in the street, to the amusement of the local populace. Westerners still, as in the 19th Century, uphold their sense of identity in foreign places at all costs.**

3. **BULLETIN BOARDS**, a grass-roots movement, are shared-text systems that don't charge money to their users. They are generally run by altruistic enthusiasts, referred to as "sysops" (pronounced sissops), who hook a small computer to a phone line and let people put items on it for others to read by calling up through *their* computers. There are *hundreds* of bulletin-boards; a large listing can be found in *Computer Shopper* magazine.

The problem with this is the lack of control. Anyone can put in anything, and has. Pirated copies of software are often posted. This has caused justifiable outcry among software vendors. In one case, a contributor "shared" a copy of PC-Write (which carries a standing permission to be given away), maliciously modified so as to destroy the contents of users' hard disk drives.

In other cases, bulletin boards have carried ads for prostitutes.

It is clear that sysops may have legal liabilities of which, in their enthusiasm, they are only dimly aware. Indeed, bulletin board operators have had their equipment seized by FBI raiders who left explanatory notes on the refrigerator. This contentious area is a promising route for government oppression to begin.

4. **ELECTRONIC PUBLISHING**, on a network basis, takes various forms; some services, such as *The Source*, allow you to place a document on line that users may make copies of, paying a royalty. One problem, however, is not being able to sample it. (However, digital publishing is also done by such database vendors as Mead.)

5. **"COMMUNITY ACCESS" SYSTEMS**

Then there is the Community Memory movement. The basic idea of Community Memory is to have a computer resource of information and ideas commonly available. In its more glorified and mystical form, the idea seems to be to have a place, inside the computer, where information can be shared by The People, free of institutional obstruction or the profit motive.

This vision is perhaps unclear to others besides the author, but it attracts a variety of people interested in some form of grass roots revitalization of our society. Some of these are disillusioned sixties radicals who look to "community organi-

zation" as a building block for a new society; others are interested in more nuts-and-bolts applications, such as trying to make barter a viable economic form again, in an urban society with many nonstandard leftovers, skills and wants. (Presumably this would work by having the computer find pairs of people with matching wants and tradables; or even search out potential trades around multi-person rings.)

The first of these systems was Resource One, in San Francisco; I saw another Community Memory in Vancouver, which seemed to be in practice a sort of animated classified-ad system. A user sitting at the terminal can put in ads of his own, and can search through the entire file for keywords of interest. As there is no censorship, some rather surprising things get in there, for which I wish we had room.

Kirk Brainerd, of L.A., is using computers for a registry of people with something to teach. He hopes that if people are mutually available to each other at a deep enough level, people can begin to act out of altruism in general.

### SOME NOTABLE NETWORKS

ARPANET was intended for military research, and had many provisions whereby people could run programs on far-off computers, but in fact its main use has turned out to be for text interchange, especially electronic mail.

USENET is a voluntary network among UNIX users, whose computers automatically dial each other in the night to exchange mail. There are various conferences on USENET; one of them, called *Human-Nets*, is fairly gossipy. Inquiries into "Whatever happened to So-and-So?" tend to get quick answers.

*The Source*, owned by Reader's Digest, was the first service to drop prices to a few dollars per hour, night-time, and thus make itself economically accessible to the public.

Compu-Serve, originally a professional time-sharing system, likewise dropped prices and made adjustments for the personal market.

MCI, mostly a telephone company, has had considerable success with its electronic mail system.

(IBM and Sears are now sponsoring Trintex, a startup, to compete with *The Source*.)

EIES, started by Murray Turoff and now sponsored by the State of New Jersey, is a semi-commercial conferencing system run by the New Jersey Institute of Technology.

*The Well*, run by the Whole Earth Catalog people, is called the "counterculture node." (But its command structure is as complicated as most.)

Byte magazine runs something called BIX, the Byte Information Exchange, a large-scale conference with a chipmonk slant.

FRANCE has a totally different system from the United States. The phone system, which is run by the government, is now mostly a digital network, and the government has given out little keyscope terminals to subscribers throughout the country. (This system preceded the spread of personal computers in France.)

### BALKANIZATION

Now, perhaps you wonder why it can't all be unified, a single integral virtuality, and I do too. Why those strange distinctions? Why those strange rules? Why should being "in a conference" preclude publication? Why is it all so temporary? Why can't archiving be a part of it all?

The alternative some of us hope for: a unified and universal system, is discussed on p. DM 141.

### BIBLIOGRAPHY

John S. Quarterman and Josiah C. Hoskins, "Notable Computer Networks," CACM, October 1986, pp.932-971.

Murray Turoff and Starr Roxanne Hiltz, *The Network Nation: Human Communication Via Computer*. Addison-Wesley, 1978.

Jacques Vallee, *Computer Message Systems*. McGraw-Hill, 1984.

John Brunner, *The Shockwave Rider*. Ballentine, 1978. Seventies science fiction about network hacking.

Vernor Vinge, *True Names*. Bluejay Books, 1984. Eighties science fiction about network hacking.

For a discussion of the U.S. Post Office's ill-fated foray into electronic mail, correctly predicting disaster even before the Post Office's service began, see T. Nelson, "The Magicians, the Snark and the Camel," *Creative Computing*, December 1981.

# NEW INTERACTIVE TECHNIQUES

Designing screen systems that focus the user's thought on his work, with helpful visualizations and no distractions, is the great task of fantic design.

Indeed, many of the wonderful interactive techniques that we saw in the video-game days have not yet made it into the software markets: things like shrunk parallel views, jingle-sound feedback (as far as I know, a word processor I did was the only one yet with different sounds for the different edit operations, see p. DM 71). Video games, too, showed us how much can be done with a joystick (as in Pac-Man). But still the notion of **high performance** has not really sunk in.

Yet it is amazing to see how fast people settle down to a certain way of doing things. First the Macintosh was "radical"; now it's "the other way of being" (compared to conventional computers) and its well-intentioned but clumsy mechanisms have acquired an odd sacredness among its acolytes. (Dogmatically enforced, too, by directives from Apple.)

However, to move on to tomorrow's **high-performance** interactive systems which will be smoother and more natural a wide variety of new interactive techniques will have to be perfected. I present this gallery of my own designs and thoughts with inspirational intent.

## XANADU

(c) 1972 T. Nelson

*This is the original full design for the Xanadu™ user's console. The Xanadu system has become a much more general storage system, but it is still targeted for this kind of high-performance interactive console (see p. DM 141).*

### THE PARALLEL TEXTFACE™

This user-level system is intended to aid in all forms of writing and scholarship, as well as anywhere else that we need to understand and manipulate complex clusterings of text (i.e., *thought*). It will also work with certain animated graphics.

The Parallel Textface, as described here, furnished the initial impetus for the development of the Xanadu™ system (see p. DM 141). Xanadu was developed, indeed, originally for the purpose of implementing some of these functions, but the two split apart. It turned out that the Parallel Textface required an extremely unusual data structure and program techniques; these then became the Xanadu system. As developed in the final Xanadu design, they turn out to handle some very unusual kinds of screen animation and file retrieval. But this grew out of structuring a system to handle the functions described here.



(c) 1972 T. Nelson

Thus, the Parallel Textface basically requires a Xanadu system.

It is hoped that this system can be sold complete (including minicomputer or microprocessor—no connection to a large computer is required) for a few thousand dollars by 1976 or 1977. (Since "business people" are extremely skeptical as to whether anybody would want such a thing, I would be interested in hearing expressions of interest, if any.)

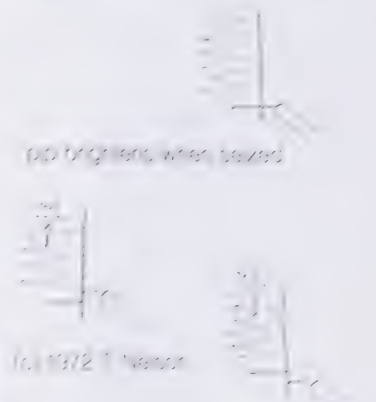
These economics are swimming into reach in 1987. What you see here are front-end functions, not yet programmed; the storage system that supports them, however, has finally reached prototype stage.

As shown here, the screen presents two panels of text; more are allowed. Each contains a segment of a longer document. ("Page" would be an improper

term, since the boundary of the text viewed may be changed instantly.)

The other odds and ends on the screen are hidden keys to control elements which have been made to fade (in this illustration), just to lessen the distraction.

Panel boundaries and control graphics may be made to appear by touching them with the lightpen.



(c) 1972 T. Nelson

### MOVING FUNCTIONS

The first movement is the screen (horizontal). The lower right hand corner of each panel contains an arrow pointing down and to the right. The light pen can touch this arrow to move the panel up or down. The light pen can touch the pip up or down. "Up" causes the text to move smoothly upward (forward in the material), at a rate proportional to how far you push the pip; "down" causes it to move back. (Note that we do not refer here to jerky line-by-line jumps, but to smooth, linear motion, which is essential in a high performance system. If the text does not move, you can't tell where it came from.)



## MANIFESTO

My work is concerned principally with the theory and execution of systems useful to the mind and the creative imagination. This has practical and theoretical aspects: I claim that the precepts of designing systems that touch people's minds, or contents to be shown in them, are simple and universal; making things look good, feel right and come across clearly. I claim that to design systems that involve both machines and people's minds is art first, technology second, and in no way a derivative specialty off in some branch of computer science.

However, presentational systems will certainly involve computers from now on.

Since hundreds of such systems are now being built, many of them all wrong, we must teach designers (and certain others) the basics of computers, and give them some good examples to emulate.

Further, the popular superstitions about computers must be fought—the myths that they are mechanical, scientific, objective or independent of human intent and contemplative involvement.



## PARALLEL TEXTFACE (1971)



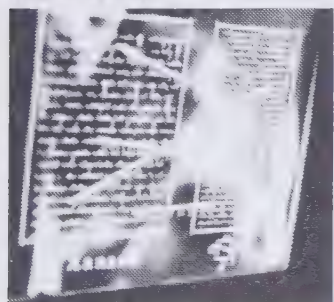
Real person sits at cardboard Xerox UI mockup



"Nice keyboard! But what happened to your typewriter?"



Two panels are about right for a 10 x 10 screen



Independent text pulls dependent text along. Painted streaks simulate motion, not icicles

**DERIVATIVE MOTION:** when links run sequentially, connecting one-after-the-other on both sides, the contents of the second panel are pulled along directly: the smooth motion in one panel is matched in the other. This may be called *derivative motion*, between independent text (being handled directly with the lightpen) and dependent text (being pulled along). The relationship may be reversed immediately, however, simply by moving the lightpen to the control pip of the other panel, whose contents then become the independent text.

Irregularities in the links will cause the independent text to move at varying speeds or jump, according to an average of the links' connectivity.

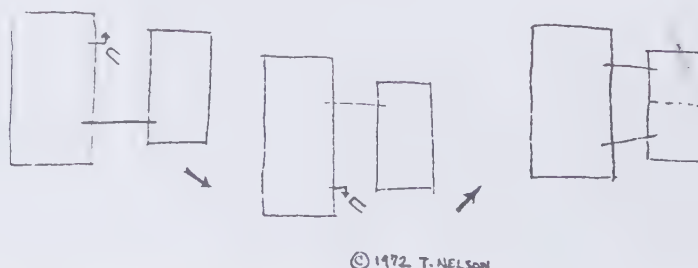
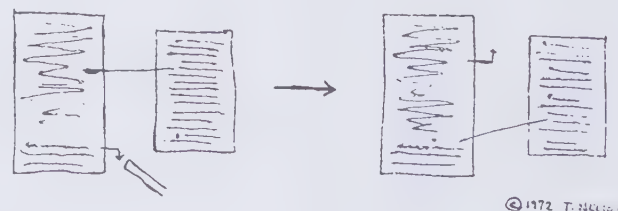
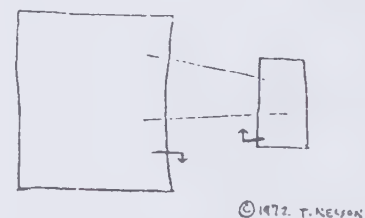
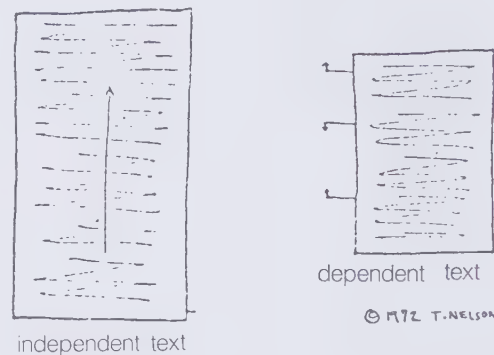
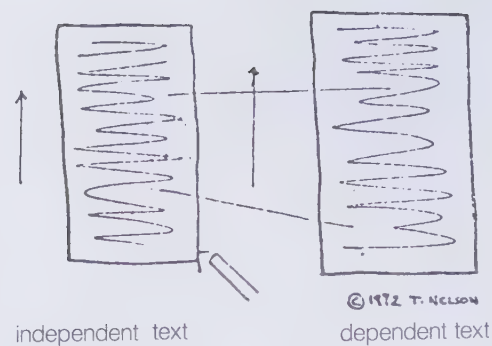
If no links are shown, the dependent text just stops.

Collateral links between materials in the two panels are displayed as movable lines between the panels. (Text omitted in this diagram; panel boundary has been made to appear.)

Some links may not have both their endpoints displayed at once. In this case we show the incomplete link as a broken arrow, pointing in the direction of the link's completion.

The broken arrow serves not merely as a visual pointer, but as a jump-marker leading to the linked material. By zapping the broken arrow with the lightpen, the user summons the linked material—as shown by the completion of the link to the other panel. (Since there has been a jump in the second panel, we see that in this case the other link has been broken.)

When such links lead to different places, both of these destinations may nevertheless be seen at once. This is done by pointing at both broken links in succession; the system then allows both links to be completed, breaking the second panel between the two destinations (as shown by dotted line across panel).



## FAIL-SAFE AND HISTORICAL FEATURES

In systems for naive users, it is essential to safeguard the user from his own mistakes. Thus in text systems, commands given in error must be reversible. For instance, Carmody's system (see p. DM 35) requires confirmation of deletions.

Another highly desirable feature would allow the user to view previous versions, to see them collaterally with the corresponding parts of current versions, and even go back to the way particular things were and resume work from the previous version.

In the Parallel Textface this is all comprised in the same extremely simple facil-

ity. (Extremely simple from the user's point of view, that is. Inside it is, of course, hairy.)

In an egregious touch of narcissistic humor, we use the very trademark on the screen as a control device (expanded from the "X" shown in the first panel).

Actually the X in "Xanadu™," as it appears on the screen, is an hourglass, with a softly falling trickle of animated dots in the lower half, and Sands of Time seen as heaps above and below. These have a control, as well as a representative, function.

TO UNDO SOMETHING, YOU MERELY STEP "BACKWARD IN TIME" by dagging the upper part of the hourglass with the lightpen. One poke,

one editing operation undone. Two pokes, two operations.

You may then continue to view and make changes as if the last two operations had never taken place. This effectively creates an alternative time-line.\* However, if you decide that a previously undone edit operation should be kept after all, you may step **forward**—stepping onto the previous time-line—by using the lower half of the hourglass.

We see this clarified in a master time diagram or Revision Tree which may be summoned to the screen, never mind how. In this example we see that three versions are still "current," various other starts and variations having been abandoned.

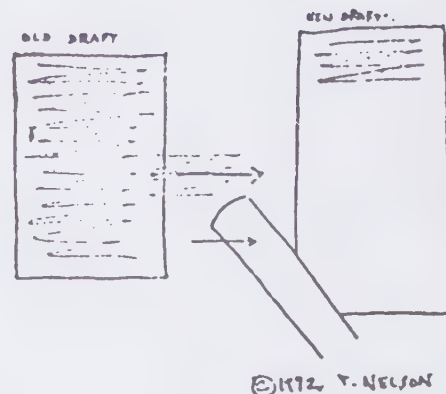
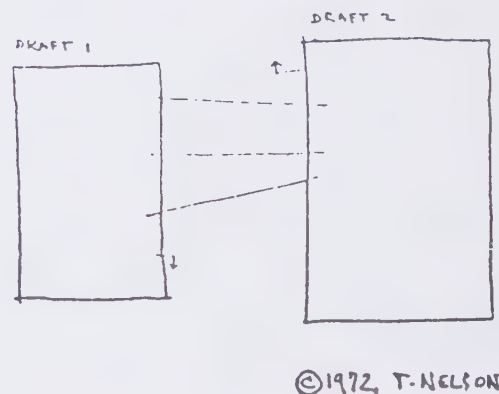
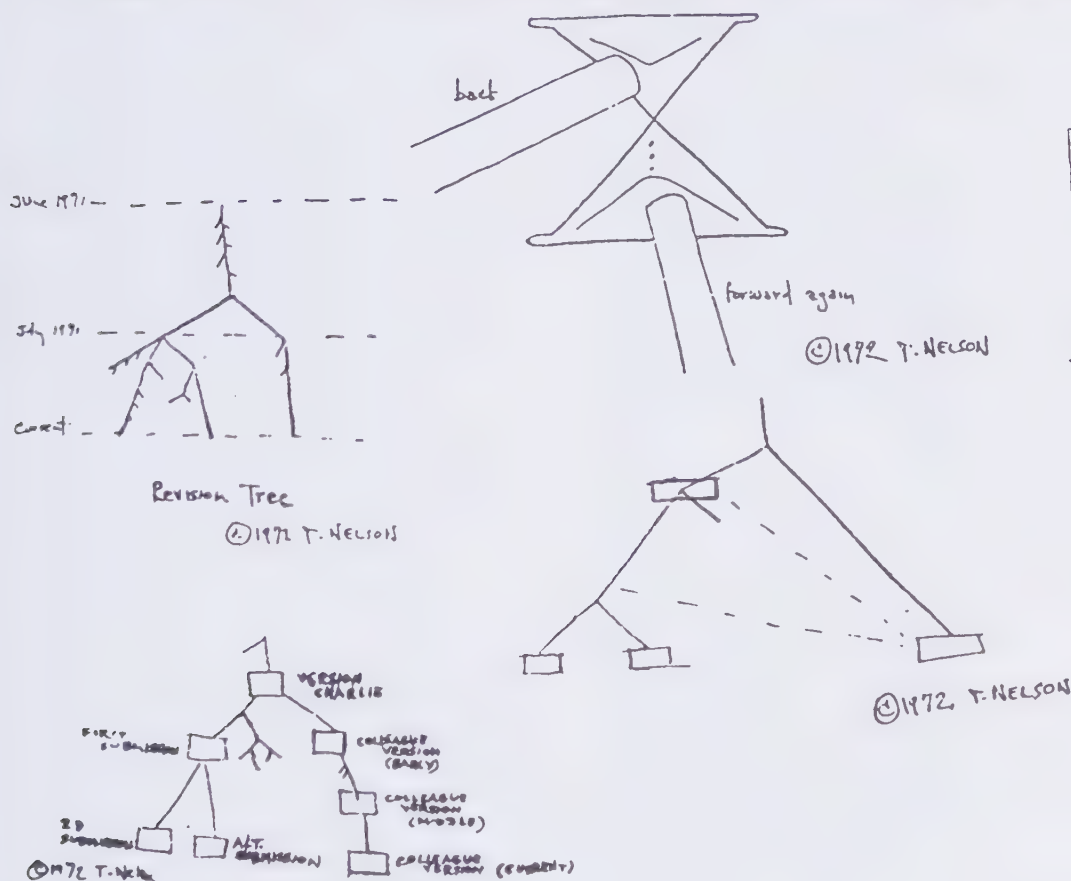
(The shaggy fronds correspond to short-lived variations, resulting from operations which were then reversed. In other words, "excised" time-lines, to use Gerrold's term—see footnote.)

The user—let's say he is a thoughtful writer—may define various Versions or Drafts, here marked on the Revision Tree.

He may, indeed, define collateral linkages between different versions defined at various Times in the Tree...

...and see them displayed collaterally; and revise them further.

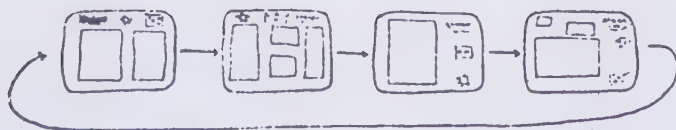
Materials may be copied between versions. (Note that in the copying operation of the Parallel Textface, you actually see the moved text moved **bodily as a block.**)



\* Oddly, this has the same logical structure as time-travel in science-fiction.

There are basically three alternate premises of time-travel: 1) that the past cannot be changed, all events having preceded the backstep; 2) that the past **can** be changed; and 3) that while time-travelers may be deluded into thinking (2), (1) is really the case—leading to various appointment-in-Samarra plots. Only possibility (2) is of interest here, but there are various alternative logics of mutability and time-line stepping. One of the best I have seen is in *The Man Who Folded Himself* by David Gerrold (Popular Library, 1973); logic expounded pp. 64-68. I am bemused by the parallel between Gerrold's time-controls and these, worked out independently.





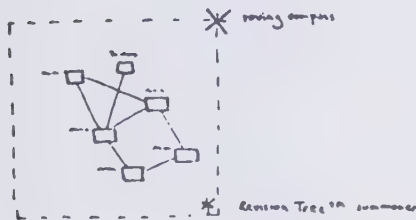
©1972 T. NELSON

## GETTING AROUND

The user may have a number of standby layouts, with different numbers of panels, and jump among them by stabs of the lightpen.

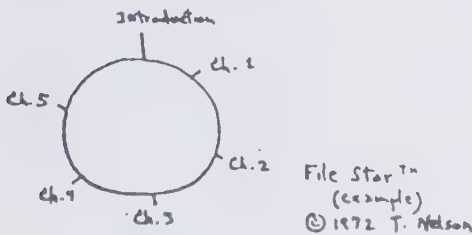
Importantly, *the panels of each can be full*, each having whatever the contents were when you last left it.

### File Web™

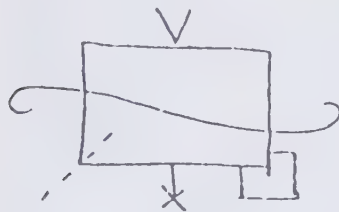


©1972 T. NELSON

The File Web™ is a map indicating what (labelled) files are present in the system, and which are collaterated.



The File Star™ is a quick index into the contents of a file. It expands as long as you hold the lightpen to the dot in the center, with various levels of headings appearing as it expands. Naturally, you may jump to what you point at.

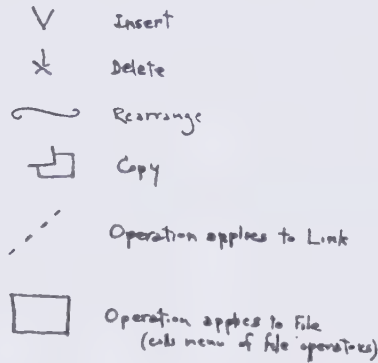


©1972 T. NELSON

## EDITING

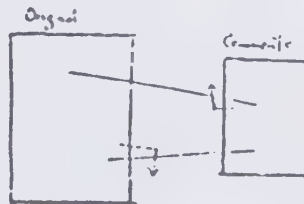
Rather than giving the user anything complicated to learn, the system is com-

pletely visual. All edit controls are comprised in this diagram, the Edit Rose™. Viz.:



©1972 T. NELSON

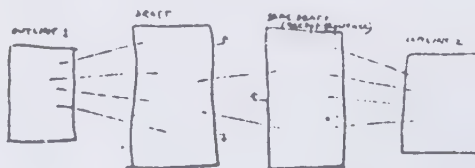
Separate portions of the Edit Rose invoke various edit operations. (You must also point with the lightpen to the necessary points in the text: once for insert, twice for Delete, three or four times for Rearrange, three times for Copy.)



©1972 T. Nelson

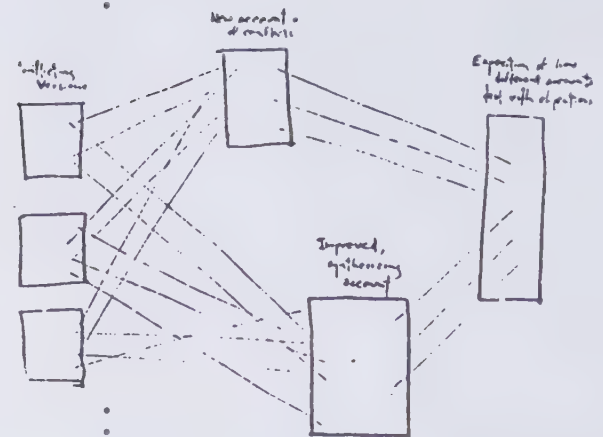
## GENERALITY

The system may be used for comments on things,



©1972 T. NELSON

for organizing by multiple outlines or tables of contents;

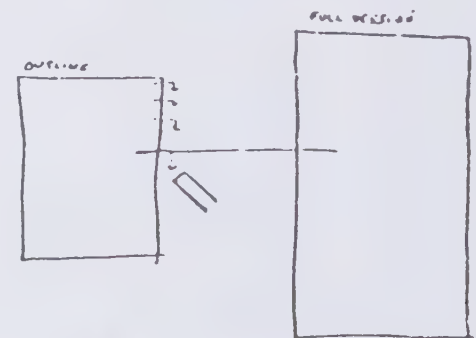


©1972 T. NELSON

and as a Thinkertoy, organizing complex alternatives. (The labels say: "Conflicting versions," "New account of conflicts," "Exposition of how different accounts deal with objections," "Improved, synthesizing account.")

In other words, in this approach we annotate and label discrepancies, and verbally comment on differences in separate files or documents.

In ways this may seem somewhat obtuse. Yet above all *it is orderly*, and the complex of collateral files has a clarity that could be all-too-easily lost in systems which were programmed more specifically to each problem.



©1972 T. NELSON

The fundamental strength of collaturation, seen here, is of course that any *new* structure collateral to another may be used as a table of contents or an outline, taking the user instantly to parts which are of interest in some new context.

## SEGMENTED ICONS

"Icons" are very popular right now—meaning little screen symbols that show you an option (so you don't have to remember or type a name for the option).

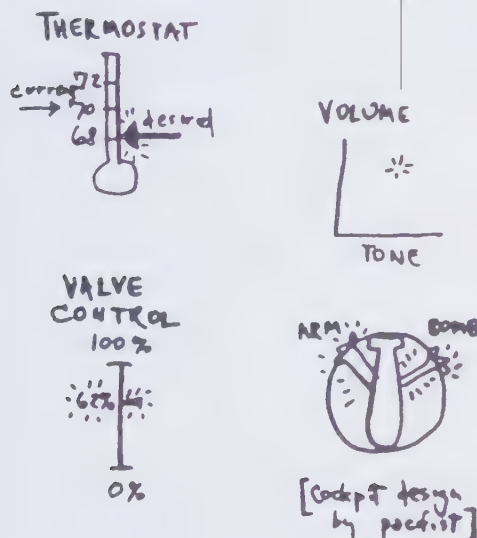
In my own iconics, starting in the sixties, I have assumed that icons and other visual screen controls should have **multi-faceted manipulable parts** (such as the Edit Rose of the Parallel Textface on the previous page). This idea has not generally gotten across yet.

Many important possibilities need to be tried with these cuing screen graphics: animating parts of them, or having them palpitate to show more than one thing at once, or having their changing palpitations indicate different ways you can manipulate and explore them at that instant. It's surprising how few things have been tried.

## SCREEN CONTROLS

The great thing about CRT displays is that they can be used to *control things by manipulation of pictures*. Instead of moving buttons or levers, you can seize parts of the picture with the lightpen and move some part of the picture. The computer, sensing the choice or adjustment you have made, can then perform whatever operations you have directed.

Some samples:



The design of screen controls—easy-to-use, clear and simple controls for everything—is one of the frontiers of computer graphics. (See "Fantics," p. DM 74.)

*Just as the parts of icons should often be manipulable, so should the borders of windows. I now give you*

# QFRAMES

The screen-windowing systems that are currently popular are descended from Engelbart's NLS (see p. DM 16), but took their current shape in the Smalltalk environment at Xerox PARC. In these PARC windows, also called "pop-up" windows (though they do not usually come up from anywhere, but simply materialize), the text or graphics are not shown as connected to anything *outside* the window. The contents of the window have no con-

nections and no context. The window usually shows a title for what's in it, and a few other controls, such as a go-away button and "elevators" at the sides, but there is no systematic way of showing interconnectedness. (Macintosh software is built around this type of window.)

I propose a different way of looking at, and thinking about, windows: one that will show **interconnections between separate views**. This means a new style of

frame for the windows. I call these **Qframes** because the frames are designed with various cues about the contents and how they relate.

For instance, when linked text is to be shown (as in the Parallel Textface), it makes sense to indicate **in the frame** what links are available as options, whether for jumps or side-by-side viewing. Here is a Xanadu-class application: the Parallel Textface shown in Qframes. ☛

## The QFRAME A NO-NONSENSE SEMITILED MULTIPANEL DISPLAY TECHNIQUE for linked and related contents

Bush, Vannevar	PUB	NELSON, Theodor H.	PUB
As We May Think		A File Structure for the Com	
the essential feature of the memex. The process of tying two items together is the important thing.		filing arrangements, Bush stressed his file's ability to store related materials in <i>associative trails</i> , lists or chains of documents joined together.	
When the user is building a trail, he names it, inserts the name in his code book, and taps it out on his keyboard. Before him are the two items to be joined, projected onto adjacent viewing positions. At the bottom of each there are a number of blank		"When the user is building a trail, he names it, inserts the name in his code book,	
		n, Theodor H. PUB	
		g It Out of Our System	
		em, and the newcomer hear of them less.	
		ents may be incorporated. system imposes no assumption of completeness or	
		tn	
		PRV	
		HYPERTEXT BOOK: Notes	
		TRAIL STRUCTURE	
		Bush's original "trail" appears to be sequential, but with cross-trails allowed.* I'm not sure whether this maps to any of the newer forms, but we ought to check this.*	
		THE PROBLEM OF DISSENT	
		= Dissents may be incorporated. The system imposes no assumption of completeness or infallibility on its constituent parts. Inconsistencies will	

The borders of text panels may be used to indicate adjacent (or available) linked material, equivalent material, or quotations. Graphic frames may indicate other relations to their neighbors.

from Literary Machines 87.1.



Adjoining linked windows, with cues in their frames like those illustrated, can show such relations neatly. Cues in the frame show the principal xanalogical relations, which are **commonality** and **links** of various types. (See pp. DM 144, DM 50.) Cues in the frames can also indicate what other adjoining frames are available on request.

(Note that this is essentially a version of the Parallel Textface that does not need diagonal lines. Another advantage is that Qframes can be slid sideways or vertically to compress them and make room for other connected Qframes. (Qframes may be called "semitiled.") A number of compressed Qframes may be on the screen at once.)

In what follows I will leave out specific cues to be embedded in the frames (which are arbitrary), and show some of the things that can be done with the Qframe approach.

### MILLER WARS

Another example. While a student at Yale, Mark S. Miller (see p. 108) de-

veloped a multi-user game under UNIX which he called "Star Wars" (an unauthorized use of the name). For this game he developed a unique visualization (recreated below):

The six panels were a three-dimensional cubic view of the entire universe, unfolded to the flat screen. Adversary ships (it was a shoot-em-up) could be seen in perspective, except that they never disappeared, always occupying a minimum of one pixel.

This is another nice example where the Qframe approach is useful. Suppose you were to show less than six panels, or to keep changing the array. Cues in the frames (I have not chosen a set) would indicate the following:

This edge connects to a contiguous view in an adjacent frame

This edge connects to a contiguous view in a non-adjacent frame (as illustrated for edges A and B, C and D, E and F)

This edge connects to a contiguous view not shown but available.

From these two examples—linked documents and Miller Wars—we see three principal attributes of Qframes: that they in some way show the **availability** of adjacent windows, **whether the windows are connected**, and the **relation of the contents** of the windows.

The contents of a window can have various relations to its borders, and these relations, too, ought to be indicated in the Qframes. These include:

This border is at the edge of the space (e.g. text left margin)

There is nothing beyond this border; OR, there is more—and which way it is to be found (orthogonally or radially), and how far

This is a pushing border (e.g. text margin with reformat)

The relations of contents between windows that we've looked at so far were commonality, links, and adjacent visual continuity. (Note that in the Miller Wars case, moving the contents of one panel moved the contents of the others by im-

plication.) Other types of connections between windows you might want to see:

Hierarchy (making it a "browser")

Heredity: instance, partial instance and many more: any relations worth representing.

### OTHER FACTS AND CONTROLS SHOWN

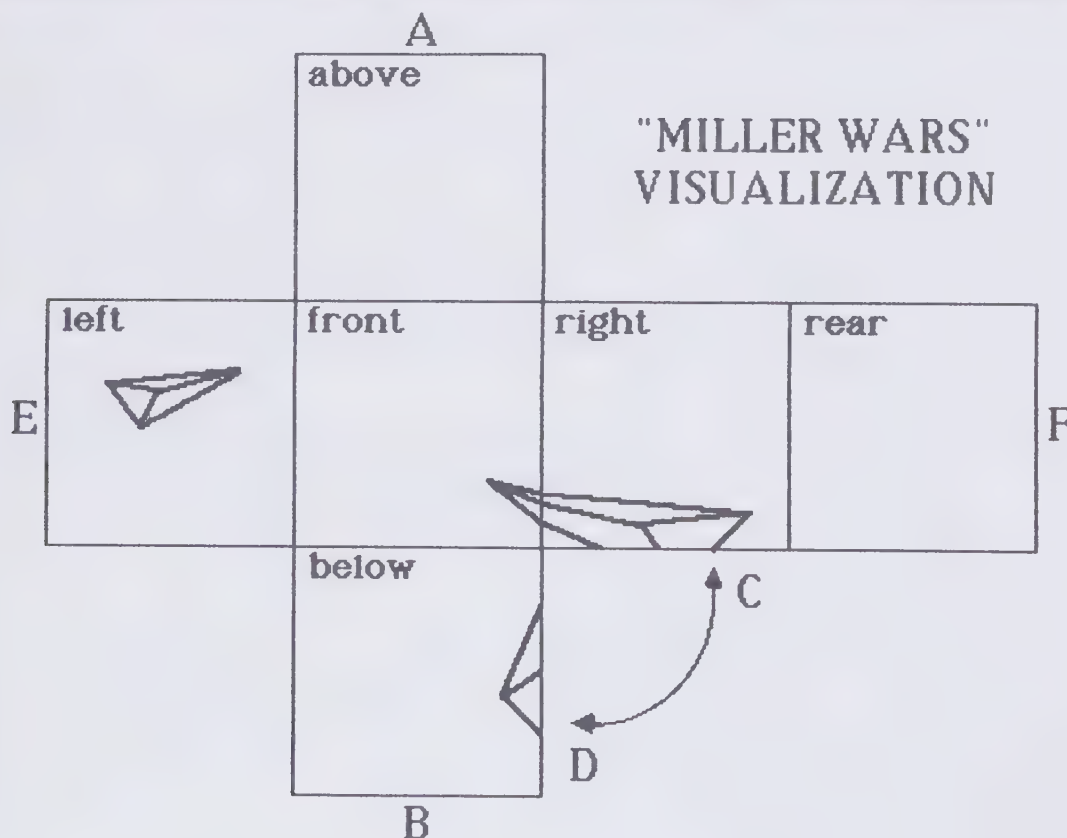
The Qframe should innocuously show the name of an object, its author, its size, its copyright status. (The "Get Info" function of the Macintosh, which tells the origins of a file on request, should clearly be a part of the frame and not a disconnected function on the top menu bar.)

Controls of the contained view should also be in the frame: being able to shrink quickly to an overall view, for example. (Thus the Macintosh "elevators" are an appropriate control to embed in the frame; except that for some mysterious reason, the Macintosh—deeply wedded to paper—does not show **page boundaries** in its elevators, an infuriating omission that violates the Mac's only philosophy.)

### METAVIRT

Qframes are not an example of a "metaphor" (see pp. DM 68, DM 69) because they do not resemble anything else. They are a **virtuality** which needs careful tuning to show just what you want shown.

Qframes can have indications of contents and controls (like Macintosh-type windows), but the philosophical difference is very important: **the Qframe and its responses must be designed as a unified structure**, a virtuality unto itself. (But which of its cues and controls should remain visually present, which to have available under interactive inquiry, and which to animate or palpitate, should be a configurable user choice.)



*This is my design for an interactive system that lets beginners browse through interactive graphics and movies. It is a*

*one-minute system, meaning that the naive beginner should be able to use it within one minute.*

# THE WALKING NET™

©1972, 1974 Ted Nelson

A one-minute system that three-year-olds can learn.

(From an old patent application.)

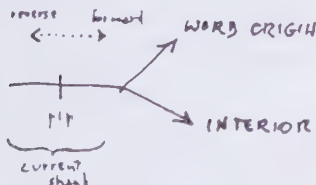
(Sorry to have to show you a writeup, instead of the real thing.)

Another application of the present invention is also in the area of pictorial display, but offers a great variety of potential user choices in a simple circumstance. I call this the "walking net" system because control is effected through a changing network of choices which step, or "walk," around the screen.

The problem of intricate computer graphics may be phrased as follows: given that a digital system can hold a wide variety of graphical materials ready to present, how may the user most simply and conveniently choose them? Indeed, how may the user keep track of what is happening, where he is and where he has been?

The external mechanism I have selected for this facility paradoxically combines great versatility for sophisticated presentations with great simplicity before the naive user. The idea is this: the user may command a continuing succession of changing presentations, making only one simple choice at a time, yet receiving intricate and rich animations with extremely clear continuity on the screen.

The exterior mechanism is this: along with an arbitrary graphic presentation on the screen, the user is continuously presented with the image of a forking set of labelled arrows, e.g.:



The pip is a conventional light-pen cursor. The "current shank" is a line whose implicit gradations control developments in the picture; and the choice of arrows at the end of the current shank

determine a discrete choice between alternatives that are to transpire.

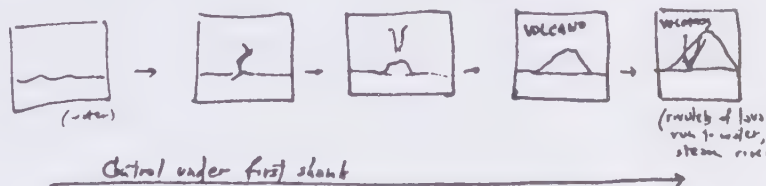
The user, seizing the pip with the lightpen, moves it (through the usual lightpen techniques) sideways along the current shank. Moving it in the "forward" direction causes progressive developments in the picture, moving it "backward" causes a reversal of animations and other previous developments.

When the pip reaches the choice point in the forward direction, the user may drag it (through the usual lightpen techniques) along either of the beckoning alternatives. This then causes further developments in the presentation consonant with the line selected.

"Developments" of the picture here include expansion, contraction, sliding movements and frame-by-frame animation.

(These materials will have been, of course, explicitly input by authors and artists.)

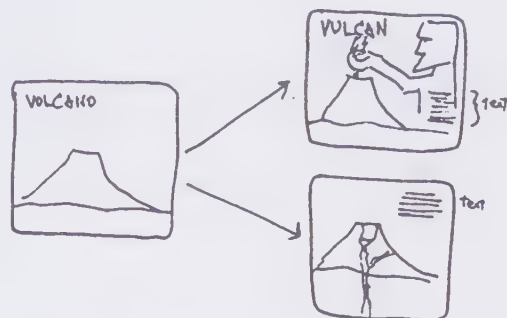
In a sample employment, consider a presentation on the subject of Volcanoes. Let the first shank of the control net control the "rise of a volcano from the sea"—an undulating ocean surface pierced first by a wisp of smoke, then a growing peak, with rivulets of lava seen to run down its sides and darken as they contribute to its growth.



At the end of the first shank, the user may branch to two arrows, labelled respectively WORD ORIGIN and INTERIOR. Either option continues the presentation without a break, retaining much of the picture on the screen. Selection of WORD ORIGIN causes the word VOLCANO to change to VULCAN, and a picture of the god Vulcan is seen to seize

a lightning bolt rising from the crater; text appears to explain this. Alternatively, if the user chooses INTERIOR, the tubes and ducts within the volcano appear, and explanatory text also.

The displayed control net is thus to be understood as a large network of choices, mostly unseen, whose currently visible portion "walks" around the screen as use

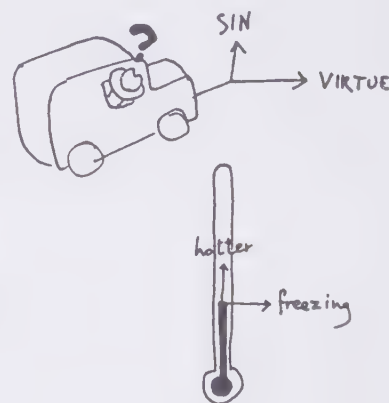


(The path unchosen fades from the screen, as does the previous shank.)

Either of these alternatives may continue with its own developments and animations under control of its own shank.

Several features of this control application are of special interest. One is that the presentation may be continuous in all directions, aiding in continuous user orientation. Another is that presentations are reversible in various ways, an aid both in user orientation and self-study. (Not only is a demonstration reversible within a given shank, but the user may back the pip through an intersection into the ante-

progresses. Within this system, then, numerous variants are possible. For instance, the currently visible portion of the net may itself be whimsically incorporated in a picture, viz.:



But how teach the beginner? How instruct the computer-naive user to get aboard the Walking Net? With **written directions**? NO.

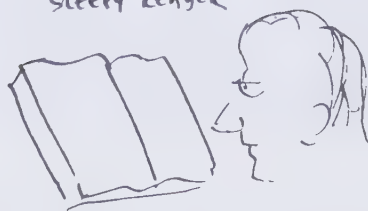
In the following storyboard I went back to first principles. What if the greatest idea-man and director of animation were still here?



OPENING INSTRUCTIONS  
FOR  
"HOW TO USE THE SYSTEM"  
HOW WOULD  
**WALT**  
HAVE DONE IT?

© TM M-107

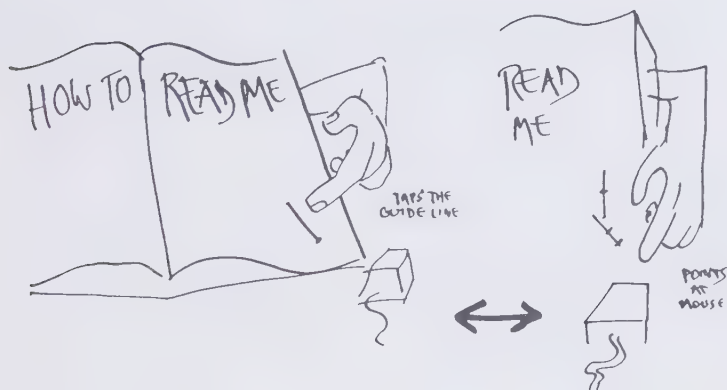
(Old Disney style)  
SLEEPY READER



musical sting —



GO TO RASTER SYNTHESIS  
(HARD-EDGE)



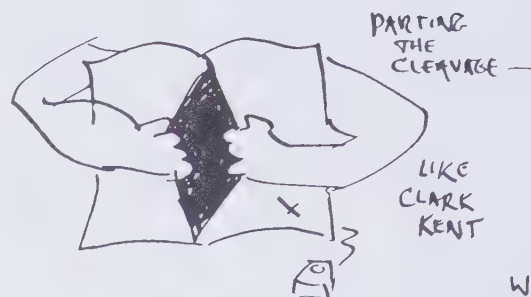
WAIT CYCLE



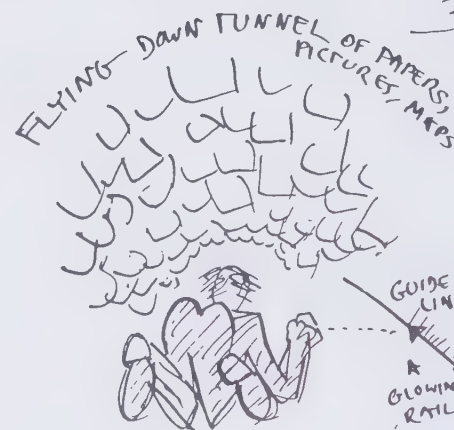
MUSICAL STING — SOUND OF FARAWAY MYSTERY

(when noise is secret —)

YOU ARE NOW  
**IN CONTROL!**  
HANG ON!



WIND,  
ROARING  
SOUND



AND OUT! (CLOUDS)  
THE COSMOS,



AND YOUR CHOICES

# STELLAVISION

A nice example of a unified presentational system would allow you a “feelie” glove along with your computer display—the sort of thing Mike Noll has been doing at Bell Labs.

Now, suppose you are playing with a diagram of a star on a computer display screen. It’s all very well to see its layers, flowing arrows representing convection currents, promontories and so on—but some things you ought to be able to *feel*. For example, the mechanical resonance-properties of stars. It would be nice to be able to reach and *grasp* the star, to *squeeze* it and feel its pulsations as it regains its shape. This could be done in the

glove—at the same time the *image* of the glove grasps the star on the screen, and the star is squished.

Of course, to build such a responding glove, particularly one that gave you subtle feelings back in your fingers, would probably be very expensive. But it’s the kind of possibility people should start considering.

*The intention to build a Feelie setup for exploring molecular models was announced by Glenn Culler at the first SIGGRAPH, while this book was first in press. My understanding is that Culler has since been working on the complex digital engines needed to support that sort of thing, such as the “barrel-roll integrator” that can calculate multiple spatial interactions. Here’s hoping the Feelies are not forgotten.*

# DIMENSIONAL FLIP

3D scopes are about the best we’ve got—so what do we do about multidimensional phenomena?

One very good solution is to show a selection of three dimensions at a time, and provide for easy “flip” from one dimension to another—so that instead of looking at something on dimensions A, B and C you are looking at it on dimensions A, B and X.

For example, suppose you’re a sociologist looking at measurements of various traits among a group of people. It’s a cloud of dots in three dimensions—whatever three dimensions you’re looking at. Some could be: age, height, weight, sex, ethnic background, premarital experience, education...etc.

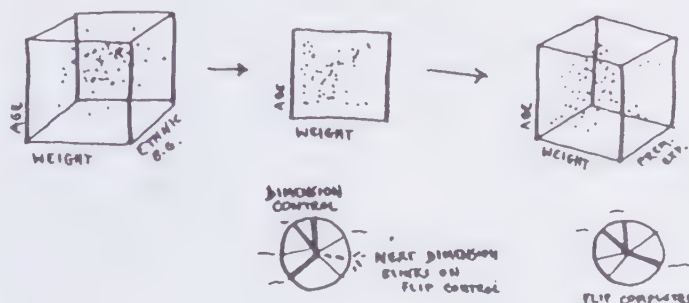
You view this cloud of dots, say, according to age, weight and ethnic background. That means you can rotate it around and see how many people in the group are what.

Using *dimensional flip*, however, you can change the view as follows: rotate the box-frame till it becomes a square to your eye. Then you hit the control that makes the unseen dimension “flip” to another dimension that interests you. The cloud still looks the same—until you rotate it, and the third dimension is now “prema-

tal experience.” So you can quickly get a view of how populations are really divided up. (Note to sociologists: this same operation, with stretching and clipping, provides a visual technique for “partialing” operations of the Lazarsfeld type.)

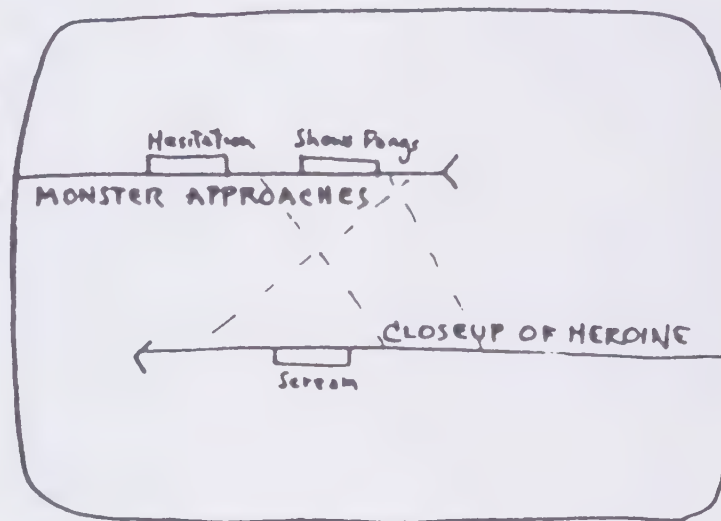
**FINALLY!**

*Someone has implemented a nice dimensional-flip program for the Macintosh. It’s called MacSpin, a “graphical data analysis” program from D<sup>2</sup> Software, Austin TX.*



# CINENYM

In a system I designed for CRT motion-picture editing, the user could manipulate written descriptions on the screen (corresponding to the usual yellow-pad notes). To see the consequences of a particular splice, for instance, the editor would only have to draw a line between two annotated lines representing shots. Trim variations could be seen by moving this cut-line (illustrated).



CINENYM™ © 1968 T. Nelson

Not long after, CBS and Memorex did introduce a system for movie-editing by CRT—but I’ve heard that in their system the user has to actually deal with numbers. If so, this is missing the whole point.

*This picture illustrates a fundamental principle: distilling an interactive system to its fundamental conceptual structure, and stripping away, stylistically, the conspicuous display of other technicalities (such as footage counts and frame numbers).*

*Note that this approach is very different in style and feel from that taken by other designers of computer-based movie editing systems.*

Livermore Labs, those hydrogen-bomb design people, will have a “Laboratory for Data Analysis,” an opulent facility for experimenting with multidimensional visualization.

One of your jolly ironies.

I have seen pictures of beautiful multibutton control handles which were designed for project SMASH, would you believe Southeast [Asia] Multisensory Armament System for Helicopters. Aargh.

The best with the worst.

Everything is deeply intertwined.



# THINKERTOYS

Our greatest problems involve thinking and the visualization of complexity.

By "Thinkertoy" I mean, first of all, a system to help people think. ("Toy" means it should be easy and fun to use.) This is the same general idea for which Engelbart, for instance, uses the term "augmentation of intellect."

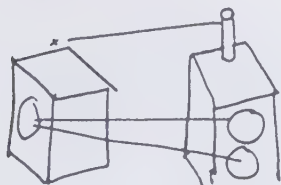
But a Thinkertoy is something quite specific: I define it as a computer display system that *helps you envision complex alternatives*.

The process of envisioning complex alternatives is by no means the only important form of human thought; but it is essential to making decisions, designing, planning, writing, weighing alternate theories, considering alternate forms of legislation, doing scholarly research, and so on. It is also complicated enough that, in solving it, we may solve simpler problems as well.

We will stress here some of the uses of these systems for handling *text*, partly because I think these are rather interesting, and partly because the complexity and subtlety of this problem has got to be better understood: the written word is nothing less than the tracks left by the mind, and so we are really talking about screen systems for handling ideas, in all their complexity.

Numerous types of complex things have to be inter-compared, and their relations inter-comprehended. Here are a few of the many types:

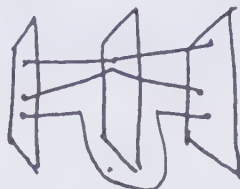
Alternative designs.



Discrepancies among the testimony of witnesses.



Successive drafts of the same document.



Pairs of things which have some parts the same, some parts different (contracts, holy books, statutes of different states, draft versions of legislation...).



Different theories and their ties to particular examples and evidences.



Under examination these different types of intercomparison seem to be rather different. Now, one approach would be to create a different data structure and viewing technique for *each* different type of complex. There may be reasons for doing that in the future.

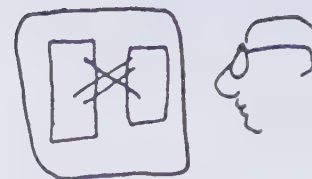
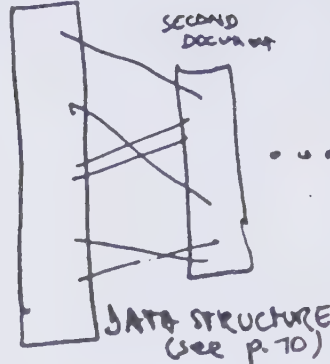
For the present, however, it makes sense to try to find the most general possible viewing technique: one that will allow complex intercomparisons of all the types mentioned, and any others we might run across.

One such technique is what I now call *collateralation*, or the linking of materials into *collateral structures*,\* as will be explained. This is fairly straightforward if you think enough about the problem; Engelbart discovered it independently.

Let us call two structures *collateral* if there are links between them, connecting a selected part of one with a selected part of the other. The sequences of the connected parts may be different. For simplicity's sake, suppose each one is a short piece of writing. (We will also assume that there is some convenient form of rapid viewing and following between one end of a link and another.)

\* In my 1965 paper (see bibliography) I called collateral structures *zippered lists*.

FIRST DOCUMENT



Now, it will be noted first off that this is an extremely general method. By collateral structuring we can easily handle the equivalents of: tables of contents; indexes; comments and marginalia; explanations, exegesis, explication; labeling; headings; footnotes; notes by the writer to himself; comments and questions by the reader for later reference; and additional details out of sequence.

*Collateralation*, then, is the creation of such multiple and viewable links BETWEEN ANY TWO DATA STRUCTURES, in principle. It is general and powerful enough to handle a great variety of possible uses in human intellectual endeavor, and deserves considerable attention from researchers of every stripe.

The problem, then, is how to handle this for rapid and convenient viewing and whatever other work the user wants to do—writing and splicing, intercomparing, annotating and so on. One solution appears on p. DM 41: The Parallel Text-face™, designed as a seminal part of the Xanadu system (see p. DM 141), which I hope will be marketed with that system in the near future.

## CLARITY AND POWER

We stressed on the other side of the book that computer systems must be clear, simple and easy to use. Where things like business uses of computers are concerned, which are intrinsically so simple in principle, some of the complications that people have been forced to deal with in ill-designed computer systems verge on the criminal. (But some computer people want others to think that's the way it has to be. "Your first duty is to keep your job, right?" one computer person said to me recently. "It wouldn't do to

set up systems so easy to use that the company wouldn't need you anymore." See "Cybercrud," p. 27.)

But if it is desirable that computer systems for *simple-minded* purposes be easy to use, it is *absolutely necessary* that computer systems for *complicated purposes* be simple to use. If you are wrangling over complex alternatives—say, in chess, or in a political simulation game (see "Simulation," p. 149), or in the throes of trying to write a novel, the last thing you will tolerate is for your computer screen to introduce complications of its own. If a system for thinking doesn't make thinking simpler—allowing you to see farther and more deeply—it is useless, to use only the polite term.

But systems can be both powerful and simple at the same time. The myth that things have to be complicated to do anything for you is pernicious rubbish. Well-designed systems can make our mental tasks lighter and our achievements come faster.

It is for this reason that I commend to the reader these two designs of mine: as examples of user-level control and viewing designs—fanciful environments, if you will (see p. DM 74)—that are pruned and tuned to give the user great control over the viewing and cross-consideration of intricate alternatives, *without* complication. I like to believe that both of these, indeed, are *ten-minute systems*—that is, when we get them running, the range of uses shown here can be taught to naive users in *ten minutes or less*.

## BIBLIOGRAPHY

Theodor H. Nelson, "A File Structure for the Complex, the Changing and the Indeterminate." Proc. ACM 65, 84-100.

# NEW SUPER VIRTUALITIES

*I believe that interaction with computers can be at least ten times easier, ten times more powerful, ten times more vivid; and that these are issues not of hardware but of virtuality design.*

*Just around the corner (or perhaps around the block) are a collection of new large virtualities—whole interactive systems—with much more power and ease than we've seen before. (Not as much as I'm predicting, just yet, but big jumps.)*

*Of course, these require that we make the BIG TRANSITION: letting go of paper as an ideal and as a security blanket, and realizing that screen-systems, without paper, will and must be the home of the mind in the future.*

*Some are:*

*GUIDE from Owl, International, Inc. (Seattle WA), a hypertext system which may be used also for word processing, outline processing, filing, and the sending of documents.*

*Hypercard from Apple, which promises hypertext capabilities and wild new programming features. Finally, my own ZigZag™ integrated-software construction set (not yet implemented or revealed publicly), which takes a startling and implausible visual idea and extends it in half a dozen different directions—to perform the functions we now call word processing, outline processing, spreadsheet, database, filing, scheduling, programming and animation—and knits them together into a unified, easy-to-learn, high-power environment. I hope.*

*This is obviously the place to tell you about Alan Kay and the Dynabook.*

*The hottest project at Xerox PARC is Alan Kay's Dynabook, formerly the Kiddy Computer. As lots of people will tell you, it's going to cost five hundred dollars, be small enough to carry around on a shoulder strap, have a built-in screen, run on batteries, and have all the books a kid wants to read from the screen stored on a cassette.*

*And the demos! They'll knock you out. On a color TV screen, they'll show you a wildly changing pageant of toy sol-*

*diers, photographs, beautiful patterns, all generated by the computer in real time (see "Bit Maps," p. DM 92). And if you're into computers, they'll show you how all this is run by the beautiful Small-talk language (it was previously called the Kiddy Computer, remember), which any bright child can learn and which has some awfully powerful features.*

*Alan has left Xerox PARC. He still talks about Dynabook, but there is evidently no rush about it.*

*It's not clear what Dynabook is really going to be. Part of the idea was always a megabyte portable computer, so **that** part is taken care of; another part of the idea was ease of programming, and that's still open.*

*Alan talks about many ideas for the Dynabook, including canned programs that users could easily modify, and "view-point" languages as a different way of letting users make their own versions of things. Lately he's talking about "agents." Thus the Dynabook is still under dynamic definition.*

*Alan Kay has a license to diddle. (He is an Apple Fellow.) Because of his great achievements with the Alto and the Small-talk language, which he created, and object-oriented programming (which he did not invent but **elucidated**, since its inventors, Sutherland and Nygaard, did not know what they had invented), Alan may if he wishes spend the rest of his life conference-hopping and making sardonic comments. But it is clear that he wants to do much more. He is presently working with flight-simulator-class equipment in Los Angeles public schools.*

*I'm betting on Dynabook as one of the great virtualities of the future. But only after Alan decides what it is.*

*A LAST THOUGHT TO PONDER: the explorable space that does things just before you get there.*

*You will have to do no programming, and indeed "programs" will never be explicitly invoked at all; they will simply take effect as you *get near*, in the display space, something which needs update. A display-driven information complex.*

# MOVIES

*And Other Experiential Media*

*A movie is something you look at. It's a one-way medium: you don't control it much, although you can watch different parts of the screen. Usually it communicates stories and feelings, rather than complex facts and relationships.*

*Movies are, indeed, a good model for all the experiential media. What is music, but a movie without the visual part, the dialog and the sound effects? What is a painting but a freeze-frame with no sound track? What is writing but the voice-over without the video and sound effects?*

*What I am getting at is this. In music, movies, painting and writing, someone has gone to a lot of trouble to arrange the experience of the audience. The movie combines more different modalities; these sum to a more high-bandwidth experience.*

*Computer graphics and audio (to be discussed soon) give us a whole new palette of techniques for tomorrow's media. There are many new techniques of movie-making to understand, which is one purpose of this book. Today you can buy a movie studio in a desk, allowing you to create whole three-dimensional worlds in living color and photograph them. But it is still the human experience that is being unfolded.*

*It is the artistry and intelligence, the storytelling and presentation, that count above all. You can still be an artist in movies, and indeed develop new techniques, without going near a computer, as have Jim Henson in terry cloth and Will Vinton in clay.*

*It is from the study of existing movie artistry, however, that we will best learn to use the new graphic techniques.*

*(I will even argue later that movies are the best model for the new interactive hypermedia.)*

## "THE BIG SCREEN"

*35mm was Edison's original standard. 70mm arrived in the thirties, but only caught on in the fifties, and has remained*



# COMs

## (Computer Output Microfilm devices)

are what you use to make computer movies. Basically, they consist of a CRT and a movie camera in a box.

Mostly they are used to put text on microfilm by computer, so generally they are not connected to a computer but run off magnetic tape.

This turns out to be very annoying if you want to hook up the computer directly to the COM, and make movies that fill the frames spot-by-spot. For that, you really need your own movie camera and a minicomputer. (Movie cameras that can be made to start and stop by computer are called "pulse cameras" or "instrumentation cameras.")

(NOTE: for them as want to make color movies, the two alternatives have been either to have separate primary negatives combined at a lab—the "old Technicolor" process—or to add a complicated color-filter box to a COM or other CRT setup. Such things are available commercially now.

About 1972, there was announced an electronically-controlled color filter that could change to any hue in nanoseconds. That would be just what we all need for color movies from COMs—but what happened to it?

The price of COMs is down considerably now, but still in the tens of thousands. It's not the sort of thing you should have to buy; making the actual movie from your data should be a service.

## Movies From Your Computer

*Not only will we soon be able to make movies on our personal computers, but the resolution of these movies will be unlimited by our personal equipment.*

*This is perhaps hard for the beginner to understand.*

*It is the resolution of the output device that slows us down. Let service firms own the fancy output devices, and let's you and I send off our diskettes to be rendered on their high-quality COMs.*

The prospect surprised them, but MAGI (see p. DM 106) allows as how they might let you make movies on their over-the-phone movie-making setup (sketched on p. DM 108). Price to capable outsiders, if the software meshed, would be \$50 an hour. (Six hours makes one minute of film, not counting the phone bill. Cheap if you know movie nomics.)

*No, not yet, it seems; nobody yet offers film-making services to the small user. Mainly that's myopia: nobody realizes yet that processing over-the-phone computer data into videodiscs and movies will be an enormous consumer service, comparable to the film-developing industry. It should be only a couple of years more till this opens up.*

the standard (except for a brief fling with 105mm, on which John Huston's "The Bible" was shot). (Cinerama, a spectacular process of the fifties, used three 35mm projectors, and there were terrible edge effects between their respective projection areas.

Now, for special installations, there are Cinemax and Omnimax, which use, respectively, superwide film projected flat and hemispherically.

But, as always, there are super-hi-res systems in prototype that producers would love to use. Getting payback from these is a problem, however, for the developers and manufacturers, especially since there's no way any foreseeable market could support a much more lavish movie standard.

Meanwhile, John Lowry, at Digital Video Laboratories in Toronto, has been developing high-quality video suitable for transfer to theatrical film. He and they have developed a 655-line color system—with heavy digital enhancement (see "Picture Processing," p. DM 81). I scarcely believe my notes, but I saw it, and wrote down that it was comparable to 35mm studio color. The day of "electronic cameras"—that is, film-quality video—may be upon us soon.

In fact, feature films have been shot in video and transferred to film; this saves a lot of money. "Crack in the Mirror" (1987) was ballyhooed as the first feature film to be shot in video, but I've heard there were earlier ones.

Video technology can also give us electronic still photographs without film (now available in Japan, at least, from Canon).

## COMPUTER MOVIES

How do computers make movies? Well, first of all, computers do not make movies unless thoroughly provoked.

In fact, only people make movies. But computers, if sufficiently provoked, will do a lot of it: enact the movie and photograph it, frame by frame.

There is no single method.

All forms of computer display and computer graphics may be used to make computer movies.

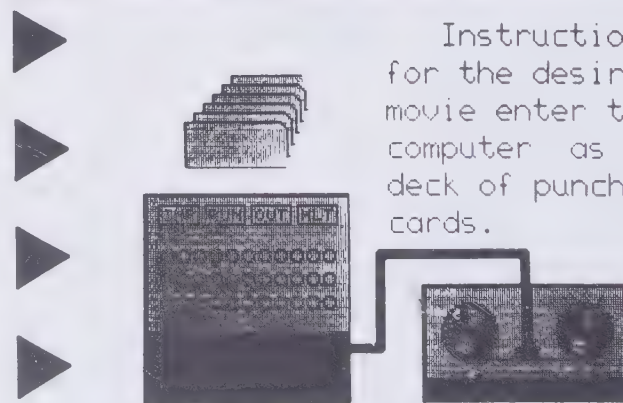
"Computer animation" is any method of making movies in which a computer successively draws or paints the successive individual frames, which may be done by any of the methods mentioned in this book. Now, since there are numerous methods of making pictures by computer, any method of making *different* individual pictures, in a succession of changing frames, is computer animation. So a "computer movie" is any film made by, or with the picture-making aid of, computers.

In other words—it's no one thing.

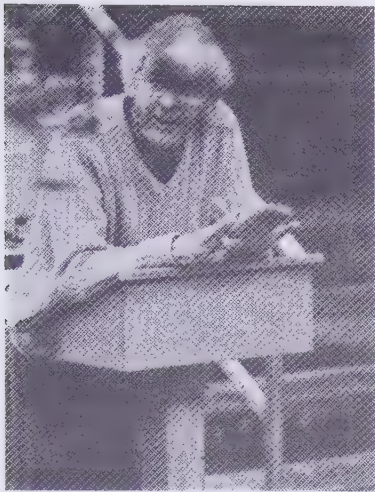
Now, there already exist hundreds, if not thousands, of computer movies. So far most of them have been on technical topics—the mechanics of satellite orbit stabilization, the mechanics of explosions, and so on.

(As of 1987, the largest quantity of computer-generated footage is probably big-letter station breaks and commercials.)

Here are a few stills from some other movies, more humanistic and various classic artists and early practitioners to know about.



Vintage Knowlton, using BEFLIX. (This language used the COM quite efficiently: dots were actually out-of-focus letters.)



courtesy of John Whitney

## JOHN WHITNEY

John Whitney is the ancestor of us all,\* probably the first computer movie-maker. He is also a gripping speaker.

In the forties, he built a special animation stand, using *analog* computers.

Deeply concerned with music, Whitney has in his images emphasized rhythmic and contrapuntal movement of shapes and lines.

*John invented the motion control camera. Thus all the rocket ship effects in*

\*Technically, he is only an ancestor(1) of John Whitney, Jr., a leading figure in computer image synthesis. See p. DM 100.

modern movies are due to his work except the rocket ships in "The Last Starfighter," which are due to his son's work.

He also invented the slit-scanner, a mechanism which produced the "trip" effects in "2001."

In his own movie-making John's concern and uses for these mechanisms has been the creation of abstract fountains and mirror-chambers of a strange beauty which soothes while it requires your undivided attention.

## WHERE ARE THEY NOW?

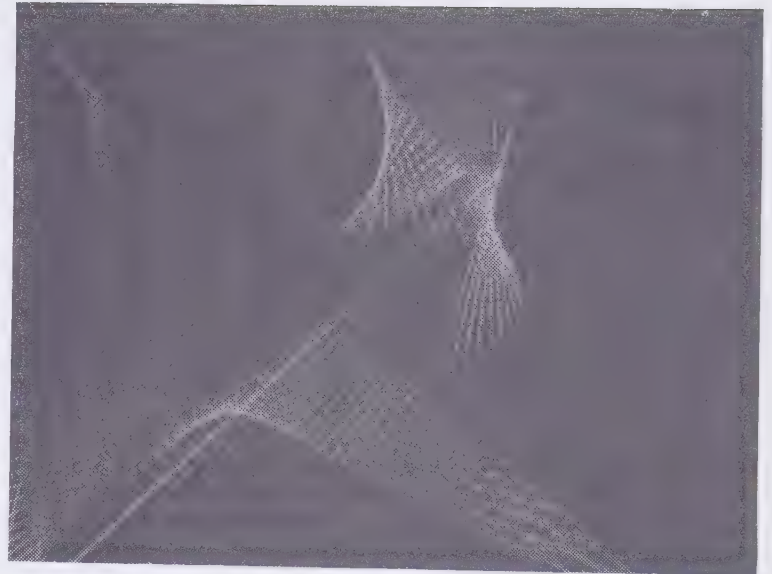
Whitney is finishing a music-and-graphics editor "like a word processor" for the simultaneous editing of music and his own style of graphics (harmonic field of colors and points). It runs on the AT and will probably be commercially available.

## BIBLIOGRAPHY

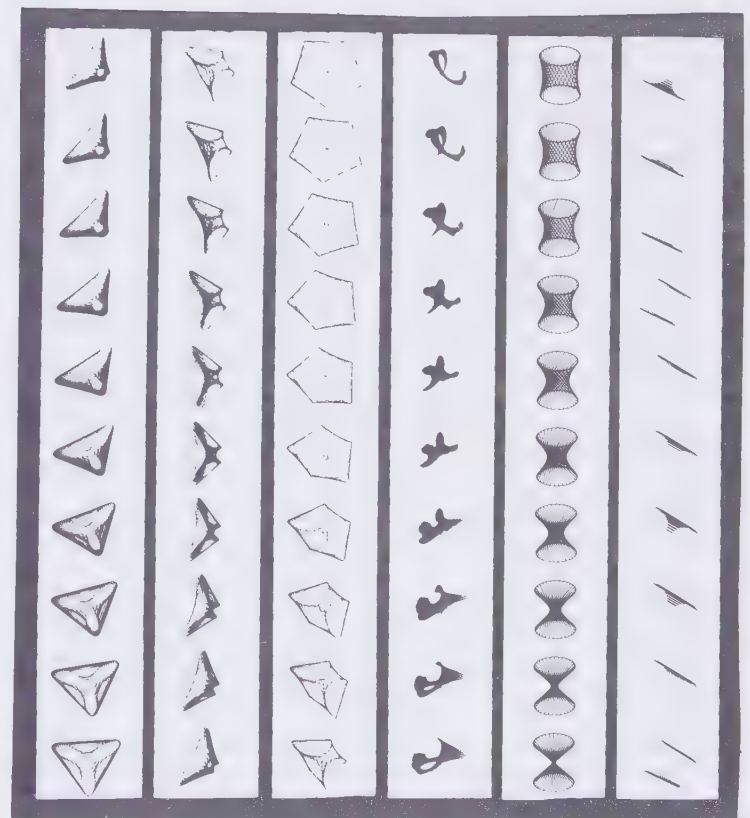
*John Whitney, Digital Harmony: On The Complementarity of Music and Visual Art, McGraw 1981. (Has programs that run on the Apple II.)*

Whitney films available from: Pyramid Films, Box 1048, Santa Monica, CA 90406.

A laserdisc will be issued in 1987 with all of John Whitney's films on it. (Contact: Laserdisc Corporation of America, Montvale, NJ.)



Courtesy of John Whitney



Courtesy of John Whitney

**N**OW IT CAN BE TOLD: John Whitney did the original Jack-in-the-Box logo. It hangs on his wall.



Kenneth Knowlton is a Bell Labs lifer. Tall, patrician and gracious, his work, like Sutherland's, shows the inner light of unifying intelligence. He works in Max Mathews' section of Bell Labs at Murray Hill, where they do all that interesting stuff with music and perceptual psychology and so on. During the last decade, Knowlton has turned out vast quantities of articles, processed pictures, movies, and actual computer languages; while any ordinary man would be satisfied to be so productive, apparently he does a lot of other things in his work that he doesn't talk about.

Some of Knowlton's best-known work has been in picture processing, where he has converted photographs into mosaics of tiny patterns—which nevertheless show the original.

His first widely-known language was BEFLIX (Bell Labs' movie-making system); this was programmed for the 7094 in the early sixties. BEFLIX allowed the user to create motion pictures by a clever mosaic process that used the output camera more efficiently. (Actually, the lens was thrown out of focus manually and the entire frame created as a mosaic of *alphabetical characters*; this did the whole thing much more quickly and inexpensively.)

(Some of the clever data-handling techniques of BEFLIX Knowlton then turned around and used in L6, a language which made these techniques available to other computer people. This may sound like only a computer technicality, but it's the sort of thing that's widely appreciated. (L6 stands for "Bell Labs' Lower-Level List Language.")

Wanting to get outside artists interested in BEFLIX and related media, he worked for a time with film-maker Stan Vanderbeek; from this Knowlton saw that artists' needs were more intricate than he had anticipated. Augmenting BEFLIX with some of the things Vanderbeek asked for, Knowlton came up with a new language called TARPS (Two-Dimensional Alpha-Numeric Raster Picture System). This in turn led to EXPLOR (EXPLICITly provided 2D Patterns, Local neighborhood Operations, and Randomness). EXPLOR is fascinating because of its originality and generality—not only does it modify pictures and serve as an artist's tool, but it has fascinating properties as a computer language and may even have applications in complex simulations for technical purposes.

Since Vanderbeek, Knowlton has entered into a long and fruitful collaboration with Lillian Schwartz, a talented artist. Their many films have been clever, startling and powerful. I must say that they grow on you: I liked them at first, but when I saw five or six in a row this January, I found them just incredible. Because they are abstract, and full of fast-changing patterns and reversals, they take some adjusting to; but they're worth seeing over and over.

EXPLOR may be thought of as a highly generalized version of Conway's Game of Life (see p. 115). You start with two-dimensional patterns as your data structure; these can be abstractions or even converted photographs, as in a recent Knowlton-Schwartz film showing Muybridge's Running Man. In your EXPLOR program, you may then cause the pattern to change by degrees, each cell of the pattern reacting to the cells around it

or to random events as specified by the programmer.

EXPLOR, running without external data, comes up with some extraordinary snakeskin and Jack Frost patterns. But its uses in traffic simulation and various other studies of populations in space could be very interesting.

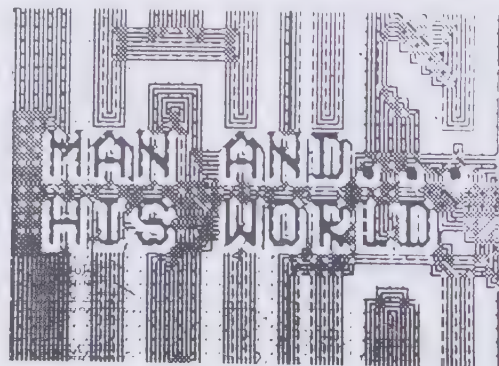
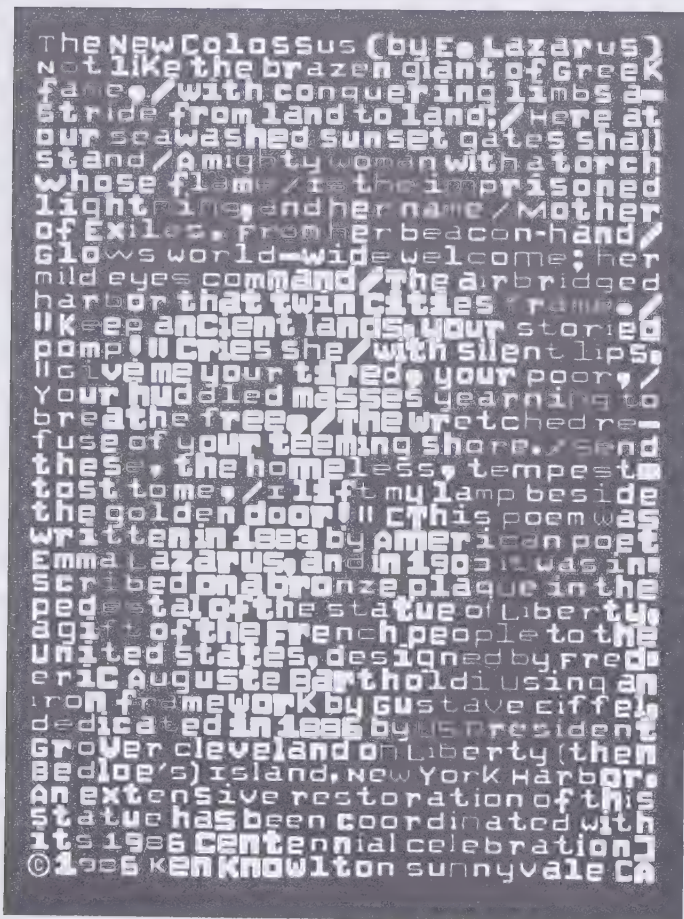
EXPLOR has obvious artistic applications. Lillian Schwartz is using it extensively in film-making. It's now running on a minicomputer feeding to a modified Sony Trinitron color TV. (This color setup was created by Mike Noll and is described in a recent issue of the CACM, though only for black-and-white TV; the color is more recent. It stores the color picture as a list of sequential colors represented in the computer's core memory, each dot being represented. Cf. "Boyell's Terrarium," p. DM 111.)

Knowlton has used EXPLOR for teaching computer art at the University of California; the language is available programmed in "medium size" FORTRAN from Harry Huskey, Dept. of Information and Computer Science, U. of Cal. at Santa Cruz, Santa Cruz, California.

#### WHERE ARE THEY NOW?

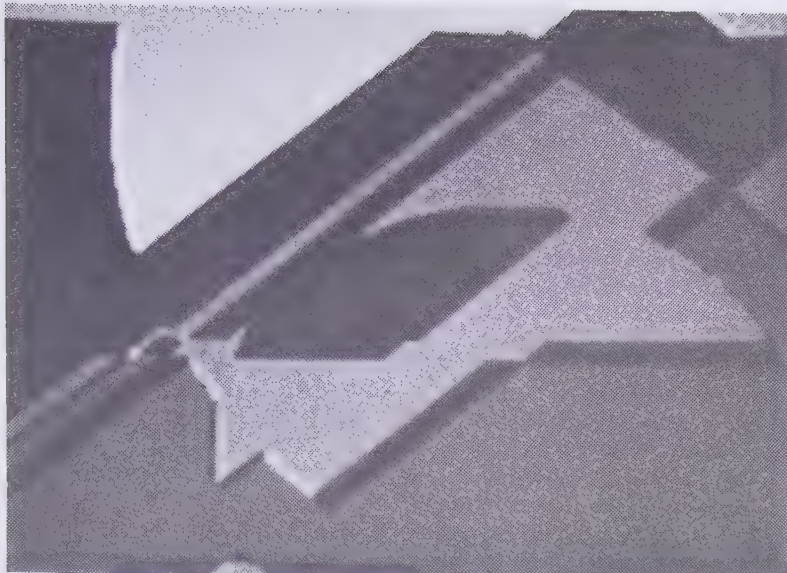
*Ken Knowlton has moved around quite a bit in the last decade, as artist, teacher and consultant.*

*His artwork has been specially interesting in his mapping of odd picturelets to photographs. His obsession is the relative darkness of arbitrary spots. He uses computer modeling to fit a given array of dots to a picture (he for a time sold plans for the arrangement of dominoes to depict Charlie Chaplin and Einstein, the arrangement having been worked out by his computer program to optimize the use of a whole box of dominoes).*



Vanderbeek & Knowlton





## BURTNYK, WEIN AND PETER FOLDES

Millions of people saw computer graphics for the first time on the PBS "Ascent of Man" series, where a screen drawing of Early Man's skull was seen to rotate and gradually change in its features. This was startling even if you know about computer graphics, since it seemed to be proceeding from complex data concerning the entire skulls and their changes.

Not so. Actually, what you saw was a series of skull drawings by Peter Foldes, a Parisian artist, with the computer generating transitional drawings between them. (Indeed, though you saw Prof. Brodowski next to the screen, you did not see him next to the screen at the same time

the drawings were changing—because that had to be filmed very slowly.)

The system was created by Nestor Burtnyk and Marcelli Wein, of the National Research Council of Canada. It currently runs only on a SEL 840A. (It was also used by the National Film Board of Canada for creating Foldes' splendid film "Hunger.") They can preview by rolling through bit-map video on a moving-head disk. (See Burtnyk and Wein, "Computer-Generated Key-Frame Animation." *Journal of the Society of Motion Picture and Television Engineers*, March '71, pp. 149-53).

*Note: Burtnyk and Wein's work illustrates the important computer-animation technique of key-frames: manually drawing certain frames, and having the computer draw the frames in between.*

Lillian Schwartz

## LILLIAN SCHWARTZ

A talented artist with a feel for technology, Ms. Schwartz has been working for several years with Knowlton and others at Bell Labs. Her films with Knowlton, mentioned elsewhere, are marvelous. She now works at a more permanent setup, a minicomputer that runs successive images on a color TV screen, employing a modified form of Knowlton's EXPLOR Language. The work is immediately viewable. This allows rapid film construction, not previously possible when the work had to go through a slow animation camera before she could see the result.

For Knowlton & Schwartz films, contact: Martin Duffy, AT & T, 195 Broadway, NY, NY.

No longer at Bell Labs, Ms. Schwartz hit the news recently for lining up a da Vinci self-portrait with the Mona Lisa, by computer, and concluding that he was his own model for the painting. "Any two photographs can be lined up," says a critic. But there's a question of how much you have to push the parts around, and this piece of research remains intriguing.

Speaking of pushing parts of faces around, that brings us to

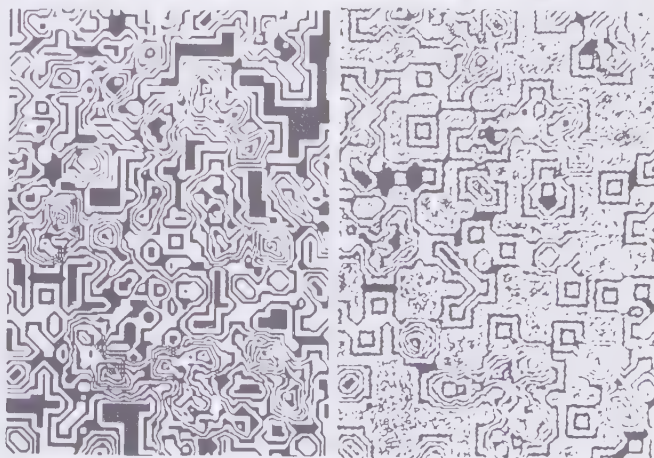
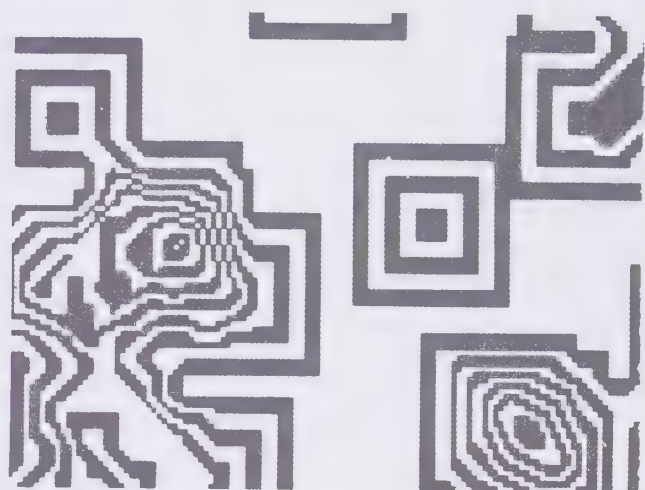
## NANCY BURSON

Nancy Burson is a New York artist whose company, Face Systems, uses a VAX to make *averaged faces* out of different pictures. That is, features are apparently set in correspondence, rearranged (according to various weighting-functions) to make all the features line up, and then greyed-in to average the pictures.

She has a haunting book of these out, featuring composite pictures of, for instance, great beauties of the eighties, or a single face fusing the images of many members of one family.



Lillian Schwartz  
(with Henry Magnuski, see p. DM 105.)



Schwartz & Knowlton. Using the EXPLOR language, they make pictures and patterns scintillate and grow together. (EXPLOR in some ways generalizes Conway's Game of Life; see p. 115 and p. DM 81.)



## RON BAECKER'S GENESYS

By now there are dozens of computer animation languages, perhaps hundreds. Each one employs the techniques of animation which its developer wanted to use, tied together in the ways that seemed appropriate to him. (See "Computer Languages," p. 56, and note Knowlton's various animation languages, described on p. DM 54.)

One of the more influential animation systems has been Ron Baecker's GENESYS, a 2-dimensional animation system programmed in the late sixties at MIT's high-security Lincoln Laboratory. (It used the TX-2 computer, mentioned elsewhere in this book.)

Baecker, a cheery and genial fellow, expressed interest as a student in using the TX-2 for animation, and was allowed to. The system he produced has a number of lessons for us all.

GENESYS is a "Good-Guy" system, as discussed on p. DM 10. Meaning, in this case, that it is easy to learn and simple to use. As argued elsewhere in this book, making computer systems clear and simple is often hard for the programmer (and may go against his grain), but is essential.

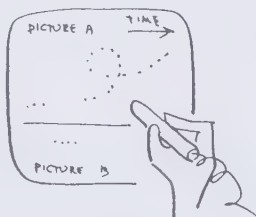
### PICTURES AND MOTIONS

GENESYS makes the following simplifications of your movie: all images are made up of dots. They do not change as you watch; animation consists of the images either *moving* or being *replaced*.

To create an image, you draw it onto the screen with a lightpen or a tablet. (As in the SKETCHPAD system; see p. DM 96.) Parts of the image may be changed until you're satisfied.



Now, to create the animation, you do the same thing. Each image can be made to move on the screen; and the path of the motion may be drawn on the screen, through the picture area. Not only that, but the *timing* of the motion is controlled through the same diagram, by the spacing of the dots. (Baecker calls his control diagrams *p-curves*.)



Lastly, sections of picture may be replaced by means of the control diagram (as indicated in picture above).

Having created such an animated sequence, which is stored in symbolic form in the computer ("digitally"), you can view it on the screen, decide what you do and don't like about it, and change any part of it.

The basic elegance of the system is this: Baecker made everything work the same way, through control by screen diagrams. He simplified the animation problem in a clear and simple way.

*Note: With GENESYS, Ron invented the technique of the **timing graph** (he called it a *p-curve*), a drawing whereby you could control the motions and super-session of picture parts.*

Programs like GENESYS, called *Animators*, allow users to make animated cartoons for the computer screen. (The first one ever may have been GENESYS; the first one on the personal market was *Movie Maker* from IPS, programmed by Eric Podietz and Guy Nouri.) These let you make movies by moving shapes around in a spritely fashion and replacing them like animation frames.

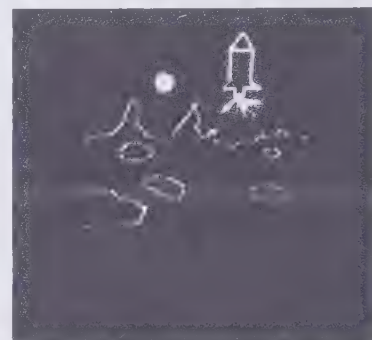
Currently none of these, as far as I know, can be made interactive (to branch among different animated sequences depending on your choices), or will change only part of the picture as part of the branching, which you really need. (For instance, for proper animation of such things as the *Walking Net*™—see p. DM 47). But these will come.

## LYNN SMITH

Lynn Smith is a young Boston artist who has worked extensively with Baecker's GENESYS (see at left). One result has been a movie which should be an example to us all: "The Wedding Movie for Bob and Judy." (Her friends Bob and Judy were getting married, so she made this movie, a few minutes long and quite clever, to celebrate it.)

This is my favorite example of how computers should be used in the human world; it says more on the subject than any dozen articles.

(One question that remains unanswered is how a system like GENESYS could have been used for such a purpose, seeing that most people in the field believe GENESYS only runs on the heavily-guarded TX-2 computer. Regretfully, I can shed no light on this here.)



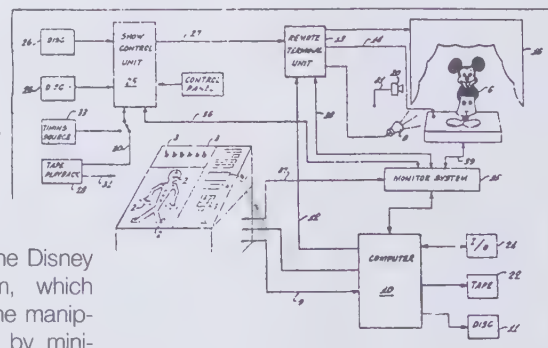
## Super Puppets

Jointed dolls clad in foam rubber have been an important special-effects medium ever since Willis O'Brien in the thirties (*Lost World*, *King Kong*, *Son of Kong*). In the fifties they added remote control by cable and pneumatics. In the eighties they added remote control by computer, giving us *E.T.*

## Traditional Cartoons By Computer

Various firms have developed systems for doing old-fashioned movie cartoons by computer, allowing a human artist to do key frames and then in-betweening automatically. These include *Hanna-Barbera in Hollywood*. A system called *Antics*, from England, offers similar capabilities.

Patent 3,767,901 is for the Disney Audio-Animatronics system, which now basically consists of the manipulation of rubber puppets by mini-computer, through cables and puffs of air.





## VIDEO. The Happy Medium? some mutterings


Would you believe there was television broadcasting over the airwaves in the nineteen-twenties? The thing is, it used bizarre spinning equipment because there were no CRTs (see "Lightning in a Bottle," p. DM 84). Only with the development of radar in World War II did there also come a practicable Cathode Ray Tube, making home television feasible.

But the big companies were at first very conservative in their marketing, figuring television would be a luxury item only. It took a man named Madman Muntz, who caricatured himself in a Napoleon hat, to see that millions would buy television if the price was right. So he came out with Muntz TV in the late forties. As I recall, the Muntz TV cost \$100 and had *one tuning knob*. (This was less

intimidating than the row of knobs on more expensive sets.) I don't know how Muntz came out on it all, but his opening of the mass market made the bigger corporations realize it was there. (This same thing may yet happen again in newer media.)

Originally all there was was Krazy Kat and Farmer Brown cartoons. But behold, sooner than you could say "vertical hold," there were Sid Caesar and Imogene Coca on the Admiral Show, and we were off.

A quarter of a century later, the best of television is no better and the bulk of television is about as bad as it ever was.

We "understand" television. That is, we know what a TV show is, how it fits together and so on. 

*My position on television is that television is movies that follow a slotted schedule. The time-slot decrees the nature of television just as surely as the time-slot decrees the nature of schooling in our current system.*

*A few years ago someone came up with the hypothesis that TV is a drug. Supposedly you get the same brain-wave patterns when you turn on the tube as when you take heroin. And turning it on sure is addictive. Everything is set up to make you think what you watch is important, but if you go for a few weeks without it you find you didn't miss much.*



But what people don't realize about TV is that the governing feature is the *time-slot*. In any medium with time-slots, whether TV, radio or classroom education, the time-slot rules behavior. Whatever can happen is as constrained as ice cubes in a tray.

This is the limiting factor when optimists try to use TV for teaching. If it's coming over a cable, everything has to be scheduled around it, and the contents are clipped and constrained to fit the time-slot. It may be better with videotape.

## CABLES

In the last dozen years, Cable TV, or CATV, has become big business. A Video Cable is a high-capacity electrical carrier that runs through a given neighborhood or region. Business and individuals may "subscribe" and get their own sets hooked onto the cable.

What this does first of all is improve reception. The fouled-up video picture caused by such extraneous objects as the World Trade Center in New York can be corrected by hooking into the video cable: you get a nice, sharp picture.

In addition, though, the cable offers extra channels.

Now, the businessmen who have been throwing together these video cable outfits are aiming for something. They have been thinking that these extra channels would net them a lot of money: by showing things on them that can't be offered on the air—highbrow drama, or perhaps X-rated stuff—they could get extra revenue. (You'd pay extra to watch it by buying an unscrambler, or whatever.)

This is turning into somewhat of a disappointment.

The cable people had foreseen, evidently, that people would stay home in droves to see the new offerings on the cable. In Show Business it's easy to forget,

though, that everybody has only 24 hours in a day, and far less than 24 hours to dispose of freely; so every leisure occupation is competing with every *other* leisure occupation. Moreover, the *residual* leisure occupation, when there's nothing else to do, is TV. It would seem that few people would watch *more* television if it were better, but many would watch *less* if they could afford to go out.

## EXTRA CHANNELS

In recent years, a number of extra channels have been made available by law. These are the UHF, or Ultra High Frequency channels. These, like cables, represent a consumer breakthrough but will have only negligible impact.

## THE PROBLEM OF ORGANIZATION

Whatever else you may say about them, the networks and TV stations are at least organized as going concerns within

the institutional structures of the country. Ideas of "community television" and other such schemes which call for some new form of social organization to spring forth are about as plausible as "community control" of schools and police—or at best likely to be as influential as "community social centers."

## INTERACTIVE TV?

Some people, I won't say who, have gotten a lot of money for something they call "interactive television." What this turns out to mean is any form of computer time-sharing that will use home TV terminals and video cables. The questions are why use home TV terminals and video cables, insofar as they would seem to promise only comparatively low-grade performance; and whether these people have thought out anything about the potential characteristics of the various media they propose with such abandon.

# "Alternate" Television, or Video Freaks

In recent years, many young folks have taken to video as a *way of life*. In the most extreme cases they say things like "the written word is dead," prompted perhaps by McLuhan. I have found it rather difficult to talk to video freaks. (It may be that some of them are against spoken words as well.) I really just don't know what they're about.

The work of these people is as exuberant as it is strange. I haven't seen much of it or understood much of what I have seen.

In some cases, "alternative television" simply means documentaries outside the normal framework of ownership and reporting. In one example cited by Shamberg (see bibliography), video freaks did excellent coverage of the 1968 Republican convention. People were allowed to speak for themselves, unlike "normal" TV journalism where "commentators" tell you what *they* see.

Now, this is hardly revolutionary; it is just good documentary-making that shucks dumb traditions artistically, much like the Pennebaker films. However, video enthusiasts claim it is somehow different, and indeed claim that video is different *in principle* from films. I have been unable to get a satisfactory clarification of this idea.

Video is being used in other ways, harder to understand, by artists (best defined as persons called "artists" within the art world today). Very odd "video pieces" have been shown at art shows, where the object seems to be to confuse the viewer—or knock him into a condition of Enlarged Perspective, shall we say. And a variety of non-objective videotapes are now being created. (A gallery show in 1969 was called "Video as a Creative Medium"—implying sarcastically that it had not been before, on the airwaves.)

Some video freaks think of video as intrinsically radical or Revolutionary. In this respect they differ interestingly from, say, the editors of the *National Lampoon*. The editors of the *National Lampoon* appear to be political radicals, but do not suggest that the very media of cartoon and joke-piece are themselves revolutionary. Some video freaks appear to be persuaded that the medium of television itself is inherently a vehicle for change.

I understand one interesting sense in which this may be true: Shamberg talks about video as a method of *self-discovery*. Seeing yourself on TV does, of course, confer certain insights. But Shamberg suggests it may expand people's consciousness in larger ways—allowing people to see the bleakness of certain pursuits (he uses the example of shopping), for instance. But if this does hit home to people, it doesn't seem to me to be the medium that's doing it but the selected content—as in all previous media. Maybe I've missed the point in some way.

These developments are all very interesting. It can be hoped that those trying to develop new forms of communication will make an effort to communicate better with those who, like the author, often cannot comprehend what they are doing.

"But decentralized transmission of information should be dominant, not fugitive. Each citizen of Media-America should be guaranteed, as a birthright, access to the means of distribution of information." (Shamberg, p. 67)

"Well, we went down there with our Porta-Pak and tried to take it inside. A guard came over and said we couldn't and even threw one of us out of the booth while the other was inside. A guard telling you what to do in a cybernetic environment?" (Shamberg, p. 53)

("Cybernetic" is evidently a code word here for what they think is good, true, beautiful, and inevitable. Cf. p. DM 126.)

"About the only generalization to be made is that community video will be subversive to any group, bureaucracy or individual which feels threatened by a coalescing of grassroots consciousness. Because not only does decentralized TV serve as an early warning system, it puts people in touch with one another about common grievances." (Shamberg, p. 57)

Nothing I have seen or heard about this is reassuring.

## BIBLIOGRAPHY

Michael Shamberg and Raindance Corporation, *Guerrilla Television*, Holt, \$4.

Nicholas Johnson, *How to Talk Back to Your Television Set*. Bantam, \$.95.

## FEELIES

OMNISENSORY MOVIES.  
THE ULTIMATE EXPERIENTIAL  
MEDIUM

In Aldous Huxley's *Brave New World*, you didn't go to the movies; you went to

the *feelies*. All the senses of an experience were prerecorded for you. (This has recently been explored in Douglas Trumbull's film *Brainstorm*.)

Is it possible?

A determined woman in New York named Barbara Strenkovsky is working on it.

In the classic hacker novel *True Names*, by Vernor Vinge, you could supposedly get a full surrounding multisensory experience from just *five electrodes*, not even implanted, at high data rates. This may be way off or it may be a very good guess.

## Television And The Mind

and

Computer

## Graphics

*In the nineteen-forties, Americans had much bigger vocabularies and could read maps; fewer Americans can read maps now than then, or speak in well-constructed sentences, or utter four-syllable words without embarrassment; and I think it is plain that this is because of television. Shrinking from whatever might confuse or challenge the viewer, the networks have simplified and simplified the news into pap which is "objective" but meaningless. (In a Robert Crumb cartoon, the newscaster reports: "Mighty speeches were made today by world leaders.")*

*While government control of media has obvious drawbacks, in European television networks, which are governmental, there is at least the sense of an obligation to raise public consciousness and understanding; whereas our broadcast leaders, who are among the most sophisticated people in the country, are more concerned with ratings than quality or elevation of the user.*

## THE GREAT OPPORTUNITY OF COMPUTER GRAPHICS

*What is offered by new computer-graphic tools (discussed on pp. DM 80-99) is the possibility of showing more on the screen: of showing foreground material for the casual or uninformed viewer together with subtle background visualizations, presentations and contents that would tell more to the discerning viewer and subliminally increase the understanding of those who know less or pay less attention.*

*In other words, computer graphics is the opportunity for television to redeem itself, to rebegin the long task of informing and upgrading public mentality.*

*This great opportunity of graphic Renaissance has been ignored; perhaps it has not even been perceived. Instead, the computer graphics of the network news have been degraded to showing huge histograms and outsize arrows in the same twisted, shallow and misleading presentations. The best example of this inanity: whether the stock market moves by two or two hundred, the arrow showing its direction is the same size.*

*Meanwhile, television commercials (notably those of Bob Abel studios) have used computer graphics to rich, complex and subtle effect, at least educating the public as to what the graphics themselves could do.*

## Technically Better TV? We Badly Need It

Systems exist for television that look as sharp as 35mm movies. These are called HDTV (high-definition television), 1125 horizontal lines. (See "Lightning in a Bottle," p. DM 84.) They could in principle begin broadcasting now at this level (it only requires two of today's channels instead of one). There are just a few problems.

1. They've really got the bugs out of NTSC, the standard system in the United States and Asia.

2. There's an awful lot of investment in NTSC.

3. There's no agreement on an HDTV standard. Last I heard, Japanese companies were divided between two rival standards and Korea was advocating yet a third.

However, high-definition TV is probably the best possible way to build a computer workstation that is compatible with any sort of video feed. And fitting computer graphics and video technologies together is hugely important. (See "DVI," p. DM 99).



# VIDEO DISKS SUPPOSEDLY

TURN,  
TURN,  
TURN

Since the forties, there have been continual announcements that video disks—movies you play on your TV off a record—were right around the corner. Earlier this year they were supposedly going to be available before Christmas. Now they *might* be on sale, "on a limited basis," in 1976. (*TV Guide*, 16 Aug '75, p. 7.) Because of the grave difficulties of engineering—inaccuracies in punching the center hole means the track can't help being off-center, for instance—some of us are skeptical.

Two systems have been confidently announced. Philips, the firm that gave us the audio cassette, has a system that will follow the spiral track on the disk from underneath with a laser. The disk turns at 30 revolutions per second, or one turn per TV frame, so it can supposedly freeze one frame when desired.

The other system is from RCA, which has a long history of me-too announcements, but at least two of them made it big (the 45 rpm record and the color TV system now used in the USA), so RCA should not be dismissed out of hand. Their disk system will supposedly go at 450 rpm (7.5 revolutions/second), but they still mean to *track it with a needle*. The man from *TV Guide* says he's seen it and it works perfectly, but I would personally look for hidden wires.

(MCA, an entertainment conglomerate, has hitched up with Philips and printed a catalog of all the movies they will supposedly make available on disk for the "MCA-Philips" system—such as *Destry Rides Again*, for around ten bucks. This is probably just a bluff; with the price of audio records what they are, no way is a movie going to cost ten bucks. But it makes RCA look weaker, which is probably the purpose.)

Well, they finally made it work. The twelve-inch videodiscs (yes, in the fifty-dollar price range) caught on only marginally; RCA dropped out; and Philips brought out the miniature laserdisc called CD, a Compact Disc for audio. That has caught on furiously worldwide.

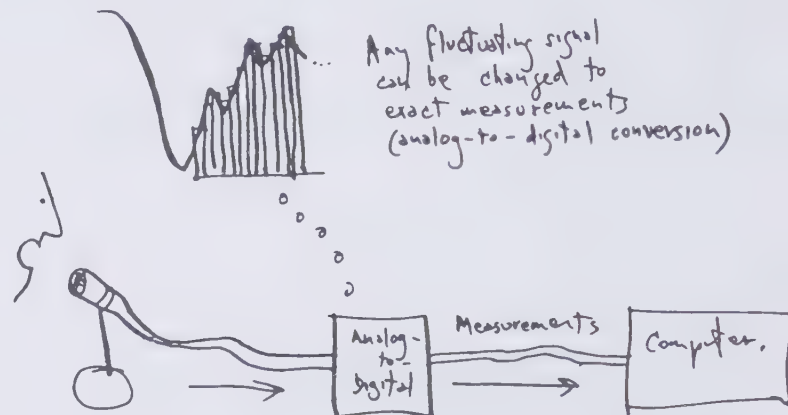
Flushed by this success, Philips has announced many uses for CD besides audio, all of which they intend to standardize. One is CD-ROM, Read-Only-Memory for computers (see p. 51), which I predict will not catch on; one is CDI, or Compact Disc Interactive (see p. 65). Other formats have been announced which do not yet merit consideration.

# AUDIO AND COMPUTERS

People are occasionally still startled to hear that computers can make sound and music. They can indeed.

First of all, note that an incoming sound is a fluctuating voltage and can thus be turned into a data structure, i.e., a string of measurements.

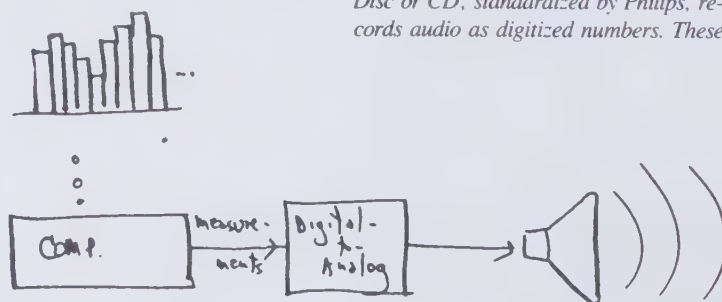
To make sound by computer is the obverse. If the computer can be set up to send out a string of measurements, these can be turned *back* into a fluctuating voltage, and thus make sounds.



In the easiest case, the computer can just send back out the voltages it originally got in. This is rather ridiculous—using the computer just as a recording device—but it's a clear and simple example.

## CDs and DATs

Ridiculous or not, digital recording has made quite an impact. A possibility that was mentioned lightly in the old Lib has become a big industry. The Compact Disc or CD, standardized by Philips, records audio as digitized numbers. These



Wish there were room to talk about plain regular audio here—matters like "binaural" recording, and why don't they make hi-fi systems based on a Grand Bus (see p. 98)? But there's no room here.

are read out by laser, converted to analog (with some error correction), and fed to your amplifier. They make spectacular sounds, whisper-soft to thunder-loud. (All too spectacular for listeners who want **background** music, rather than silence-please undivided-attention music.) No hisses and scratches and the disks don't wear out.

Compact disc is a nice example of the "diaphanous" computer mentioned earlier (p. DM 8). "The computer" in the compact disc system has **disappeared conceptually**, except insofar as it is an understood part of an understandable whole. Since the use of a computer for recording sound is well understood, we don't call it a computer any more, but "digital recording."

The same digital recording techniques used in CD should soon be available on tape machines (Digital Audio Tape or DAT recorders). I say "should" because the recording industry is lobbying fiercely to cripple these recorders in various ways so they can't copy CDs. (Studio versions can; the industry thinks that all the public does is sit around making copies of disks for friends.)

#### SAMPLING SYNTHESIZERS

The same technique of digital audio recording can also be used to make musical instruments.

The **sampling synthesizer** accepts audio signals, digitizes and saves them. Then each time you press a key on the piano-like keyboard, it plays back the measurements at the speed required for the appropriate pitch.

Top-of-the-line is the Kurzweil, at ten thousand dollars. Lowest price is a little jobbie from Casio that goes for about sixty dollars at K-Mart. The Ensoniq Mirage,

#### AUDIO ANALYSIS AND ENHANCEMENT

The problem of analyzing audio is very like the problem of analyzing pictures (see pp. DM 81-82), and indeed some of the same techniques are used. The audio goes into the computer as a stream of measurements, and the selfsame technique of Fourier Analysis is employed. This reduces the audio to a series of frequency measurements over time—but, paradoxically, loses little of the fidelity.

Once audio is reduced to Fourier patterns, it can be reconstituted in various ways: changed in timing and pitch independently, or *enhanced* by polishing techniques like those used in image enhancement (see pp. DM 81-82).

This has been done with great success by Tom Stockham at the University of Utah, who has reprocessed old Caruso records into improved fidelity. In the picture we see him with equipment of some sort and an old record.

(Stockham has been in the news lately, as one of the panel puzzling over the notorious 18-Minute Gap.)

The body of techniques for enhancing old records (as mentioned above) grows and grows. One audio engineer is disdainful, however, referring to these techniques as "painting mustaches on the tapes."

A new industry has become the "digital remastering" of old recordings to make them Sufficiently Exciting for this medium. A lot of the mixes of the old recordings have been changed, though, causing (for instance) no end of controversy among rock'n'roll purists. (George Martin has defended the digital remix of the Beatles albums as what they should have sounded like.)



University of Utah

*"Music is absolutely too much of a human experience to be computerized."*

—Karl Haas

*"This means a welcome end to computer music, which we needn't distinguish from 'real' music too much longer."*

—David Levitt in Dr. Dobbs

It was a truly stellar group that reported to Judge Sirica on 15 Jan 1974 that the 18-minute Watergate tape buzz had at least five starts and stops.

The six panelists included:

Richard Bolt, a founder of Bolt, Beranek and Newman, Inc.

Franklin Cooper, head of Haskins Laboratories.

Thomas Stockham, audio resynthesizer extraordinary

The news, however, generally referred to them as "technicians."



for under two thousand, is quite good (designed in part by Bob Yannes, who also designed the Vic-20 computer, predecessor to the Commodore 64.)

The question after that is what next; how to have the computer make interesting streams of output measurements, i.e., sounds and tones.

There are numerous methods we can't go into. Max Mathews, at Bell Labs, has for years been doing music by computer; his current system is called GROOVE. Heinz von Foerster, at the University of Illinois (Urbana), has been doing the same. Another lab at MIT has just gotten a PDP-11/45 for the same purpose.

(The problem is: can the computer keep up with the output rate needed to make music in real time?

Well, computer music synthesis is now done everywhere. The top research studio is IRCAM in Paris, whose researchers tend to be very snobbish.

Many different software packages are available. Some, like the highly-regarded MusicWorks from MacroMind (Chicago), are both notation and performance systems—you can see the notes and have the computer play them.

So you can also build up your computer as a musical instrument. Various keyboards are available, either special to the computer (there are several for the Apple II) or MIDI network keyboards (see below).

The work of Bill Buxton, at the University of Toronto, has to be mentioned. Bill has done exceptionally fine work creating different inputs and controls for the computer's musical performance, with multi-dimensional views of such elements as timbre and rhythm.

Another approach is to relieve the computer itself from making the tones, and use other devices—music synthesizers—for this, controlled by the computer. This is essentially the approach taken with General Turtle's Music Box (see p. 50), and at the Columbia-Princeton Electronic Music Center, where their RCA Mark II music synthesizer—an immense one-of-a-kind jobbie—is under more general computer control.

While none has been announced as yet, a music synthesizer that plugs into the Altair will almost certainly be available in 1976.

### **MIDI, the Musical Connection System**

Well, people have been hooking synthesizers and music boxes to computers for decades now, and it was a lot of messy custom situations until MIDI came along. Proposed by a company called Sequential Circuits in 1982, MIDI has swept the music biz—partly because this system of interconnection sells new equipment, and partly because it ties the equipment together.

MIDI (Musical Instrument Digital Interface) may be thought of as a long cable that snakes through a bunch of keyboards, synthesizers, drums and whatnot. The signals from the keyboards (or even drums and guitars) trigger notes and chords from the machines that actually generate the sounds. Instruction clusters and notes shoot down the cable, firing off the music boxes; typically these instructions and notes will be prestored in a "sequencer" and triggered by the performer. (Programs to make your PC,

Mac, Apple II or even Commodore into a sequencer are available.) The electronic settings that decide the voices of the electronic instruments—the "patches"—can be adjusted over the MIDI line as well.

As with all computer standards, MIDI is being called on to do things it wasn't designed for. It's too slow, for instance (31.25 khz), to synchronize chords from a lot of machines, since each note can take 1/100 of a second to turn on. If you send out fifty single notes to different playing devices, they will come wandering back

from the synthesizers over a period of half a second. There are various solutions, all complicated, sufficient to more than distract the nimble musician from her or his music.

While MIDI is a fairly good scheme for tying together a lot of separate pieces of equipment, it can be argued that as computers get faster, the voices can all be done in software by general-purpose computers, and so a lot of that specialized MIDI equipment won't be needed. Stay tuned.

## **MUSICAL NOTATION**

Note that the computer handling of musical notes, as symbols, is another task entirely, closely resembling computer text handling (mentioned variously in the book). A high-power structured-text system or Thinkertoy (see p. DM 50) is fine for storing and presenting written music.

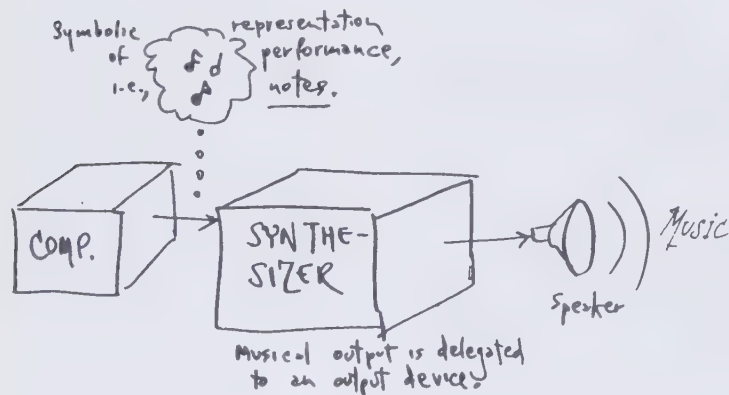
For instance, Mark of the Unicorn Professional Composer (Mark of the Unicorn, Cambridge, MA) is basically a "note processor," supporting the input and transcription of complex musical scores. And Marc Resmer, at Vassar College, has a high-res notation system that runs on VAXen over a net. These are just examples.

And, of course, such stored musical notation (a data structure) can obviously be played by the hookups mentioned.

## **AT LAST: YOUR ORCHESTRA**

It is possible to simulate an orchestra with a roomful of Macintoshen, but presumably a composer wants a less-expensive way.

Well, MacroMind (Chicago) is working on programming a setup that will permit you to layer tracks onto a Macintosh disk, recording one channel at a time, and play up to 64 channels back, in parallel, straight off the disk, mixing as you go.



## BIBLIOGRAPHY

Thomas G. Stockham, Jr., "Restoration of Old Acoustic Recordings by Means of Digital Signal Processing." Audio Engineering society preprint no. 831 (D-4), presented at Audio Engineering Society 1971 convention.

Prentiss H. Knowlton, "Capture and Display of Keyboard Music," *Datamation* May '72, 56-60. Describes a setup he built at U. of Utah that allows pianists to play music on an ordinary keyboard, and converts the input to symbolic representation in the computer. It uses an organ, a PDP-8, and a couple of CRT displays.

Heinz von Foerster and James Beauchamp, *Music by Computers*. Wiley 1969. HAS RECORDS IN BACK.

Some of the early Bell Labs work may be heard on an excellent Decca LP with the misleading title "MUSIC from MATHEMATICS." (Decca DL 79103). (Computer myths are discussed on pp. 31-32.)

David Levitt, "Music: Pushing the Sound Envelope," Dr. Dobbs' Journal of Software Tools, May 1987, 16f. An excellent brief summary of computer music.

Mark Garvin, "Music: Designing a Music Recorder," Dr. Dobbs' Journal of Software Tools, May 1987, 22f.

This somewhat-mistitled piece is actually a very good technical introduction to MIDI, just in case you were looking at the Levitt piece above. But stuff on MIDI is everywhere.

## SPEECH RECOGNITION

Surprisingly, a voice input device is now commercially available from Threshold Technology, Inc., Cinnaminson, NJ. For \$10,500 you get a device that will recognize 32 spoken words, and microphones. (Each user has to train it on his 32 words, but separate vocabularies may be stored on the computer for different users or purposes. This is still some way from the fabled "talking computer"—see pp. DM 122-126 for problems and objections—but it's undeniably a useful step.)

Lots of such devices are now on the market—the kind that accept a fixed and limited vocabulary, with a pause after each word, and that each speaker has to train to her or his own voice and dialect.

*(Genuine spoken speech has no gaps. Unlike words on the page, divided by spaces which make the word boundaries recognizable even if you don't know the language, the acoustics of real speech do not group themselves by easy recognition. Spoken speech is jammed together and requires elaborate knowledge just to find the word breaks. Compared to that problem, the rest of speech recognition is comparatively easy.) Personalized-limited-vocabulary-recognizers are still the best that can be done. The machine that will be able to take dictation—let alone understand speech—may be as far away as ever.*

*Yet for decades, now, we've heard from company after company that they're On The Verge of giving us a machine that will recognize real spoken speech. And it will probably happen someday. But don't hold your breath.*

## SPEECH BY COMPUTER

You may have heard about the various kinds of "talking computer." This deserves some explanation.

Computers may be made to "talk" by various means. One is through an output device that simply stores recordings of separate words or syllables, which the

computer selects with appropriate timing. (Machines of this type have been sold by both IBM and Cognitronics for a long time.)

A deeper approach is to have the computer synthesize speech from phonemes, or actually make the tones and noises of which speech is composed. These are very tricky matters. Bell Labs, and others, have been working on many of these approaches.

The real problem, of course, is how to decide *what to say*. (This is discussed under Artificial Intelligence, pp. DM 120.)

## LAURIE SPIEGEL

*is an eloquent New York artist-intellectual. Her musical program, "Music Mouse," is an expert system to help both expert and novice play music, exploring the musical space they like the most. Indeed, she calls her program "an anti-elitist statement." Says Laurie, "Music more than all the other arts is ruled by a political elite who control access to the tools.*

*"Manual dexterity is the stupidest possible criterion of whether you can be a musician or not.*

*"Most people's lives are pretty rotten. And most of them have been told they can't play music. And if they can come home and PLAY MUSIC, they'll begin to feel better about their lives..."*

*Consequently she set out to create a program that was "harmonious, interactive, gratifying, physical, expressive and emotional."*

*Her program, "Music Mouse," is available for both Macintosh and Amiga from Opcode Systems, Palo Alto, CA. The screen holds an unusual coordinate-space: TWO PIANO KEYBOARDS AT RIGHT ANGLES. Control is two-dimensional. In the default mode, you play the melody line on the vertical axis, the accompanying chords on the horizontal axis. With one hand you are able to control the tune and harmony, drawing patterns with the mouse in your musical space. That leaves your other hand free to select various settings, which is what the computer keyboard is now for. (It normalizes your mouse-movements to a rhythm you select, but can follow up to thirty note-changes per second.)*

*She makes no apologies for the fact that it plays music in her own style. "It won't do anything I don't like musically," she says. In fact, that's part of the point: that she is, in some sense, your accompanist.*

*Laurie never liked to get hung up in equipment talk; instead she goes straight for the virtuality (see p. DM 68), asking, "What is the parameter space and the dimensional space a person works in?" Elucidating that working space is the software design problem she has taken on.*

*(Music Mouse is also a teaching device and a hands-on MIDI controller for stage lighting, she says.)*



# HYPERMEDIA

The AI  
people don't  
understand,  
the IR  
people don't  
understand,  
the CAI  
people don't  
understand,  
and for  
God's sake  
don't tell  
IBM.

In recent years a very basic change has occurred in presentational systems of all kinds. We may summarize it under the name *branching*, although there are many variants. Essentially, today's systems for presenting pictures, texts and whatnot can bring you different things automatically depending on what you do. Selection of this type is generally called *branching*. (I have suggested the generic term *hypermedia* for presentational media which perform in this (and other) multidimensional ways.)

A number of branching media exist or are possible.

Branching *movies* or hyperfilms (see next column).

Branching *texts* or hypertexts (see p. DM 29).

Branching audio, music, etc. (see "CDI," p. DM 65).

Branching slide-shows (see "CDI," p. DM 65).

*I wish there were room here for a whole book just on this subject.*

## THEATRICALS

*The theatrics of hypermedia—the creation of "effects," and the induction of these effects into the mind of the viewer—is a completely new game, now that sequence can no longer be fully controlled by the designer. (But I think movie-makers have a special handle on it; see below, "Movies—The Model...")*

## BRANCHING MOVIES, HYPERMOVIES

*Hypermovies are suddenly upon us: branching and interactive films, or stuff very close to that idea (such as CDI).*

*Up to now, however, it's been mostly technoids experimenting with the techniques. Things will change dramatically when real film-makers start doing it. (And when the film schools recognize their pivotal role.)*

The idea of branching movies is quite exciting. The possibility of it is another thing entirely.

The only system I know of that worked was at the 1967 Montreal World's Fair (Expo '67). At the Czech Pavilion—you will recall that before the crackdown they had quite a yeasty culture going in Czechoslovakia—there were some terrific fantic systems going. One was a wall of cubes with slide projectors inside (that rolled toward you and back as they changed their pictures). And then the Movie.

The Czechoslovakian Branching Movie—I forget its real name—had the audience *vote* on what was to happen next at a number of different junctures. What should she do now, what will he do next, etc. And lo and behold! after they had voted, the lights went down, and that's what would happen next. People agreed that this gave the movie a special immediacy.

I never saw the movie—I waited in line several hours but the line was too long to get into the last showing. So instead I went backstage and talked to Radosz Cincera, who worked out the system. It turns out that it didn't work quite the way people supposed. A lot of people thought that "all the possibilities" had

## Movies ● The Model For Interactive Media and Software

People think this "new media" stuff—interactive video and whatnot—is an absolute departure from what has ever been.

Not so. It's all THE ARTFUL PLANNING OF EXPERIENCE. This applies to both the design of the actual media themselves, to basic designs **within** the media (where do "title sequences" go?) and to "authoring" within the new media.

The same for the design of software and interactive systems, whether for business or leisure. We can look to movies for all kinds of ideas, visualizations, stylistic models, structures, transitions. Study your Disney, your Eisenstein, your Welles, your Kubrick.

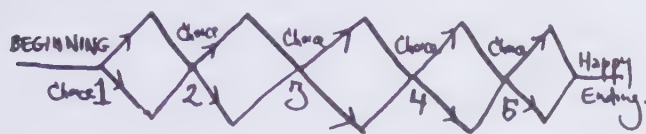
Strangely, the film schools don't know this yet: that they are in a unique position, that **the artful planning of everything that moves** is their special province. There is no question that movie-makers have the best intuitive understanding of interactive media as **experience and effect**.

(However, a few points sometimes elude them. One Hollywood producer, who has a very good feel for some of it, seemed shocked when he realized that an interactive production has **no running time**. That is, different lengths of usage are all legitimate.)

A very large hyperwork recently opened as a media installation at the Museum of Contemporary Art (Los Angeles, December '86). It is "The Erl King," a 3-video disc interactive production by Grahame Weinbren, based on Goethe's poem The Erl King. With music by Schubert (who originally set "The Erl King" to music), Zev (percussionist and performance artist), bagpipes, etc.

It uses a touch screen for user control, and, according to writeups, is continuously sensitive and responding throughout its presentational structure. The images of sound "are connected by a well-planned, though sometimes hidden, system of logic. The participant gains an overall impression of the self-contained world these images, music, texts and ideas form." —Catherine Gudis in The Contemporary.

been filmed in advance. Actually, there were always only two possibilities, and no matter *what* the audience had chosen, somehow the film was plotted to come down to the same next choice anyway:



In the actual setup, they simply had two projectors running side by side, with Film A and Film B, and the projectionist would drop an opaque slide in front of whichever wasn't chosen. But Cincera said that audiences almost always chose the same alternatives anyway, so half the movie was hardly ever used...

In the early sixties a movie was making the rounds in which audiences were supposedly allowed to vote on the *ending*—"Mr. Sardonicus," I believe it was called. From the ads it seemed that audiences would be polled as to which last reel to show. Whether the villain was to get his comeuppance, or whatever.

Then there was that Panacolor cartridge projector, mentioned elsewhere, which would have allowed choices by the user.

More recently there's the CMX system, also mentioned elsewhere. This is a setup, being jointly marketed by CBS and Memorex, for computer-controlled movie editing. But actually it could also be used as a branching movie system. Essentially the movie itself is stored frame-by-frame (as video) on big disks, made by Memorex; and, under computer control, the output can be switched rapidly among the frames, effectively showing the stored movies. (To my knowledge, the video networks haven't yet recognized the possibilities of this.)

The only trouble is, it's extremely expensive (half a million?), it has an exact storage capacity limited by the number of disk tracks (presumably one track per frame)—perhaps five minutes total on one big unit, but you can buy more—and it can only give its full performance to one viewer at a time. (Or to the whole network, live.)

It may be that the most practical branching movie system would be a cartridge movie viewer and a big stack of car-

tridges. When you make your choice, change the cartridge. But of course that's not as much fun as having it happen automatically.

## INTERACTIVE DISCS BIG AND SMALL

Some of the twelve-inch Philips videodisc machines of the early sixties allowed rapid jumps to different parts of the disc—and thus branching in the show. However, the big discs never caught on with the consuming public: videotape was cheaper for that era's main use of videodisc, which was showing movies at home.

Philips' biggest success since the audio cassette (which they designed in the mid-sixties) is the audio compact disc or CD (see pp. 60-61).

So Philips presses on: they have developed an interactive format for a smaller disc, the same-size disc as CD; this is called CDI, or Compact Disc Interactive. To play it or play with it, you will need a special new device: a CDI player—which will actually be a 68000 computer and a megabyte of memory. The disc itself can hold programs, data, text and so on. So the CDI medium can be many things: all programs, all data, all text, all computer graphics, all audio, all music, or a mix. Which it usually will be.

Many companies are betting very heavily on CDI, thinking it will be the True Home Computer (many accessories will be available).

On the other hand, the principal use of CDI will for some time (perhaps forever) be *interactive stuff* of one kind or another. (That will be the only thing you can use the plain vanilla CDI box for, without buying accessories.) So the real issue is how CDI is as a one-way medium, with no storage. Remember how we talked about the design of media at the beginning of the book? Well, they're trying to make this one the standard. Take a close and careful look.

Opinions vary. It's great for interactive slide-shows, interactive graphics (if you

don't care about saving anything), interactive music, even the delivery of text (if you don't care about making marginal notes or bookmarks).

Some very knowledgeable critics don't like the track allocations (which give priority to audio rather than to the operating system). Others, like the author, consider a read-only medium in this day and age to be intrinsically oppressive.

But the big question has to do with the public. What will it, or they, think? Will this product develop critical mass? Or will it be another design that doesn't catch on?

Many CDI discs are in production, though as of this writing (May '87) no players have been even *manufactured*, let alone sold. Those who believe in the Great Wisdom of Big Companies are betting heavily on CDI. Others are not.

## The Doomsday Disc

In 1066, some nine hundred and twenty years ago, England was conquered by one William of Normandy. William then asked for an inventory of his new possessions. The inventory was dutifully written up in something called the Doomsday Book, listed so things would be in order in case the world ended.

Now a team at the BBC has created a comparable inventory, but this on two 12-inch interactive discs. They call it the Doomsday Disc. It is an explorable map of England, allowing you to zoom in; it is writeups of England's many towns, with pictures of each (as selected by the students in those towns); and it is a statistical database on many things about the nation.

The Doomsday Disc is a big hit. In British libraries, people are supposedly standing in line to use it. Other countries are inquiring about doing their own National Discs.

Interactively, the Doomsday Disc is somewhat clumsy, but it has a number of interesting features—for instance, a three-dimensional "gallery," rather like an art gallery, where you wander spatially around pictures of the nation; but whose pictures you may walk through into the corresponding set of the town's pictures. (But when you look back, you see back into the Gallery.)

This is a large project and the BBC has a lot riding on it, including the computer they sell.

## A Perhaps Incredibly Important Machine

An important new product is the Panasonic writable videodisc drive, model TQ-2026F. It gives you 13 minutes of color video, or 24,000 frames, on an eight-inch disk, plus two channels of sound. It can be computer-controlled; seek time is half a second maximum. The price was \$15,000 in January '87 (before the devaluation of the dollar). A play-only unit costs less.

Hooking this to your computer, with appropriate software, creates an in-house hypermedia production facility of considerable power.



## PROJECTED-DUMMY CONFERENCING

The funniest, wonkiest thing ever done in the Negroponte world was a system for "video conferencing."

Now, ordinarily that term means talking to other people who aren't present while looking at them on TV.

But the system I'm about to describe is kind of hard to explain. Let's say you are a participant in this conference, along with three others, each calling in from a different place. Each of you is in a different room somewhere, but you **get the feeling** that the other three are right there, talking to you.

That's because in the room with you are three milliner's dummies—that's right, smooth headlike knobs with no features—and **projected on each one is the video image of one of the people you're talking with.** As this person speaks, his or her projected mouth on the dummy opens and closes, the eyes move, the face smiles or frowns. Because it is the person's video image.

Now, each participant has the same general experience, because each one of you has a complete set of dummies with the other people projected on them.

But more than this: when the person on your left speaks, you also see the person on your right watching him or her. You look at the person you're talking to; everyone is in the right place, glancewise.

This is because the MIT guys have created a virtual space—a virtuality, a fantasmic space—which all participants share. The outrageous mechanism is as follows.

**At each dummy there is a TV camera, pointing at the room's live participant.** Thus for four participants there are twelve video feeds in all.

In each participating roomette, the video feed of you that is shown is the one which the participant in that room would see if you were really there. If your assigned position is on the participant's left, you are projected on that dummy to that person's left. For the one to whom you are on the right in the virtual space, you are projected on the dummy to that person's right.

And correspondingly for the views of them shown to you.

So each of you is alone in a room with three talking, glancing, smiling and scowling, responding three-dimensional projections, Alone In A Crowd.

This represents, I think, the apex of techno-humor: not just creating a hilariously surrealistic environment, but doing it with money from the Defense Department.

### THE ASPEN DISK: Your Prototype Explorable

*In this tour de force (actually a tour of Aspen, Colorado), the doughty disconauts of Negroponte's Media Lab drove repeatedly through Aspen—Gstaad of the Rockies—and filmed all its turns and boulevards in winter, summer, spring and fall. The voyager, sitting at the screen of this hypershow, can drive down the street as if in a car, turn left or right, and can even turn the season from summer to winter as easily as turning on the car's air-conditioner.*

*People get different ideas from trying this, which is fine. (Its Defense Department sponsors obviously could imagine various pre-invasion uses—but getting the camera crews into Nicaragua is another problem.) Some people think the Aspen disk means that "videodisc is magnificent," which is a mistaken way to think about it. The point is not the mechanism but the explorability.*

## THE NEGENTROPY OF NEGROPONTE

They say Nicholas Negroponte, high professor of interactive whatnot at MIT, talks to the Joint Chiefs of Staff.

Gifted at fund-raising and politics, NN has been called "Toymaker to the Pentagon" and compared to MM, the empire-builder in *Catch-22*. But such invidious characterization is by those less well funded. (Which, come to think of it, includes everybody else.)

There are onlookers who have gotten the impression that Negroponte and his group invented all interactive techniques—mainly because they had the resources to demonstrate them.

Nicholas (as everyone calls him) has always known the value of gloss and a good demonstration. What some cynically refer to as a tendency to mystification might be better understood as not confusing onlookers with too many ideas.

But in fact Negroponte's realm—first the Architecture Machine Group, then the Media Lab, now a department called AMT (Art, Media, Technology)—has been a fountainhead of splendid research and design in the field of interactive sys-

tems. And, of course, they give great demo.

Best known, perhaps, are the fabled "Aspen disc" (see nearby), and a zooming-map retrieval system called Spatial Data Management. But hundreds of other grand pieces of work have been done there, and demo'd at many a conference. (Among the less-known was the interesting discovery that gray levels are an excellent way of deagging typographical outlines.)

In a world where **printed papers** are the standard medium of scholarly achievement, the importance of the demo—the in-person demo, and the taped demo that can be seen later—have been underestimated. And for their great influence in the world of interactive systems, the demonstrations done by Negroponte and his alumni loom large.

### BIBLIOGRAPHY

A videodisc is available from the Media Lab which offers a compilation of their most vivid ideas and demonstrations, including the Aspen material.

## HYPERMUSIC

Music as we know it has a sequence. There may be improvisation, but that is by the **performer** in relation to the audience. Improvisation by the **listener** is not ordinarily possible.

In today's studio recording, dozens of tracks are often recorded. But there is always a final mixdown.

These things may change dramatically.

Various new devices (such as CDI, or just today's interactive computers) open the way to **explorable music**, wherein the user may branch among modulations, rhythms, styles, themes, movements, orchestration. An immense new genre beckons.

### THOUSAND-TRACK MUSIC

Recordings whose mix is controllable by the listening user are now possible.

Thousand-track branching recordings are imaginable. It is as though the listener could wander in an orchestra, first hanging out near the brasses, then the violins.

But carry it further. New sections, themes and melodies could be accessible. This new genre could be like wandering through a forest of music—music that never stops, but that changes with every movement of the listener through a fantasmic space. Whole new themes are just over a hill. And the beginnings of other sounds beckon the listener ever onward.

### BACKGROUND MUSIC FOR OTHER HYPERMEDIA

In a related vein, it would be important to have **music which branches to accompany other hypermedia**—such as hypermovies. In this case the music's branching is derivative; but the composer's task is similar.

# OPEN VERSUS CLOSED HYPERMEDIA

The trouble with interactive media is that they all tend to be **closed**—objects that can't be combined or viewed together. (Whereas books can quote each other, and lie open at once on the same desk.)

These things are becoming more and more scattered and Balkanized. Different users can not have access to the same things, since they require big commitments to particular systems; and the products are not combinable. This is contrary to the ideals of western scholarship and authorship.

But we may be able to create a common system of open hypermedia. It may be possible to find a common method of uniting them all (provided the separate hardware pieces can be connected). This is one purpose of Project Xanadu, which is, among other things, a "hypermedia server" (see p. DM 141).

## CDIvy

There is something called the Harvard Explorer, which presents illustrated medical hyper-text on the Macintosh. On the other hand, someone at **Yale** is creating a hypershow on foreign policy alternatives—both as history and as a strategic game.

### SOCIAL CONCERNS

*Hypermedia tend to be solitary rather than social, with one person in control. But there's no reason they can't be explored by people together. If authors keep this possibility in mind, perhaps they will think of nice ways to do two-and more-person explorables.*

### VIRTUALITY, INTERACTIVE SYSTEMS, and FANTICS

This section is about the design of interactive systems, a matter which not many people had thought about in 1974, but which is now a hotbed of controversy wherever people work with computer screens.

The issues are not as many see them—mice versus keys, menus versus commands, Wizzywig versus whatever. Much larger philosophical issues are involved, are the very center of the problem.

I will discuss these matters first under the name of **virtuality**, and then under the more general heading of **fantics**.

# VIDEO GAMES OF THE MIND: THE CHALLENGE OF INTERACTIVE SYSTEMS

An interactive system is anything that responds.\* But that really includes anything with an on-off switch, since even **turning on** is a response. It includes appliances, toys, vending machines, thermostats, automobiles, cameras and recorders (the depths of whose complexities are already baffling to most). Not to mention video games.

We can now give these things enormously complex behaviors. There is indeed **no limit** to how complicated we can make the responses of these things. The problem is that **complex responses are not helpful**, unless they fit very nicely into some sensible framework.

The new class of very-responsive appliances has made possible whole new realms of atrocious design just in the household. Therefore it should not come as a surprise that those much more powerful interactive systems, our screen systems for office and other information work, are still so very bad; because these

involve the creation of large new webs of abstraction—ideas like spreadsheet and database, to say nothing of tomorrow's more powerful designs. These are our new video games of the mind, but they do not yet have nearly the performance of a well-designed video game.

It is the **clean conceptual design of these ideas** that is the problem.

Let me emphasize that this is not "psychology." Psychology asks, "What is the mind that it can deal with these ideas and systems?"

That is a nice question, but it is really not relevant to their design. The real problem is the problem of idea structure: **What are well-designed ideas and principles such that they may be easily and well grasped by the mind?**

These are the issues of virtuality. ("Fantics," discussed later, expands the subject further.)

Note that when I say "interactive systems" I intend to include **nearly all software**, since all software (at least for non-programmers) should be highly interactive. This cannot be stressed enough.

\* Living creatures are of course interactive systems, but we do not presently design them.

**INTERACTIVITY IS A CONTINUUM—**  
consider a rock, television, a ball, a map, a globe, a cat. (Sculpture is interactive, sort of: you can walk around it.)



# VIRTUALITY

Everyone knows what is real, more or less. It is the physical, the tangible, the measurable.

Now, in computerdom there is a word which is the opposite of "real." That word is **virtual**. It means **as-if**. A "virtual file" is something which is not stored as a file, but behaves as if it were a file. A "virtual disk" is something which is not a disk, but behaves as if it were a disk. And so on.

So I have proposed the term **virtuality** for the way things *seem to be*, as distinct from how they *really are*.

**Example.** In a movie, you probably can't tell whether a certain scene was shot

in a real room or a painted set, whether it was in a studio or on location. These are **realities**, and they are of no concern when enjoying (or evaluating) the movie. In terms of the movie's impact, the important thing is that **the scene takes place in a certain kind of room**. That is the **virtuality**. The virtuality is a construction that may be quite independent of its underlying materials; it is a **mental construct**.

(Thus "virtuality," as I use the term, is approximately what Seymour Papert calls a "microworld" and Myron Krueger calls also an "artificial reality" (see bibliography). But I want to take the idea in certain directions, so I use this term rather than theirs.)

Everything has a reality and a virtuality. Good examples are buildings, equipment, cars. (Using the term for com-

plex entities like people takes us in directions that are too intricate.) The extreme cases are the **movie**, which is all virtuality, and the **fishhook**, which has no virtuality—no conceptual structure or feel to the victim—until too late.

## VIRTUALITY VS. "METAPHOR"

There is a lot of talk lately about "metaphors" in interactive design, meaning resemblances or analogies that are conceptually useful. For instance, screen windows are called metaphors (see pp. DM 69, DM 74) and spreadsheets are called metaphors, because each of these resembles a physical object. (Supposedly a collection of screen windows resembles a desktop—the "desktop metaphor"—and a spreadsheet on the screen resembles a paper spreadsheet.)

But the notion of "virtuality" is much more general: a virtuality need not resemble anything; its conceptual structure and feel need only be tuned as the designer sees fit, in any direction.

The problem, you see, is **designing ideas**. The ideas someone will be working with on the computer screen can have any shape at all; their arbitrariness is mind-boggling. So the problem is carefully balancing and adjusting the ideas into a smooth whole, making these adjustments in whatever directions seem best.

## THE DESIGN OF VIRTUALITY

The real issue in computer systems, as in movies and art, is **the design of virtuality**: imagining, and carrying out, a detailed overall structure that conveys both an intended **conceptual structure** and an intended **feel** to its user. This is the

# REALITY IS OBSOLETE

The idea that objective reality is perceived by our senses is an obsolete concept. Old truisms like "seeing is believing," become much less believable as we become more aware that the biological machinery of life itself transforms images of the physical world before we are made conscious of them. These biological mechanisms share many similarities, in principle and in application, to other mechanisms observed in the natural environment and those invented for our own use. Since we are becoming more aware of the nature of perception and those mechanisms involved, now is the time to gain control of ourselves and share more discretion in the operation of our own biological machinery. We have entered the age of hyper-reality.

Day-to-day living provides only a limited variety of physical stimulus, and little incentive to manipulate the physiological and psychological processing involved. Man's historical preoccupation with the need to maintain constant images of the physical world, is a product of his extreme orientation toward physical survival in a hostile environment. The current evolving society of leisure orientations removes this need for constant images and thereby

enhances the opportunities for a more complete use of the sensory apparatus and those related brain functions. Many have turned to drugs or meditation. More specifically it is proposed here, that modern communications technology be employed as a "vehicle of departure" from this need for constant images, to bring about a more complete use of the human technology itself. Hyper-reality is the employment of technology other than the biological machinery, when used to affect the performance of the biological machinery beyond its own limitations. This is almost like making adjustments on a television set, except you are what's plugged in, and the controls are outside your body, being part of whatever technology is interfaced to the body itself. As part of such a man-machine interface you could extend your own mental processes, or if you should choose, you could just diddle with the dials. Hyper-reality is an opportunity to enhance the various qualities of the human experience. Reality is obsolete.



—How Wachspress (see p. DM 139)

COPYRIGHT 1973 AUDITAC, LTD.

central issue (and the reason for using the term). Good interactive design does not select metaphors, or figure out "interfaces," or start with tools and work upward (the UNIX philosophy, see pp. 104-105). Good interactive design is concerned with the overall structure of the virtuality presented to, and experienced by, the user: getting a conceptual structure across clearly, with an integrated feel that promotes continual understanding and fluent usage.

Virtuality has two components: **conceptual structure and feel**. The conceptual structures of all cars are alike: they go forward and back, left and right; they start and stop, and so on.

But the **feel** of cars is another matter. By this I mean not just the steering and handling (what "feel" usually refers to in cars), but the atmospherics—the paint, the smell, the door slam, the detailing. Which is much of what you pay for. (In movies, words for "feel" include **atmosphere**, **effects** and **production values**.)

#### BIBLIOGRAPHY

- Theodor H. Nelson, "Interactive Systems and the Design of Virtuality." Two-part article, *Creative Computing*, November and December 1980.
- Theodor H. Nelson, "Getting It Out of Our System." In George Schecter

(ed.), *Information Retrieval: A Critical View*. Thompson Book Company, 1967.

Published in 1967, this was an early comparison between movies and interactive systems.

Paul Heckel, *The Elements of Friendly Software Design*, Warner Books, 1982.

Ben Schneiderman, *Designing the User Interface*. Addison-Wesley.

Better than most.

Myron Krueger, *Artificial Reality*, Addison-Wesley, 1983.

Seymour Papert, *Mindstorms: Children, Computers and Powerful Ideas*, Basic Books, 1982.

Donald A. Norman and Stephen W. Draper (eds.), *User Centered Design: New Perspectives on Human-Computer Interaction*. Lawrence Erlbaum Associates, 365 Broadway, Hillsdale, NJ.

ISBN 0-89859-872-9 (paper), ISBN 0-89859-781-9 (hardback).

#### CONFERENCES:

SIGCHI, the ACM Special Interest Group in Computers and Human Interaction, meets in the spring. Next meetings are:

May 15-19, 1988, Washington, D.C.  
April 29-May 5, 1989, Austin, TX

## THE PROBLEM WITH METAPHORS

"I thought that some of the metaphysical imagery was really particularly effective...which seemed to counterpoint the surrealism of the underlying metaphor..."

Struggling heroes  
in Douglas Adams'  
Hitchhiker's Guide to the Galaxy

"Metaphors" are resemblances embodied in software that help carry meaning. Supposedly. Thus a bunch of windows on a screen is called a "desktop metaphor," since it looks to some people like papers on a desk.

This term leads all too easily to muddled discussions. "Metaphor" suggests a well-defined idea, and talk becomes clouded with pseudo-issues such as "the right way to represent a desktop"; but the problem is shaping the details of this virtuality amongst a vast field of possibilities.

The details selected are of acute importance, and different implementations of "the same metaphor" may have greatly varying characteristics. What matters is the best way to detail the virtuality, not finding a relationship to anything that came before. The best new virtualities will have no antecedents.

"God is in the details."

-Shaker motto often quoted by Mies van der Rohe.

The design of interactive systems, large and small, is the design of ideas; especially,

## THE DESIGN OF PRINCIPLE

There are many small innovations in interactive systems. Ring menus, cursors that change to show what's happening, "elevators" to pan windows on a data surface. Icons. And many more.

These are the smaller ideas of interactive systems, a large and growing vocabulary. But you can't just mix and match them.

When you are working on a design, it is the larger overall ideas that count, the overall principles that tie the system together.

It is designing these principles, and refining and tuning them, that is most arbitrary and difficult. Because there are many different ways to work out the details of the same overall ideas.

**Example: Grouping in MacDraw.** In Mark Cutter's MacDraw program, for instance, you can draw separate picture parts, position them together on the screen, and then glue them together so they may be moved (or stretched, or copied) as a unit thereafter. This is called **grouping**. To group some picture parts, you select them in one way or another; each then displays eight little squares to tell you it has been selected. When you have selected all the parts of the picture that you want to group together, you give the "group" command—and now the whole **group** shows that it is a unit by displaying the eight little squares.

There are many subtle variations that Cutter might have chosen; he carefully selected this mechanism, and what it does under various circumstances, to be powerful, simple and unobtrusive.

**Example: Ballblazer.** A particularly important example is the game of Ballblazer (from Lucasfilms, for the Atari 800; action designed and programmed by Dave Levine). In Ballblazer, each of two



The good design is a peak, from which all deviations are downhill. It is simple, clear, powerful, elegant, self-revealing, and clean; and it works well when people use it.

players is in control of a sort of rocket sled on a big checkerboard. The two sleds compete to seize a ball and carry it across opposite goal lines. (There are various ways to steal the ball from the opponent.)

Dave Levine designed this in the following way. He wanted to set it up to be easy to learn; in particular, **the simplest possible beginner's action had to be a valid strategy.**

This beginner's strategy that Dave chose was to push the joystick forward. Thus pushing the stick forward had to, in some way, **take your sled to the ball, seize the ball** (if the ball was not held by the other player), **turn your sled toward the opponent's goal** (assuming that you now have the ball), and finally, **start for the goal line.**

These parts of the system's response to that simple action Dave took as **given.** Then, working backward, he figured out clean principles of operation that would generate these examples and make sense. Though Ballblazer is a high-performance game, which took lots of programming smarts, this abstract design was probably the hardest part of the work. (I call this the **substructuring** of principle, working out the details of a principle from examples.)

Certain additional constraints were obvious: the valid beginner's strategy of pushing the stick forward could not be the **optimal** strategy, or there could be no game; and the principles chosen would have to generate a variety of interesting plausible strategies, offensive and defensive, at escalating levels of skill. Practice and study should reveal these other aspects and ramifications, as ramifications are generally revealed in Pac-Man.

Finally, the principles he was designing for the controls and behavior of the sled would have to be clear and simple.

This is a tall order, at which I believe Dave succeeded brilliantly, and it is for this reason that I place him in the first rank of interactive-system designers.

## THE GREAT EXPANSE OF OPTION

Nowhere do you have a freer hand in the substruction and detailing of principle than in interactive systems.

When you are designing principles in science, i.e., hypotheses about the world, their validity is measured against their empirical validity in the world. When you are designing principles in law, they are limited by precedent, constitutionality, enforceability, and all the considerations known to the legal profession. When you are designing principles in literature—say, the plot of a novel—you are constrained not to violate that great body of background knowledge and expectation, held by the reader, that determines plausibility and story value.

But when you are designing principles for the operation of computers, there are virtually (pun intended) no limitations. You may ring bells, flash colors, animate humorous characters, crunch numbers, print messages, or anything else at all. All the constraints upon your design have to do with your general purposes and the desirable attributes of the principles themselves: clarity, simplicity, playability, need for the principles to **jump right out at beginners**, and so on.\*

In a way this makes the job easier: there are no **extraneous** concerns, and you may focus very clearly on your intentions and sphere of involvement. But insofar as the **complexity** of the problem, its arbitrariness and the size of the option-space are concerned, the design problem can be as hard as you make it for your-

\* For readers interested in general schematics (my philosophical system, see p. 15), the design of principle I subsume under "thematics"—the consideration and crafting of generalities and groupings which are either vague (soft principle) or may be made precise (hard principle). See also "Interactive Systems and the Design of Virtuality," cited in bibliography.

self—like the design of a musical composition.

## CONDENSING THE UNIFIED DESIGN

Starting out with a lot of possibilities in mind—features, system parts, screen visualizations, controls you might use, over-all principles you might try, **and their possible variations**, the design of interactive systems is properly the condensing of all these ideas into a unified whole of principle, comparing all the possible interactions and edge cases of the mechanisms and the principles, and compacting and resolving them into a simple and elegant set, without exceptions or complications. Which takes repeated re-tuning and redesign throughout the global system.

That is what the design of interactive systems **should** be. And that is why it is so hard, which is in turn why most software is so terrible.

## THE REAL ISSUE IN SOFTWARE DESIGN

The real issue in software design is **the design of ideas.** But most people are looking at the wrong levels, fixating on particulars, and not seeing the immensity of option—or the imperative of cleanly condensed structure.

## "INTERFACES"

should be invisible, simply representing the conceptual structure. Using them should be like steering a car, like reaching into the water.

**First** you create the unified virtuality, **then** map the contents to it. Except that they should already be obvious. "The interface problem" is **almost always the wrong issue.** The real issue is usually the **conceptual design.**

EVERYTHING IS DEEPLY INTERTWINGLED.  
Especially in a well-designed interactive system.

A system I designed almost twenty years ago is still a nice example of the design of principle.\*

It was originally intended for editing free text at upper-case teletypes, but allowing you to move around quickly and flexibly. The idea was for you to be able to

\*There were three implementations of the JOT system, none of commercial grade. The author still offers the complete design on license to interested parties.

type normally, but rapidly move anywhere.

There were ways of stepping forward and back, but the real charm of the system was the main stepping key chosen for moving forward by one unit (word, sentence or paragraph, depending on the mode you were in).

You used the *space bar*.

You also used the space bar to end a word, and (following normal secretarial practice), you put two spaces at the end of a sentence. (This told the system it was the end of a sentence, and not just a contrac-

tion.)

The cursor was *always at a space between words*, unless you were typing or changing a word.

THE BEAUTY PART. There was *no interference* between these two functions. You could type normally all you liked, and you could step around the text, using the same control. (The secret: the second space after a word told it to start stepping; the third space after a sentence told it to start stepping.)

And the most important part of this was that *it seemed surprisingly smooth*

and natural in a short time.

The fitting-together of these two uses of the space bar (which took a little work) is an especially good example of designing principles to fit together.

To save extra keying it capitalized the beginnings of sentences automatically. This remained true even if you changed, or rearranged, the beginning of a sentence. In ordinary prose, only names had to be capitalized.

Amusing other feature: one of the implementations had different sound effects for every action.

# JOT<sup>TM</sup>: Juggler Of Text.

From "A Human Being's Introduction To the JOT System." ©1972 T. Nelson.

Here's how simple it is to create and edit text with the JOT system. Since your typewriter is now a JOT machine, not every key does what it used to.

CREATING TEXT: just type it in.

You type: The quick brown fox jumps over the lazy dog.

It types: The quick brown fox jumps over the lazy dog.

REVIEWING A SENTENCE YOU JUST TYPED: the back-arrow takes you back, the space bar steps you through.

You type: ← sp sp sp sp

It types: (bell) The quick brown fox

DELETIONS AND INSERTIONS: the RUBOUT key rejects words you don't want. To insert , merely type.

You type: ← sp sp RUBOUT lithe sp sp sp sp sp

It types: (bell) The quick /brown/ lithe fox jumps over the lazy dog.

REARRANGING TEXT: first we make three Cuts in the text, signalled by free-standing exclamation points.

You type: sp ! sp ! sp ! fox

It types: The ! quick ! lithe ! fox

TO REARRANGE IT, YOU TYPE: LINE FEED key. This exchanges the two pieces between the cuts.

CHECK THE RESULTS:

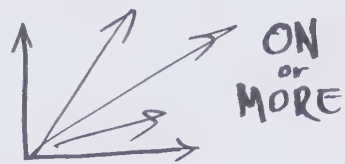
← sp sp sp sp  
(bell) The lithe quick fox



## SOME FUNDAMENTAL CONTROL RELATIONS

the rules for  
SWITCHES  
and LEVERS

OFF  
or  
LESS  
(this is Cartesian)

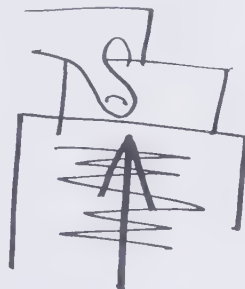


ON → MORE  
KNOBS  
OFF ← LESS

WHICH WAY SHOULD AN 'UP' CONTROL  
MOVE A SCREEN'S CONTENTS?



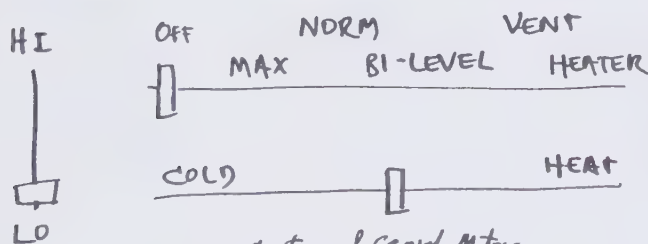
OR PAPER-PULLING MODE?



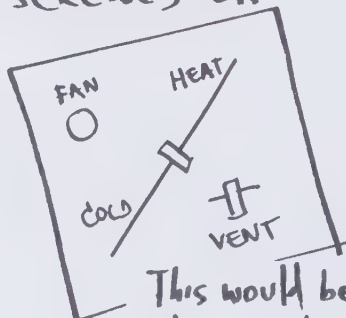
"WINDSHIELD"  
IS CORRECT —  
BECAUSE PEOPLE ARE DRIVERS.

EITHER MAKES SENSE  
IF YOU STOP AND THINK ABOUT IT —  
BUT IT'S YOUR FIRST REACTION THAT COUNTS.

AUTOMOBILE HEATERS ARE TOTALLY SCREWED UP.



Courtesy of General Motors.



This would be  
too simple,  
wouldn't it.

## THE DESIGN OF APPLIANCES

The Three Mile Island disaster occurred partly because the control-panel design was the opposite of common sense: for instance, Up meant Off.

Most people cannot work the automatic-recording-by-the-clock features of their VCRs.

If people turn on the wrong burner of electric stoves, that is **bad design** of the stove.

are unobtrusive, transparent and unconfusing at the simpler levels of operation.

Now, there are some customers who **want** stupid overcomplication, and surely they are entitled to have it, but I suggest that designers should first learn **how to design simplicity**—which is very hard—before they give in to the aesthetic of senseless complication.

### THE DESIGN OF APPLIANCES

is in an atrocious state, especially because the people who do it right now think lots of buttons are a good idea. Engineers and designers keep adding features and putting buttons in rows.

ALL EQUIPMENT SHOULD BE DESIGNED FOR USE IN EMERGENCIES BY BEGINNERS. (Generally it **can** be so designed; people just don't try.)

The design of the controls should correspond as closely as possible to the conceptual structure of the device. (See the superb TC-50 tape recorder, p. DM 77.)

Additional features should be allowed, provided they **do not get in the way**, and

### THE TRAGEDY OF INTERACTIVE SYSTEM DESIGN

Because people think they want "more and more features" (instead of extensible elegant clear methods), systems get more and more cluttered.

Lately, it is being noticed that design of interactive systems by committee compromises the design (in both senses): creating disjointed and disorganized messes.

The solution (as practiced in some places); **bigger design committees!**

There has to be a chief designer with dictatorial powers, just like a movie director; because only that way can a great variety of tradeoffs across the whole system be sensibly decided.

The standard foot-pedals  
for dictation machines  
are **BACKWARD!**

FAST FWD	PLAY	RE- WIND
-------------	------	-------------





## The French Have A Word For It ●

In French they use the term l'Informatique to mean, approximately, the presentation of information to people by automatic equipment.

As well as computer and information science. Unfortunately the English equivalent, informatics, has been preempted. There is a computer programming firm called Informatics, Inc., and when I wrote them about this in the early sixties they said they did not want their name to become a generic term. Trademark law supports them in this to a certain extent. (Others, like Wally Feurzeig, want that to be the word regardless.) But in the meantime I offer up the term *fantics*, which is more general anyhow.

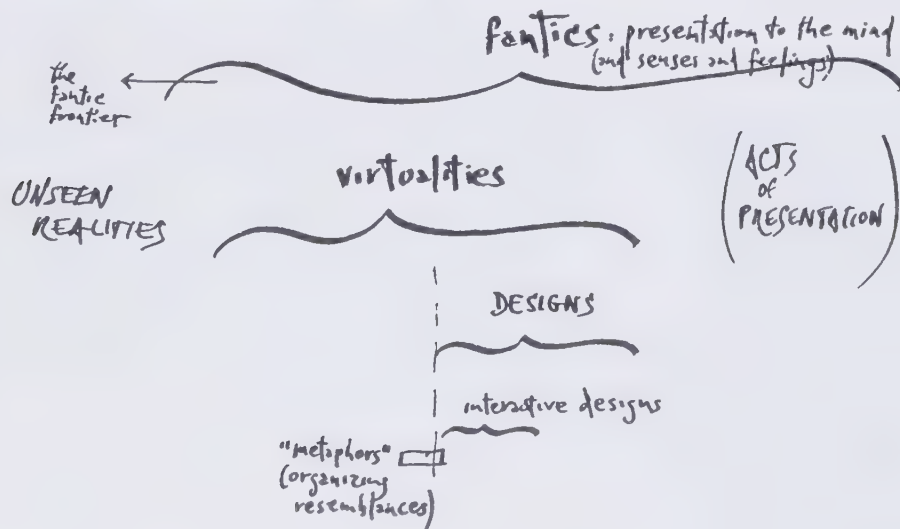
More than one associate has told me that the word "*fantics*," for my combined conception of explanation and showbiz and the design of virtualities, is too big. "You can't mean that much," said one. "You're not allowed to lump all those things together," said another.

Very strange assertions. In a world where there are words like *physics*, *psychology* and *universe*, how can a new word be said to comprehend too much?

Nevertheless, it is clear that lumping all these things together presents large conceptual difficulties for many people. That is one reason I have in recent years concentrated on the term *virtuality*, which helps focus more exactly on various issues of interactive design.

(Note that *fantic space*, discussed nearby, is about the same as *virtuality*.)

Just to tie all these terms together, here's a handy diagram:



About the "*Fantics*" piece: many people found this article hard and strange. While revising this book, at first I thought I'd cut it all up and change it, but it's still what I think. These ideas are the heart of the book.

FEELED-EFFECT SYSTEMS ARE THE NEW FRONTIER

## FANTICS

BUT IT'S SHOWMANSHIP THAT'S PARAMOUNT, NOT ANY TECHNICAL SPECIALTY

Ah, Love! could you and I with Him conspire  
To grasp this sorry Scheme of Things entire,  
Would not we shatter it to bits—  
and then  
Re-mould it nearer to the Heart's Desire!

—Edward Fitzgerald

Almost everyone seems to agree that Mankind (who?) is on the brink of a revolution in the way information is handled, and that this revolution is to come from some sort of merging of electronic screen presentation and audio-visual technology with branching, interactive computer systems. (The naive think "the" merging is inevitable, as if "the" merging meant anything clear. I used to think that too.)

Professional people seem to think this merging will be an intricate mingling of technical specialties, that our new systems will require work by all kinds of committees and consultants (adding and adjusting) until the Results—either specific productions or overall Systems—are finished. Then we will have to Learn to Use Them. More consulting fees.

I think this is a delusion and a con-game. I think that when the *real* media of the future arrive, the smallest child will know it right away (and perhaps first). That, indeed, should and will be the criterion. When you can't tear a teeny kid away from the computer screen, we'll have gotten there.

We are approaching a screen apocalypse. The author's basic view is that RESPONSIVE COMPUTER DISPLAY SYSTEMS CAN, SHOULD AND WILL RESTRUCTURE AND LIGHT UP THE MENTAL LIFE OF MANKIND.

I believe computer screens can make people happier, smarter, and better able to cope with the copious problems of tomorrow. But only if we do right, right now.

WHY?

The computer's capability for branching among events, controlling exterior devices, controlling outside events, and mediating in all other events, makes possible a new era of media.

Until now, the mechanical properties of external objects determined what they were to us and how we used them. But henceforth this is arbitrary.

The recognition of that arbitrariness, and reconsideration among broader and more general alternatives, awaits us. All the previous units and mechanisms of learning, scholarship, arts, transaction and confirmation, and even self-reminder, were based in various ways upon physical objects—the properties of paper, carbon paper, files, books and bookshelves. To read from paper you must move the physical object in front of you. Its contents cannot be made to slide, fold, shrink, become transparent, or get larger.

But all this is now changing, and suddenly. The computer display screen does all these things if desired, to the same markings we have previously handled on paper. The computer display screen is going to become universal very fast; this is guaranteed by the suddenly rising cost

of paper. And we will use it for everything. This already happens wherever there are responding computer screen systems. (I have a friend with two CRTs on his desk; one for the normal flow of work, and one to handle interruptions and side excursions.) A lot of forests will be saved.

Now, there are many people who don't like this idea, and huff about various apparent disadvantages of the screen. But we can improve performance until almost everyone is satisfied. For those who say the screens are "too small," we can improve reliability and backup, and offer screens everywhere (so that material need not be physically carried between them).

The exhilaration and excitement of the coming time is hard to convey on paper. Our screen displays will be alive with animation in their separate segments of activity, and will respond to our actions as if alive physically too.

The question is, then: HOW WILL WE USE THEM? Thus the design of screen performances and environments, and of transaction and transmission systems, is of the highest priority.

## MEDIA

What people don't see is how computer technology now makes possible the revision and improvement—the transformation—of all our media. It "sounds too technical."

But this is the basic misunderstanding: the fundamental issues are NOT TECHNICAL. To understand this is basically a matter of MEDIA CONSCIOUSNESS, not technical knowledge.

A lot of people have acute media consciousness. But some people, like Pat Buchanan and the communards, suggest that there is something shabby about this. Many think, indeed, that we live in a world of false images promulgated by "media," a situation to be corrected. But this is a misunderstanding. Many images are false or puffy, all right, but it is incorrect to suppose that there is any alternative. Media have evolved from simpler forms, and convey the background ideas of our time, as well as the fads. Media today focus the impressions and ideas that in previous eras were conveyed by rituals, public gatherings, decrees, parades, behavior in public, mummies' troupes... but actually every culture is a

world of images. The chieftain in his palanquin, the shaman with his feathers and rattle, are telling us something about themselves and about the continuity of the society and position of the individuals in it.

Now the media, with all their quirks, perform the same function. And if we do not like the way some things are treated by the media, in part this stems from not understanding how they work. "Media," or structured transmission mechanisms, cannot help being personalized by those who run them. (Like everything else.) The problem is to understand how media work, and thus balance our understanding of the things that media misrepresent.

## THOUGHTS ABOUT MEDIA:

### 1. ANYTHING CAN BE SAID IN ANY MEDIUM.

Anything can be said in any medium, and Inspiration counts much more than 'science'. But the techniques which are used to convey something can be quite unpredictable.

### 2. TRANSPOSABILITY

There has always been, but now is newly, a UNITY OF MEDIA OPTIONS. You can get your message across in a play, a tract, a broadside, a textbook, a walking sandwich-board, a radio program, a comic book or fumetti, a movie, a slideshow, a cassette for the Audi-Scan or the AVS-10, or even a hypertext (see p. DM 29).

(But transposing can rarely preserve completely the character or quality of the original.)

### 3. BIG AND SMALL APPROACHES

What few people realize is that big pictures can be conveyed in more powerful ways than they know. The reason they don't know it is that they see the *content* in the media, and not *how* the content is being gotten across to them—that in fact they have been given very big pictures indeed, but don't know it. (I take this point to be the Nickel-Iron Core of McLuhanism.)

People who want to teach in terms of building up from the small to the large, and others who (like the author) like to present a whole picture *first*, then fill in the gaps, are taking two valid approaches.

(We may call these, respectively, the Big Picture approach and the Piecemeal approach.) Big pictures are just as memorable as picky-pieces if they have strong insights at their major intersections.

## 4. THE WORD-PICTURE CONTINUUM

The arts of writing and diagramming are basically a continuum. In both cases the mental images and cognitive structures produced are a merger of what is heard or received. Words are slow and tricky for presenting a lot of connections; diagrams do this well. But diagrams give a poor feel for things and words do this splendidly. The writer presents exact statements, in an accord-structure of buts and indeeds, molded in a structure of connotations having (if the writer is good) *exact impreciseness*. This is hardly startling: you're always selecting what to

say, and the use of vague words (or the use of precise-sounding words vaguely) is simply a flagrant form of omission. In diagrams, too, the choice of what to leave in and out, how to represent overweening conditions and forces and exemplary details, are highly connotative. (Great diagrams are to be seen in the *Scientific American* and older issues of *TIME* magazine.)

This word-picture continuum is just a part of the broader continuum, which I call Fantics.

## FANTICS

By "fantics" I mean the art and science of getting ideas across, both emotionally and cognitively. "Presentation" could be a general word for it. The character of what gets across is always dual: both the explicit structures, and

### Should I have called it

TEACHOTECHNICS?  
SHOWMANSHIPNOGOGY?  
INTELLECTRONICS? (S.Ramo)  
THOUGHTOMATION?  
MEDIA-TRONICS?

### ABOUT THE TERM 'FANTICS.'

First of all, I feel that very few people understand what interactive computer systems are *about*. It's like the story of the blind men and the elephant—each thinks it's a different thing (based, usually, on his own technical specialty).

But I think it's all show business. PENNY ARCADES are the model for interactive computer systems, not classrooms or libraries or imaginary robot playmates. And computer graphics is an intricate branch of movie-making.

Okay, so I wanted a term that would connote, in the most general sense, the showmanship of ideas and feelings—whether or not handled by machine.

I derive "fantics" from the Greek words "phainein" (show) and its derivative "phantastein" (present to the eye or mind).

You will of course recognize its cousins *fantastic*, *fantasy*, *phantom*. ("Phantom" means *what is shown*; in medical illustration it refers to an opaque object drawn as transparent; a "phantom limb" is an amputee's temporary feeling that the severed limb has been restored.) And a *fantast* is a dreamer.

The word "fantics" would thus include the showing of anything (and thus writing and theater), which is more or less what I intended. The term is also intended to cover the tactics of conveying ideas and impressions, especially with showmanship and presentational techniques, organizing constructs, and fundamental structures underlying presentational systems.

Thus Engelbart's data hierarchy (p. DM 17), SKETCHPAD's Constraints (p. DM 96), and PLATO's fantic spaces (p. DM 93) are fantic constructions that need to be understood if we are to understand these systems and their potential usages.



feelings that go with them. These two aspects, exactness and connotation, are an inseparable whole; what is conveyed generally has both. The reader or viewer always gets feelings along with information, even when the creators of the information think that its "content" is much more restricted. A beautiful example: ponderous "technical" manuals which carry much more connotatively than the author realizes. Such volumes may convey to some readers an (intended) impression of competence, to others a sense of the authors' obtuseness and non-imagination. Explicit declarative structures nevertheless have connotative fields; people receive not only cognitive structures, but impressions, feelings and senses of things.

Fantics is thus concerned with both the arts of effect—writing, theater and so on—and the structures and mechanisms of thought, including the various traditions of the scholarly event (article, book, lecture, debate and class). These are all a fundamentally inseparable whole, and technically-oriented people who think that systems to interact with people, or teach, or bring up information, can function on some "technical" basis—with no tie-ins to human feelings, psychology, or the larger social structure—are kidding themselves and/or everyone else. Systems for "teaching by computer," "information retrieval," and so on, have to be governed in their design by larger principles than most of these

people are willing to deal with: the conveyance of images, impressions and ideas. This is what writers and editors, movie-makers and lecturers, radio announcers and layout people and advertising people are concerned with; and unfortunately computer people tend not to understand it for beans.

In fantics as a whole, then, we are concerned with:

1. The art and science of presentation. Thus it naturally includes
2. Techniques of presentation: writing, stage direction, movie making, magazine layout, sound overlay, etc. and of course
3. Media themselves, their analysis and design; and ultimately
4. The design of systems for presentation. This will of course involve computers hereafter, both conceptually and technically; since it obviously includes, for the future, branching and intricately interactive systems enacted by programmable mechanisms, i.e. computers. Thus computer display, data structures (and, to an extent, programming languages and techniques) are all a part.

Fantics must also include:

5. Psychological effect and impact of various presentational techniques—but not particular formal aesthetics, as of haiku or musical composition. Where directly relevant fantics also includes
6. Sociological tie-ins—especially supportive and dysfunctional structures, such as tie-ins with occupational structure; sponsorship and commercials; what works in schools and why. Most profoundly of all, however, fantics must deal with psychological constructs used to organize things:
7. The parts, conceptual threads, unifying concepts and whatnot that we create to make aspects of the world understandable. We put them into everything, but standardize them in media.

For example, take radio. *Given* in radio—the technological fundament—is merely the continuous transmission of sound. *Put into it* have been the "program," the *serial* (and thus the *episode*), the *announcer*, the *theme song* and the

*musical bridge*—conventions which are useful presentationally.

The arbitrariness of such mental constructs should be clear. Their usefulness in mental organization perhaps is not.

Let's take a surprise example, nothing electronic about it.

Many "highways" are wholly fictitious—at least to begin with. Let's say that a Route 37 is created across the state: that number is merely a series of signs that users can refer to as they look at their maps and travel along.

However, as time goes by, "Route 37" takes on a certain reality as a conceptual entity: people think of it as a *thing*. People say "just take 37 straight out" (though it may twist and turn); groups like a Route 37 Merchants' Association, or even a Citizens to Save Scenic 37, may spring up.

What was originally simply a nominal construct, then, becomes quite real as people organize their lives around it.

This all seems arbitrary but necessary in both highways and radio. What, then, does it have to do with the new electronic media?

Simply this: till now the structures of media somehow sprang naturally from the nature of things. *Now they don't anymore*. Radio, books and movies have a natural inner dynamic of their own, leading to such constructs. While this may prove to be so for computer media as well (—as I argued in "Getting It Out of Our System," cited p. DM 128), then again it may not. In other words, WE MUST ACKNOWLEDGE THAT WE ARE *INVENTING* PRESENTATIONAL TECHNIQUES IN THE NEW MEDIA, not merely transporting or transposing particular things into them because they seem right. *The psychological constructs of man-machine systems may turn out to be largely arbitrary*. Thus bringing to terminal systems conventions like dialogue instruction ("CAI"), or arbitrary restrictions of how things may be connected, presented or written on the computer may be a great mistake.

The highway-number analogy continues. The older highways had numbers for convenience, and our travels became organized around them, and particular highways (like "U.S. 1" and "Route 66") came to have special character. But now with the Interstates, a highway is a *planned, sealed unit*, no longer just a col-

lection of roads gathered together under a name.

This unit, the Interstate, is not merely a psychological construct, but a planned structure. Knowing what works and what doesn't in the design of fast highways, the Interstates were built for speed, structured as closed units. Designing them with limited access has been a conscious decision in the system design for well-based reasons, *not* a chance structure brought in from horse-and-buggy days.

Now, the constructs of previous media—writing, films, other arts—evolved over time, and in many cases may have found their way to a "natural" form. But because of the peculiar way that computer media are currently evolving (—under large grants largely granted to professionals who use very large words to promote the idea that their original professions are largely applicable—), this sort of natural evolution may not take place. The *new* constructs of computer media, especially computer screen-media, may not have a chance to be thought out. We need designs for screen presentations and their mixture—vigetting, windows, screen mosaics, transformed and augmented views, and the rapid and comprehensible control of these views and windows. We are still just beginning to find clever viewing techniques, and have hardly begun to discover highly responsive forms of viewability and control (cf. collateration in "Thinkertoys," p. DM 50). (See T. Nelson, "A Conceptual Framework for Man-Machine Everything," cited p. DM 80, and material on controls, nearby.)

## THE MIND'S UNIFICATION

One of the remarkable things about the human mind is the way it ties things together. Perceptual unity comes out of nowhere. A bunch of irregular residential and industrial blocks becomes thought of as "my neighborhood." A most remarkable case of mental unification is afforded by the visage of our good friend Mickey Mouse. The character is drawn in a most paradoxical fashion: two globelike protrusions (representing the ears) are in *different* positions on the head, depending on whether we view him from the front or the side. No one finds this objectionable; few people even notice, it seems.

THE  
GESTALT,  
DEAR  
BRUTUS,  
IS NOT  
IN OUR  
STARS  
BUT IN  
OURSELVES.





MICKY MOUSE  
(frontal)



(lateral)

#### POSSIBLE RECONCILIATIONS:



Diagonal Mounting



Rolling  
Relative  
to Camera

What this shows, of course, is the way the mind can unify into a consistent mental whole even things which are inconsistent by normal rules (in this case, the rules of three-dimensional structure).

Even perceptions are subject to the same principle of unification. The fingernail is an excrescence with no nerves in it; yet somehow you can *feel things with your fingernails*—tying together disparate sensations into a unified sense of something in the world (say, a coin you're trying to pick up). In the same way, an experienced driver *feels the road*; in a very real sense, the car's wheels and suspension become his own sensory extensions.

This principle of mental unification is what makes things come together, both literally and figuratively, in a fantic field. A viewer sees two consecutive movie shots of streets and unifies them into one street; controls, if you are used to them, become a single fused system of options; we can have a sense of a greater whole, of which one view on a screen is a part.

## CONTROLS: THEIR UNIFICATION AND FEEL

Controls are intimately related to screen presentation, just as arbitrary, and just as important.

The artful design of control systems is a deeply misunderstood area, in no way deconfused by calling it "human factors." There are many functions to be controlled, such as text editing operations, views of the universe on a screen, the heading of a vehicle, the tilt of an aircraft, the windage and adjustments of artillery, the temperature of a stove burner and any other controllable devices. And nowadays *any conceivable* devices could control them—pushbuttons, knobs, cranks, wheels, levers and joysticks, triggers, dials, magic wands, manipulation by lightpen on CRT screens (see p. DM 88), flicks of the finger, the *turning of the eyes* (as in some experimental gun-aiming devices), the human voice (but that introduces problems—see p. DM 61), keyboards, electronic tablets, Engelbart mice and chordwriters, and so on.

The human mind being as supple as it is, anything whatever can be used to control systems. The problem is having it be a *comprehensible whole*.

As already remarked, our ability mentally to unify things is extraordinary. That we somehow tie together clutch, gear, accelerator and brake into a comprehensible control structure to make cars go and stop should amaze and instruct.

Engineers and "human factors" people speak as though there were some kind of scientific or determinate way to design control systems. Piffle. We choose a set of controls, much like an artist's Palette, on the basis of general appropriateness; and then try best and most artistically to fit them to what needs doing. (See "JOT," p. DM 71.)

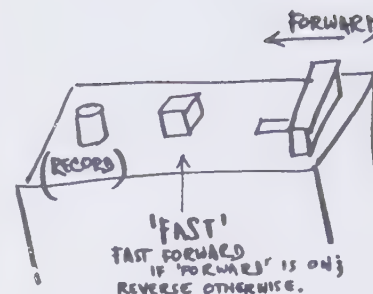
The result must be conceptually clear and retroactively "obvious"—simply because clarity is the simplest way to keep the user from making mistakes. Clear and simple systems are easier to learn, harder to forget, less likely to be screwed up by the user, and thus are more economical—getting more done for the resources put in. (See "JOT," p. DM 71.)

There is a sort of paradox here. The *kinds* of controls are totally arbitrary, but their unification in a good system is not. Smoothness and clarity can come from disparate elements. It is for this reason

that I lay particular stress on my JOT system for the input and revision of text, using a palette of keys available on the simplest standard computer terminal, the 33 Teletype. I cannot make the final judgment on how good this system is, but it pleases me. JOT is also an important example because it suggests that a conceptually unified system can be created from the artful non-obvious combination of loose elements originally having different intended purposes.

Mental analogy is an important and clear control technique. We tend to forget that the steering wheel was *invented*, separately replacing both the boat's tiller and the automobile's tiller. We also forget that the use of such steering mechanisms must be actually learned by children. Such continuous analogies, though, require corresponding continuities in the space to be controlled—an important condition.

Simplicity and clarity have nothing to do with the *appearance* of controls, but with the clarity and unique locatability of individual parts. For this reason I find deplorable the arrayed controls that are turning up, e.g. on today's audio equipment. Designers seem to think *rows of things* are desirable. On the contrary: the best designed controls I ever used are on the Sony TC-50 pocket tape recorder (see figure next column). But of course this is now phased out; instead most cassette recorders have *five or six* stupid buttons in a row. (Was it too good to last?)



Spurious control elegance comes in many guises. Consider Bruce McCall's description of the Tap-A-Toe Futuroidic Footless De-Clutching™ system. This was offered on the fictitious 1934 Bulgemobiles, and allowed you to drive the car with one pedal, rather than three (see below).

Careless and horrible designs are not all fictitious. One egregious example also indicates the low level of design currently going into some responding systems: computer people have designed CRT

ONE  
singular  
sensation—  
—“A Chorus Line”

**“Take new Tap-A-Toe Futuroidic Footless De-Clutching. Instead of old-fashioned gas, brake and clutch pedals that kept your feet busier than a dance marathon, Tap-A-Toe Futuroidic Footless De-Clutching offers the convenience of Single Pedal Power Control—combines all foot functions in one single pedal!”**

**“Think of it: one tap—you go, moving off faster than a barfly after Repeal.**

**“Two taps—you change gears, as smooth and automatic as a mortgage foreclosure.**

**“Three taps—you stop quicker than the U.S. economy.**

**“And that’s all there is to it. Tap-A-Toe Futuroidic Footless De-Clutching with Single Pedal Control is as easy and effortless as the Jap march on Manchuria!”**

Bruce McCall,  
“1934 Bulgemobile Brochure,”  
National Lampoon, May '74, pp. 76-77.



writing systems for newspapers which actually have a “kill” button on the console, by which authors would accidentally kill their stories. In a recent magazine article it was explained that the eventual solution was to change the program so that to kill the story you had to hit the “kill” button *twice*. To me this seems like a beautiful example of what happens when you let insulated technical people design the system for you: a “kill” button on the keyboard is about as intelligent as installing knives on the dashboard of a car, pointing at the passenger.

There is another poor tendency. When computer programmers or other technical people design particular systems without thinking more generally, things are not likely to be either simple or combinable. What may result is intricate user-level controls for one particular function, controls that are differently used for another particular function, making the two functions not combinable.

What makes for the best control structures, then? There is no simple answer. I would say provisionally that it is a matter of unified and conspicuous constructs in the mental view of the domain to be controlled, corresponding to a well-distinguished and clearly-interrelated set of controlling mechanisms. But that is hardly the last word on the subject.

## THE ORGANIZATION OF WHOLENESS

It should be plain that in responding screen-systems, “what happens on the screen” and “how the controls respond” are not really distinguishable. The screen events are *part* of the way the controls respond. The screen functions and control functions merge psychologically.

Now, there is a trap here. Just as the gas pedal, clutch, gearshift and brake merge psychologically, any control structure can merge psychologically. Clutch and gear shift do not have, for most of us, clear psychological relevance to the problem of controlled forward motion. Yet we psychologically integrate the use of these mechanisms as a unified means for controlling forward motion (or, like the author, get an Automatic). In much the same way, any system of controls can gradually come through *use* to have a psychological organization, even spuriously. The trap is that we so easily lose sight of arbitrariness and even stupidity of design, and live

with it when it could be so much better, because of this psychological melding.

But useful wholeness can be helped along. Just as what I have called the accordance-structure of writing (see “Writing,” p. DM 36) moves it along smoothly, fanciful design that builds from a well-organized internal dynamic should confer on a fanciful system the same momentum and clarity that carefully-organized writing has.

This contribution of wholeness can only occur, however, if the under-level complications of a system have been carefully streamlined and smoothed back, at least as they affect the user. Consider the design of the JOT text editing system (p. DM 71): while it is simple *to the user*, computer people often react to it with indignation and anger because it hides what are *to them* the significant features of computer text editing—explicit preoccupation with storage, especially the calling and revision of “blocks.” Nevertheless, I say it is the details at this level which must be smoothed back if we are to make systems for regular people.

## FANTIC SPACE

Pudovkin and Eisenstein, great Russian movie-makers of the twenties, talked about “filmic space”—the imaginary space that the action *seems* to be in.

This concept extends itself naturally to *fantic space*, the space and relationships sensed by a viewer of any medium, or a user in any presenting or responding environment. The design of computer display systems, then, is really the *artful crafting* of *fantic space*. Technicalities are subservient to effects. (Indeed, I think computer graphics is really a branch of movie-making.)

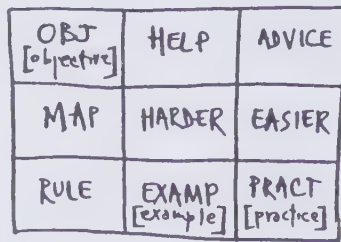
## FANTIC STRUCTURE

The *fantic structure* of anything, then, consists of its noticeable parts, interconnections, contents and effects.

I claim that it is the *fantic unity*—the conceptual and presentational clarity of these things—that makes fanciful systems—presentational systems and material—clear and helpful, or not.

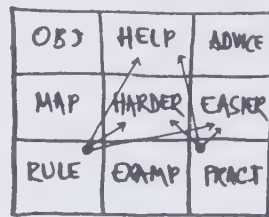
Let us take an interesting example from a system for computer-assisted instruction now under implementation. I will not identify or comment on the system because perhaps I do not understand it sufficiently. Anyway, they have an array

of student control-buttons that look like this:



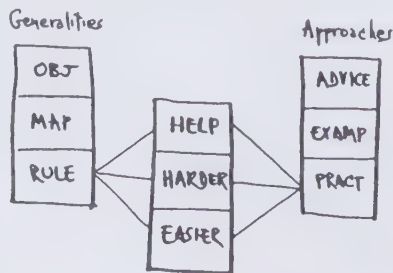
The general thinking in this system seems to be that the student may get an overall organizing view of what he is supposed to be learning (MAP); information on what he is currently supposed to be about (OBJ); canned suggestions based on what he's recently done (ADVICE). Moreover, he can get the system to present a rule about the subject or give him practice; and for either of these he may request easier rules or practice, or harder rules (i.e., more abstruse generalities) or harder practice.

For the latter, the student is supposed to hit RULE or PRACT followed by HELP, HARDER or EASIER, viz.:

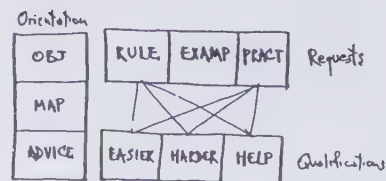


Now regardless of whether this is a well-thought-out way to divide up a subject—I'll be interested to see how it works out—these controls do not seem to be well-arranged for conceptual clarity. It seems to be the old rows-of-buttons approach.

I have no doubt that the people working on this system are certain this is the only possible layout. But consider that the student's options might be clearer to him, for instance, if we set it up as follows:



Or like this:



What I am trying to show here is that merely the *arrangement of buttons* creates different fanciful constructs. If you see this, you will recognize that considering all the *other* options we have, designing new media is no small matter. The control structures merge mentally with the presentational structures. The temptation to settle on short-sighted designs having shallow unity is all too great.

## FANTIC DESIGN

Fanciful design is basically the planning and selection of *effects*. (We could also call these “performance values”—cf. “production values” in movies.)

Some of these intended effects are simply the communication of information or cognitive structure—“information transfer,” to use one of the more obtuse phrases current. Other desirable effects include orienting the user and often moving him emotionally, including sometimes overwhelming or entrancing him.

In the design of fanciful systems involving automatic response, we have a vast choice among types of presentational techniques, tricks that are just now becoming understood. Not just screen techniques and functions, but also response techniques and functions.

(If “feelie” systems are ever perfected, as in Huxley's *Brave New World*, it's still the same in principle. See “Feelies,” p. DM 59, and “Wachspress,” p. DM 139.)

In both general areas, though—within media, and designing media—it seems to me that the creation of *organizing constructs* is the most profound problem. In particular, the organizing constructs must not distract, or tear up contents. An analogy: in writing, the inventions of the paragraph, chapter and footnote were inventions in writing technique that helped clarify what was being expressed. What we need in computer-based fanciful design is inventions which do not artificially chop up, constrain, or in-

terfere with the subject (see "Procrustes," p. DM 29).

I do not feel these principles are everywhere sufficiently appreciated.

But there is always something artificial—that is, some form of artifice—in presentation. So the problem is to devise techniques which have elucidating value but do not cut connections or ties or other relationships you want to save. (For this reason I suggest the reader consider "Stretchtext," p. DM 134, collateral linkage (p. DM 50), and the various hypergrams (p. DM 134). These structures, while somewhat arbitrary and artificial, nevertheless can be used to handle a subject gently.)

An important kind of organizing construct is the map or overall orienting diagram. This, too, is often partly "exact" and partly "artifice"; certain aspects of the diagram may have unclear import but clear and helpful connotation. (For instance, consider the wall on which the two pictures of Humpty Dumpty sit, p. DM 108. This too is a unifying fantic construct, even if we can't easily say what it means.)

Responding systems now make it possible for such orienting structures to be multidimensional and responding (cf. the orienting function of the "dimensional flip" control illustrated on p. DM 49).

Fantic design, then, is the creation either of things to be shown (writing, movie-making, etc.) at the lower end, or media to show things *in*, or environments.

1. The design of things to be shown—whether writing, movie-making, or whatever—is nearly always a combination of some kind of explicit structure—an explanation or planned lesson, or plot of a novel—and a *feeling* that the author can control in varying degrees. The two are deeply intertwined, however.

The author (designer, director, etc.) must think carefully about how to give *organization* to what is being presented. This, too, has both aspects: cognition and feelings.

At the cognitive end, the author must concern himself with detailed exposition or argument, or, in fiction, *plot*. But simply putting appropriate parts together is not enough: the author must use *organization constructs* to continually orient the reader's (or viewer's) mind. Repeated reference to main concepts, repeated shots

(in a movie) of particular locations, serve this function; but each medium presents its own possible devices for this purpose.

The organization of the *feelings* of the work criss-crosses the cognitive; but we can't get into it here.

Selection of points and parts contributes to both aspects. If you are trying to keep the feeling of a thing from being ponderous, you can never include everything you wanted, but must select from among the explicit points and feeling-generators that you have thought of.

2. The design of media themselves, or of media subsystems, is not usually a matter of option. Books, movies, radio and TV are given. But on occasion, as for world's fairs or very personal projects, we have a certain option. Which allows things like:

*Smellavision* or whatever they called it: movies with a smell-track, which went out into the theater through odor generators.

Branching movies (see p. DM 64).

"Multi-media" (multiple audio tracks and simultaneous slide projections on different screens).

Stereo movies.

And so on. The thing about the ones mentioned is that they are not viable as continuing setups for repeated productions. They do not offer a permanent wide market; they are not stable; they do not catch on. Which is in a way, of course, too bad.

But the great change is just about now. Current technicalities allow *branching media*—especially those associated with computer screens. And it is up to us now to design them.

3. MENTAL ENVIRONMENTS are working places for structured activity. The same principles of showmanship apply to a working environment as to both the *contents* of media and the *design* of media. If media are environments into which packaged materials are brought, structured environments are basically environments where you use non-packaged material, or create things yourself. They might also be called "contentless media." The principles of wholeness in structured environments are the same as for the others, and many of our examples refer to *them*.

The branching computer screen, together with the selfsame computer's ability to turn anything *else* on and off as selected by the user, and to fetch up information, yields a realm of option in the design of media and environment that has never existed before. Media we design for screen-based computer systems are going to catch on widely, so we must be far more attentive to the options that exist in order to commit—nationally, perhaps—to the *best*.

In tomorrow's systems, properly unified controls can give us new flexibilities. If deeply well-designed, these promise magnificent new capabilities. For instance, we could allow a musician to "conduct" the performance of his work by a computer-based music synthesis system (see "Audio," p. DM 60), perhaps controlling the many qualities of the performance on a screen as he goes, by means of such techniques as dimensional flip (see p. DM 49).<sup>\*</sup> (The tradition of cumulative audio synthesis, as practiced in the fifties by Les Paul and Mary Ford,

and more recently by Walter Carlos and Mike Oldfield, will take on a new fillip as multidimensional control techniques become common.)

One of the intents of this book has been to orient you to some of the possibilities and some of the options, considered generally. There is not room, unfortunately, to discuss more than one or two overall possibilities in detail. The most successful such system so far has been PLATO (discussed p. DM 93); others could not be listed for lack of space.

## NEW MEDIA TO LAST

What's worse, we are confronted not merely with the job of using computers to present specific things. The greater task is to design overall computer media that will last us into a more intelligent future. Adrift in a sea of ignorance and confusion, it is nevertheless our duty to try to create a whole transportation system that everybody can climb aboard. For the long run, fantic systems must be treated *not* as custom systems for explicit purposes, but as OVERALL GENERAL DESIGNS WHICH WILL HAVE TO TIE TOGETHER AND CATCH ON, otherwise collapse and perish.

## FINAL CONSEQUENCES

It seems to me certain that we are moving toward a generalized and universal Fantic system; people can and should demand it. Perhaps there will be several; but if so, being able to tie them together for smooth transmission is essential. (Think of what it would be like if there were *two kinds of telephones*?) This then is a great search and crusade; to put together truly general media for a future, systems at which we can read, write, learn and visualize, *year after year after year*. The initiatives are not likely to come from the more conventional computer people; some of them are part of the problem. (Be prepared for every possible form of aggressive/defensiveness from programmers, especially: "Why would you want that?" The correct answer is BECAUSE, dammit!)

But this all means that interior computer technicalities have to be SUBSERVIENT, and the programmers cannot be allowed to dictate how it is to behave on the basis of the underlevel structures that

<sup>\*</sup>I was not aware when I wrote this that William Buxton in Canada had been doing exactly this.

## The Twisted Smile ▼

You can make a character change expression on a 3D scope by making his mouth a twisted wire that can be rotated between "frown" and "smile" positions. The trick is the shape of the wire.



Can you guess the three-dimensional shape of this object?



are convenient to them. Quite the contrary: from the fullest consideration of the richest upper-level structures we want, we the users-to-be must dictate what lower-level structures are to be prepared within.

But this means you, dear reader, must develop the fantic imagination. You must learn to visualize possible uses of computer screens, so you can get on down to the deeper level of how we are going to tie these things together.

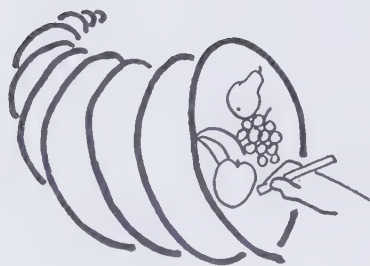
The designer of responding computer systems is creating unified setups for viewing and manipulating things—and the feelings, impressions and sense of things that go with them. Our goal should be nothing less than REPRESENTING THE TRUE CONTEXT AND STRUCTURE OF HUMAN THOUGHT. (Yes, Dream Machines indeed.) But it should be something more: enabling the mind to weigh, pursue, synthesize and evaluate ideas for a better tomorrow. Or for any at all.

#### BIBLIOGRAPHY

Theodor H. Nelson, "A Conceptual Framework for Man-Machine Everything." Proc. NCC '73.

Theodor H. Nelson, "Computopia and Cybercrud." In Levien (ed.), *Computers in Instruction*, The Rand Corporation, 1971.

IF  
COMPUTERS  
ARE THE  
WAVE OF  
THE  
FUTURE,  
DISPLAYS  
ARE THE  
SURFBOARDS.



## COMPUTER GRAPHICS

*Computer graphics is not a field. It is the frontier of every field. It is an ever-expanding pool of light, illuminating them all.*

*Computer graphics is thousands of techniques which may be combined in various ways; it is hundreds of commercial software packages which embody these techniques, unfortunately (as a rule) in incompatible ways; it is slicker and slicker hardware, a lot of it for the smaller personal-class machines (especially PCs, though the Mac II may be moving up fast); and it is hundreds of thousands of people sitting at screens.*

*When the original Computer Lib came out, most people were not familiar with wire-frame graphics and could not imagine image synthesizers; but now, thanks to TV titles and station breaks (where you fly through whirling logos) and car commercials (where you often see the wire-frame design of the vehicle), almost everybody has the idea.*

#### A NEW WORLD

*The computer graphics explosion is just beginning. What it will do to every field (and especially education) will be very interesting to see. In this section we will look at some of the techniques, and people, and stuff, of the field. There's no hope of keeping up. The field changes too fast; this is just a smattering. The pictures here are mostly from the OLD Computer Lib; new computer pictures you see every day.*

#### BIBLIOGRAPHY

Newman & Sproull, *Interactive Computer Graphics*. McGraw, \$15.

This is the textbook. Anyone interested in computer display should get this immediately.

An expensive journal, *Computer Graphics and Image Processing*, comes from Academic Press.

Sherwood Anderson, "Computer Animation: A Survey." *Journal of Micrographics*, Sept. '71, pp. 13-20. Lists nineteen computer-animation languages of that time.

Ken Knowlton, "Computer-Made Films," *Filmmakers Newsletter*, Dec. '70, pp. 14-20.

Issac V. Kerlow and Judson Rosebush, *Computer Graphics for Designers and Artists*. Van Nostrand Reinhold, 1986.

James D. Foley and Andries van Dam, *Fundamentals of Interactive Computer Graphics*. Addison-Wesley, 1982.

Computer Graphics World (subscription inquiries: (918) 831-9400) covers computer graphics more broadly.

#### SIGGRAPH!

To computer graphics people, Christmas and Saturnalia come in mid-summer, when the faithful converge on a large city to see the latest in equipment and algorithms. The conference is a week long and it's expensive (around \$25 to see the exhibits, around \$100 to hear the tech papers, around \$500 for a two-day tutorial on Monday and Tuesday). More for non members; but GO IF YOU CAN.

SIGGRAPH '88: Atlanta, August 1-5.  
SIGGRAPH '89: Boston, July 31-August 4.

THE SIGGRAPH TAPES are marvelous historical compilations of what was considered state-of-the-art in computer animation each year. They are available all the way back to the first collection (mid or late seventies).

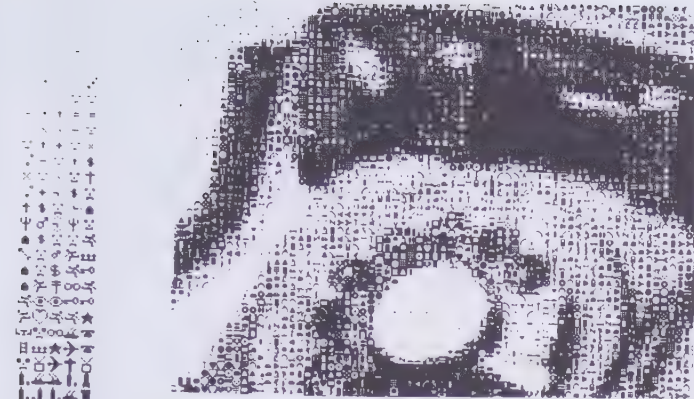
The SIGGRAPH VIDEO REVIEW (official title of the tapes) is numbered 1 to 27 so far. This is inane because in different years there have been one, two and three tapes. They are sold two "numbers" to a tape, except for #7. Cost to members: \$50 each. Cost to non-members: \$60 each.

Tragically, the SIGGRAPH PROCEEDINGS prior to 1982 are out of print and no longer being reprinted, even though much that's in them is classic and vital. SIGGRAPH CONFERENCE PROCEEDINGS, 1982-current are \$30 each (\$40 to non-members).

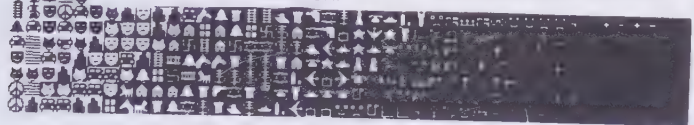
Back tapes and proceedings may be ordered on your charge card (800) 342-6626, or prepaid from ACM, Inc., Order Dept., P.O. Box 64145, Baltimore, MD 21264. (Handling charge: \$6 on orders up to \$100, \$8 above that.)

(An old piece of mine in the late lamented Creative Computing describes the madcap atmosphere of SIGGRAPH: T. Nelson, "Smootherers of the Lost Arc," March 1982.)

There is also the NCGA conference, similar but slanted to laymen in the industry, held in the spring. National Computer Graphics Association, 2722 Merrilee Drive, Suite 200, Fairfax, VA 22031.



Knowlton and Leo Harmon have done a lot of experiments with picture conversion (see bibliography). Here is a phone made into teeny patterns (shown around). © Knowlton & Harmon.



## REFOCUSING

*This is pretty startling. An evenly blurred photograph can be re-focussed by computer. This is done for satellite pictures. It takes a lot of crunch, though.*



# PICTURE PROCESSING

"Picture processing" is an important technology, largely separate from the rest of computer graphics. It means taking an incoming picture, usually a photograph, and doing something to it. (Some now call this area "computer pictorics.")

First of all, there is *image enhancement*. This means taking pictures, dividing them into points whose brightness is separately measured, and then using special techniques for making the picture *better*. To people familiar with photography, this may seem impossible; to photographers it is a maxim that photographs always lose quality at each step. Nevertheless, various mathematical tech-

niques such as Fourier Analysis (mentioned elsewhere) do just that, producing *new* data structure improving on the original data. Surfaces appear smoother, edges sharper.

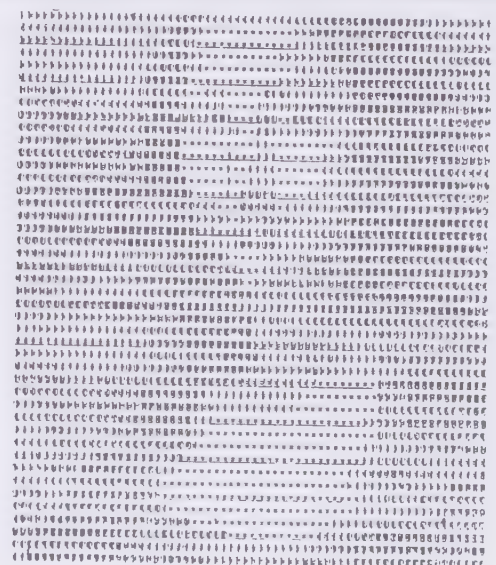
(These techniques have been extensively used to clean up photographs sent back from our unmanned space vehicles—both those used exploring other planets and those spying on our own—see *Secret Sentinels in Space*, Bibliography.)

Then there are *recognizers*—programs that look at the data structure from an input picture, and try to discern the lines, corners and other features of the

## ► Lizzie Of The Lineprinter

A famous converted picture. The painting was divided into 100,000 brightness-measured spots by H. Philip Peterson of Control Data Corporation; then each dot was made into a square of overprinted letters on the printing device. The program allowed 100 levels of grey. Left: Control Data's version, reprinted by permission. Below: a cut-down version that often turns up. (From original flat 2D artwork by Len da Vinci of Medici Associates.)

NOTE: this is not a "computer picture." There is no such thing. It's a quantization put out on a lineprinter.



(Note what Lillian Schwartz has to say about the real connection between Lizzy and Lenny. p. DM 55.)



picture. (While your eye instantly sees these things, computers do *not*, and must look at the dots of a picture one-by-one. How to analyze pictures in such tedious sequences is no simple matter.)

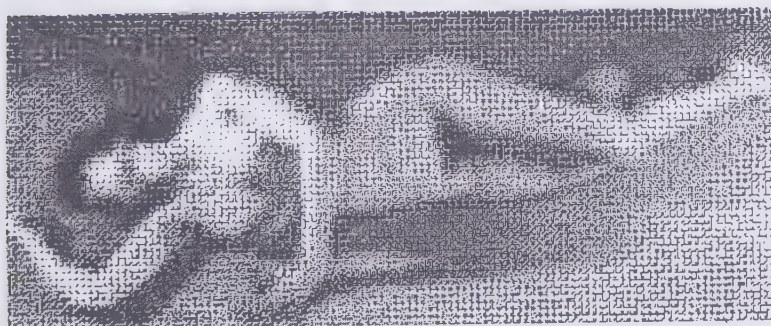
For recognizing more complex objects in pictures—boxes, spheres, faces or whatever—more complex structure-analyzing programs are necessary. As the possibilities of what might be in a picture increase, these increasingly become *guessing* programs. (This becomes a branch of *artificial intelligence*, a misleading term for a curious field, discussed on p. DM 120.)

Numerous computer people think it is

important to match up our computer graphic display systems (described variously on this side of the book) to image input systems. This is a matter of taste.

These are all basically techniques for making a *data structure*. Any data stored in computers must have, of course, a data structure, which basically means any arrangement of information you choose. (see p. 70.)

These various techniques are intended to create *reduced* data structures, recording only the “most important” data of the picture—from which new and varying pictures may be created, reflecting the “true” structures originally shown in the



This is a non-simple picture conversion. The original photograph was converted into measured points; but these were in turn made into grow-together patterns by a program in the EXPLOR language. © Knowlton & Harmon

## A Titanic Problem●

When they sent a sub down to explore the Titanic, the cable was so long that the images looked like snowstorms from the noise; the naked eye could see nothing.  
By image-processing techniques, however, it was possible to filter out the noise and get those extraordinary pictures.

initial picture. How much it's going to be possible to create these data structures from input pictures remains to be seen; some of us think it's not going to be generally worthwhile.

## BIBLIOGRAPHY

Azriel Rosenfeld, “Progress in Picture Processing 1969-71.” *ACM Computing Surveys* June '73, pp. 81-108.  
Ken Knowlton and Leon Harmon, “Computer-Produced Grey Scales.” *Computer Graphics and Image Processing*, April '72, pp. 1-20.  
Philip J. Klass, *Secret Sentries in Space*. Random, 1971, \$8. Interesting general book on geopolitical strategy and orbital photoreconnaissance. “Now-it-can-be-told” approach.

Benjamin M. Dawson, “Introduction to Image Processing Algorithms,” *Byte*, March 1987, p. 169.

## SATELLITE PICTURES OF YOUR HOME COUNTY, OR WHATEVER

You can get pictures of any area you want from ERTS (Earth Resources Observation Systems) satellites, from EROS Data Center (no, not a dating service, see p. 154), Sioux Falls, SD 57198, or call 605/594-6511 bet. 7 AM & PM central time.

Better satellite pictures can now be ordered from the French public-access satellite.

## COLORIZATION OF OLD MOVIES

Much indignation has been heaped on the colorization of old movies (such as “High Noon”) by Ted Turner and his satellite network.

Aesthetics aside, it's a complex process, and couldn't have been done before computers.

An artisan sits at a display screen and indicates to the computer which areas of the picture are to be flesh-tone, which green, which blue, and so on. (This would be easy to do by hand for a single frame; in fact, before cheap color prints, it's how color photo portraits were usually made—using “photo-oil colors.” But the problem is to get the same colors *evenly and correctly spread on all the frames of the scene.*)

Thus the computer must watch, in succeeding frames, where the same areas move to, note their boundaries, and spread the color correctly. Various pattern-analysis techniques must be built into the software.

The artisan, of course, has the last say: by watching the colorized scene, she or he can tell if the colors have been spread about appropriately.

It would be nice to see it done by artists with a flair. I suspect that the results could be pretty neat, especially on scenes where there's a fair amount of leeway as to what color things might be.

# SLICE OF LIFE

X-rays, discovered around the turn of the century, allowed us to see into the human body—but the view had to go through the whole body at once. Suddenly, however, a host of new views have become possible. New software and equipment provide more and more new windows into the body.

The CAT scanner (Computer Axial Tomography) allowed apparent X-ray slices of the body to be created. This worked by placing the patient on a sled that passed through a ring. (“Axial” means the patient is conceptually being turned on a spit.) In the ring are aimable X-ray guns and arrays of X-ray sensors. Under program control, fine beams of X-rays slice back and forth as the ring turns; the pro-

gram examines the incoming signal and infers the picture. Some areas may be aliased or blocked, but the degree to which a sharp picture may be inferred is astonishing; the result is essentially a *slice* of the brain, the neck, the chest, whatever, revealing tumors, enlargements and other conditions in remarkable detail.

The CAT scanner opened the door. Now a variety of other devices make a variety of other slices: PET, NMR, MRI, SPECT.

Most amazingly, the Lucasfilm PIXAR research (see pp. DM 113-114) turned up new processes that show muscles and organs photographically, as if the covering tissue had been removed.



# CATCH AS CAD/CAM

Perhaps the dumbest term in the computer field is CAD/CAM, which simply means **designing things on screens**. But they whopped up the term "Computer Aided Design" for making drawings, and then three-dimensional structures of things to be built, on the screen. Then when they realized the data could be taken to a machine shop to grind out automatically the thing designed, they had to have **another** name, "Computer Aided Manufacture." **Then** when they figured (duh) that the systems were really one, they called it CAD/CAM. Now they're suggesting a simpler term, CIM (Computer Integrated Manufacturing). But it seems to me that the most sensible thing to call it all is simply "screen design." Although **desktop engineering** is another popular phrase with merit.

## CAD

CAD comes in two main types: 2D (where you draw engineering pictures), and 3D (where you make shapes to be built). But the **additional features** are what make it expensive: circuit simulation (after you've drawn the circuit, seeing how it would behave); calculating moment of inertia for moving parts; and on and on.

Many CAD users now want fancy image synthesis systems so they can make fancy pictures of their designs before they're built, to sell them to sponsors—pictures of "TV commercial quality." The software is available, expensively.

Many serious CAD programs are now available for the PC and the Macintosh. However, my own favorite is still MacDraw (see "Sketchpad," p. DM 96), which is extremely simple and elegant.

## CAM

The fact is that computers are now being set up to give new flexibility to manufacturing processes. Computers, directly connected to milling machines, grind metal into any conceivable shape much faster than a human craftsman. To change the result, change the program—in a fraction of a second. Fabric design has been done on computer screens; the

obvious next step is to have the computer control the loom or knitting machine and immediately produce whatever's been designed.

## CAM AND CUSTOMIZATION

Custom clothing: soon we may look forward to tailoring services that store your measurements and can custom-tailor a suit for you to any new fashion, in minutes. (But will the price beat Hong Kong?) Customized printed matter is already here. Wherever people want individual variations of a basic manufacturing process, computers can do it.

Speaking of CAD/CAM, here is remarkable work that combines 2D and 3D—the living-hinge geometry of Ron Resch.

than construction gangs; they could be transported and stored flat. Their surface-to-volume ratios couldn't be beat.

Noting this, an architect named Ron Resch said to himself: What about making large-scale *foldable* structures, like- unto geodesic domes, that could be simply manufactured in sheet form and creased at the factory, then bolted and cabled and strutted in the field?

Resch has now for years been experimenting with complex folded structures.

There's only one trouble. If you've messed with paper airplanes, you know that folding is an inaccurate process, and so the prospect of discovering complex geometric structures by the hand-folding of paper is rather slim.

Recognizing this, Resch has contrived to work at a computer display. His work—the search for great folding structures—is one of the first practical uses of halftone polygon computer graphics. He

is, naturally, at the University of Utah.

Resch has invented some wonderful and profound structures that can be made with pleated paper, folded in different ways. the problem is it's not published (last I heard). He has wonderful movie footage, which ought to be on video tape for everybody, of some truly astonishing repeated structures he has discovered, with potential for building and other forms of engineering.

Ron may be best known in Canada for a huge Easter Egg that hangs over a small town in some western province. He won a contest and erected the egg by his folding methods, and now it turns in the wind.

## WHERE ARE THEY NOW?

*Restless Resch is hard to follow. One rumor has it that he became a follower of an Eastern mystic; another, that he is collaborating with Douglas Hofstadter on a pop-up book.*

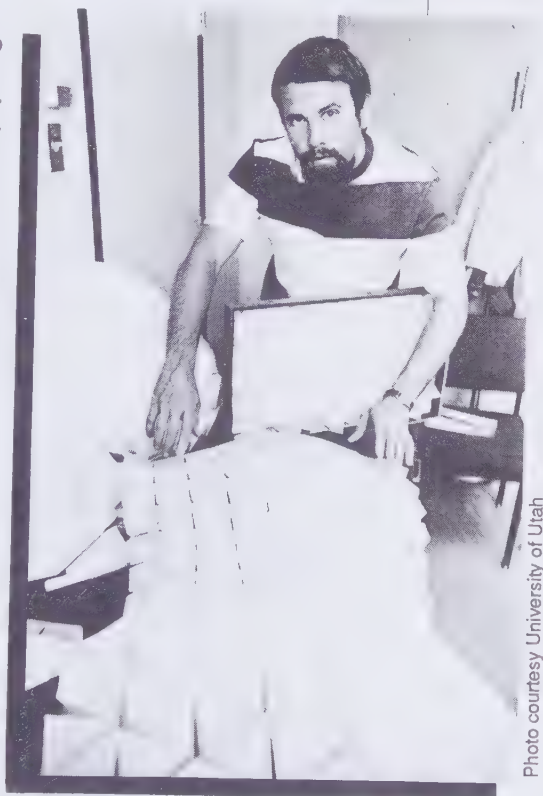


Photo courtesy University of Utah

As we take the gee-whiz out of all these processes, and simply accept that work in the future will be at screens, terminology should get a lot simpler and more straightforward.

**F**or a major fair, Bob Abel studios created a beautifully simple example of CAD: you could design a roller coaster on the screen—then ride it. An Evans and Sutherland Flight Simulator would take you for a ride on the roller coaster of your own devising—complete with sound effects. It was great.

*"Mask Parade," a program for the Apple II (Springboard Software, Minneapolis), is for making "masks, bats, ties, badges and feet." Wear them after you print them out.*

*You may now negotiate your new nose with the plastic surgeon, designing it in 3-D on a PC screen. (Making sure the result is up to spec, after the bandages are off, is a different story.)*

## RETURN TO THE FOLD

There was a lot to be said for tents. They could be made by tailors; rather



# LIGHTNING IN A BOTTLE: THE CATHODE-RAY TUBE

A cathode-ray tube is actually a bottle filled with a vacuum and some funny electrical equipment. The equipment in the neck of a bottle shoots a beam of electrons toward the bottom of the bottle.



This beam of electrons is called, more or less for historical reasons, a *cathode ray*. Think of it as a straw that can be wiggle in the bottle.

Actually the bottle is shaped so as to have a large viewing area at the bottom (the screen), and this screen is coated with something that glows when electrons hit it. Such a chemical is called a *phosphor*.



or wiggle a straw in a coke bottle. The beam can be moved with either magnetism or static electricity. This is applied in the neck of the bottle—or even from outside the neck—by *deflection plates*, whose electrical pulsations determine the pattern the beam traces on the screen. (Note that the beam can be moved on the screen at great speed.)

The *vertical* deflection plates can pull the beam up or down on the screen, controlled by a signal to them;



the *horizontal* deflection plates can pull the beam sideways on the screen, controlled by a signal to them.



By sending combined signals to both horizontal and vertical deflection plates, we can make the end of the beam—a bright dot on the screen, sometimes called a *flying spot*—jump around in any pattern on the screen. A repeated pattern of the beam on the face of the CRT is called a *raster*.

From these two capabilities, brightening and moving the beam, a number of very special technologies emerge:

TELEVISION uses a zigzag scanning pattern which repeats over and over. This zigzag pattern is always the same, night and day.

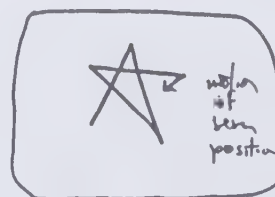


You can usually see the lines clearly on a black-and-white set. The picture consists of the changing pattern of brightness of this beam, which comes in over the airwaves as the television signal.

RADAR DISPLAY uses a CRT to show reflected images around where the radar antenna is standing. This uses a scanning raster of a star shape, brightening the beam when reflected images are received.



COMPUTER CRT GRAPHICS generally use the CRT in still another way: the beam is moved around the screen in straight lines from point to point. (Between different parts of the picture the beam is darkened, turned very low so you don't see it.)



Because the image on a normal CRT fades quickly, the computer must ordinarily draw the picture again and again

and again. (Methods for this are discussed on pp. DM 86-93).

## SPECIAL KINDS OF CATHODE-RAY TUBES

The CRT is not merely a single invention, but an entire family of inventions. The *ordinary* CRT, which we have discussed, is viewed at one end by a human being, has an image which fades quickly, and can have its flying spot driven in any kind of raster or pattern.

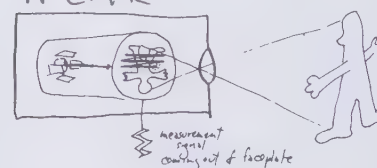
Here are some other kinds of CRT:

The *picture transmitter*, which has different versions and names: Vidicon, Image Orthicon, Plumbicon, etc. THIS IS THE MAGICAL DEVICE THAT MAKES THE TELEVISION CAMERA WORK, AND YET, BY GOSH, IT'S JUST ANOTHER CRT. Except, instead of the picture coming *into* it as an electrical signal and *out* of it as an optical image, the picture comes *into* it as an optical image and goes *out* of it as an electrical signal.

How can this be?

The tube sits inside the television camera, which is an ordinary camera, like, with a lens projecting a picture through a dark chamber onto a sensitive surface. But instead of the surface being a *film*, the surface is the faceplate of a CRT with some kind of a special pickup phosphor:

## TV CAMERA



The electron beam, which is just like any other electron beam, is made to zigzag across the faceplate in a standard television raster. And the special phosphor of the tube *measures the brightness of the picture* at the spot the beam is hitting. I have no idea how this happens, but it's chemical and electrical and mysterious, and is based on the way the phosphor interacts with the light from one side and the electrons from the other side at the same time. *Anyhow*, a measurement

## NTSC: THE GREAT RASTER SCANDAL

Unfortunately you can only get about 200 good scan-lines out of NTSC, our national video standard. And that's why we have to buy expensive monitors.

signal comes out of the *faceplate*, indicating how bright the projected picture is in the *very spot* the electron beam is now hitting.

As the beam criss-crosses the faceplate in the zigzag television raster, then, a continuously changing output signal from the faceplate shows the brightnesses all across the successive lines of the scan.

And *that* is the television signal. Together with synchronizing information, it's what goes out over the airwaves, down your antenna and into your set. Your set, obeying the synchronizing information, brightens and darkens its *own* beam in proportion to the brightness of the individual teeny regions of the faceplate in the television camera. And this produces the scintillating surface we call television.

The *color tube* is a weird beast indeed. There are several types, but we'll only talk about the simplest (and many think the best), Sony's Trinitron™ tube.

This is an ordinary CRT which has, instead of a uniform coating on the faceplate, tiny vertical stripes of three primary colors—red, blue, and green. (You thought the primary colors were red, blue, and yellow, didn't you. If you're mixing pigments that happens to be true. For some ungodly reason, however, if you're mixing *lights*, the colors that yield all others turn out to be red, green, and blue; it turns out that yellow light can be made out of red and green. If you don't believe me, go to a chintzy hardware store, get a red and green bulb, turn 'em on and see what happens in a white-walled room.)

At any rate, color television uses additional color signals, and in the Trinitron, these control the response of the faceplate. If the color signal says "green" as the electron dot crosses a certain part of the screen, the color signal tells the *green* stripes that they're free to light up when hit. If it's Yellow Time, the signal tells both the red stripes and the green, and so side by side they light up red and green, as the beam crosses them, but the total effect from more than a few inches is Yellow.

Most American color TV sets, however, at least up till this year, used something very different, something entirely weird called the Shadow Mask Tube. I'll spare you the picture, but there were several *different* electron beams—often referred to jokingly as the "red electron beam," "blue electron beam" and "green electron beam," though of

course they were identical in character. These hit a perforated sieve, up near the screen, called the *shadow mask*, and the color signal tweaked the unwanted beams so they did not hit *different-colored phosphor dots* that were intricately arranged on the screen. I'm sorry I started to explain this.

*Multigun tubes* have more than one electron gun and more than one electron beam. They can be used in different ways (aside from the old shadow-mask TV tube, mentioned above).

For instance, one gun can be driven in a video raster, to show television, while another gun can be used as a computer display, drawing individual lines with no regard to the TV pattern.



The *storage CRT* comes in two flavors: viewable and non-viewable. But what it does is very neat: it *holds the picture on the screen*. The mechanisms for this are of various types, and it's all weird and electronic, but the idea is that once something is put on the screen by the electron beam, it stays and stays. Up to several minutes, usually. The main manufacturers are Tektronix, Princeton Electronic Products and Hughes Aircraft; each of these three has a product that works by a different method.

Note: Tektronix' tube is built into a number of different computer displays, and is recognizable by its Kelly green surface. They themselves make complete computer terminals around this scope for \$4000 and up, but lots of other people put it in their products also. It shows whatever has already been put on the screen, and the electron beam does not have to repeat the action. However, it usually only stays lit for about a minute.

Princeton Electronic Products (guess where) is a much smaller outfit, so perhaps it is appropriate that they make a much smaller storage tube. It is about one inch square at its storage end, and you *don't* look at it directly. Instead, an image can be stored on it *either* with a TV raster or by computer-driven line drawing. After the image is stored on it, though, it functions as a *TV camera*: the picture stored on the plate can be read out with a scanning raster, exactly as if it were a pic-

ture transmitter in a television camera. The Princeton folks have built a quite expensive, but quite splendid, complete terminal around this device: it can hold both video and computer-drawn pictures, superimposed or combined, and sends them back out in standard black-and-white TV. \$12000.

CRTs which bring in a picture one way (such as video raster) and send it back out another way (such as by letting a computer search out individual points) are called *scan converters*.

A word about this last method. It is often desired by computer people to turn a picture into some form of data (see pp. DM 81-82). Scan converters, usually by the three manufacturers named above, can be hooked up to let the computer program poke around in the picture and measure the brightness of the picture in arbitrary places. A device which examines the brightness of something in arbitrary places is called a *flying-spot scanner*. Below are some different kinds of flying-spot scanners.

I have heard it said that it might be possible to build a CRT with a changeable mirror surface: that is, the screen becomes *mirrored* temporarily where it is being hit with the electron beam. Interesting. This would mean that you could make computer displays (and TV) bright

and projectable to any degree, say, by pouring a super-intensity laser beam on it. "Be great for writing 'Coca-Cola' on the moon," says a friend of mine. If you believe in astral projection.

## BIBLIOGRAPHY

*Color TV Training Manual*, Sams & Co./Bobbs-Merrill (\$7), is a well-illustrated and intelligent introduction to the TV use of CRTs.

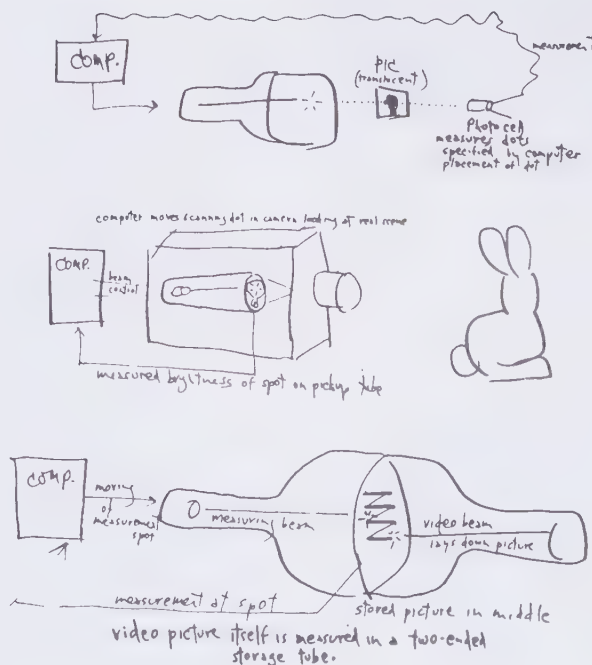
## HOW MUCH BETTER CAN CRT'S GET?

*CRT displays—that is, the monitors themselves—have gotten up to levels of resolution we did not think would be possible a decade ago. Thousand-line CRTs can be bought off the shelf. State-of-the-art workstation displays (such as the Sun) now have millions of points—1200x1600 in black and white, 1200x800 in the color version.*

*How much better can they get?*

*This is an important question because it determines how much better our workstations can get by evolution—without jumping to another technology.*

*A fair guess is that we can get to at least four times as good: say, eight million pixels black-and-white, four million color. Which will look damn good.*

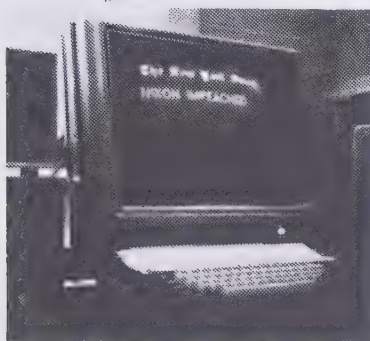




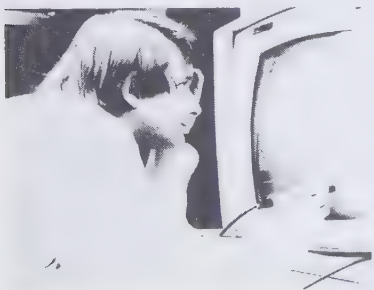
## STANDARD HARDWARE?

There is definitely no standard computer-graphic hardware—except, at the personal-computer level, for the screens of the Macintosh and PC (CGA board)—and so graphic software is usually written in generalized form, able to output to whatever display or other devices are available.

On our glorious Dig-It-All Screens we mingle the magics of air and of fire.



Responding computer displays come in all sizes and prices. This little setup (in the under-\$10,000 class) is a PDP-8 minicomputer with home-built display circuitry. Gothic lettering data structure available from somebody in the military; message courtesy of R.E.S.I.S.T.O.R.S. The big display is an IBM 2250 (over \$100,000, including minicomputer).



# THE MIND'S EYE

It was explained on the other side that computers have no fixed purpose or style of operation, but can be set in motion on detailed and repetitive tasks in any realm of human interest—as long as those tasks are exactly specifiable in certain humdrum ways.

Now, if you had a machine like that burning a hole in the corner of your office, what would you *really* want to do with it?

You can't drive it on the road.

You can't make love to it. (But see p. DM 139)

You can't cook in it, or get the news on it.

To get it to control elaborate events in the real world requires a lot of expensive equipment and interfaces, so cross that out.

Yet suppose you have an inquiring imagination—which is not unlikely, considering that you are reading this sentence.

And we are also supposing (from an earlier paragraph) that you *have* a computer.

What sorts of things would you do with it?

Things that are imaginative and don't require too much else.

I am hinting at something.

YOU COULD HAVE IT MAKE PICTURES and show you stuff and change what it shows depending on what you do;

and if this idea doesn't turn you on, the rest of the book is probably not for you.

The techniques of making pictures by computer are called computer *graphics*.

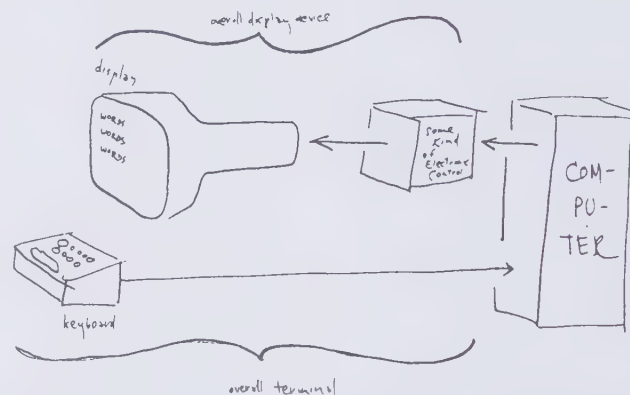
But that includes the dull kinds of making pictures by computer, the ones that do it with pens and printing machines.

The techniques of making computers present things interactively on screens is called computer *display*. (Some say “interactive computer graphics”; this is not just too long, but too restrictive as well: interactive *text* systems are not “graphic” or pictorial, but they are going to be a pro-

foundly important area of computer display.)

(Incidentally, the silly word “interaction” was coined because the previous word “intercourse,” which meant exactly the same thing, had racy connotations for some people. Cf. “donkey” and “rooster,” also relatively recent.)

You will note that computer display is what makes possible the computer terminals with screens that we saw earlier and on the other side. All that a screen-terminal is, is *some sort* of computer display to which a keyboard has been added.



You will therefore see that to understand all the different computer display terminals, you would have to understand all the different computer display techniques; unfortunately, we can only cover a few here, and those but sparsely.

Some of the types of computer display to be covered hereabouts include:

*CRT, or cathode-ray tube*, displays; these are my favorite because the stuff on their screens may be animated by the computer.

*video* displays, which use television techniques. These have troubles deriving from the way a TV picture is timed.

*panel* displays, i.e., those which appear on a flat panel. These are going to be cropping up all over. (The pictures can't move much, but the devices are going to be cheap. Flat, too. Some people think that's very important.)

*3D* displays, especially of the CRT type. NOTE: this term refers ambiguously to two different things: setups

which present *flat* views of three-dimensional scenes, and those which present *stereoscopic* views of 3D scenes; these are much rarer.

*image synthesis* or *halftone* techniques and systems. These are computer programs and special devices which make shaded or photograph-like pictures. (This happens to be a favorite topic of mine, and so there's quite a bit on it here, a lot of which is not widely known in the field.)

## YOUR BASIC TYPES OF COMPUTER DISPLAY

(Note: the term “display” is also used in this field to refer to numbers and letters that can be made to light up in fixed positions, like on your pocket calculators. Those will not be discussed here.)

## THE FORKED LIGHTNING

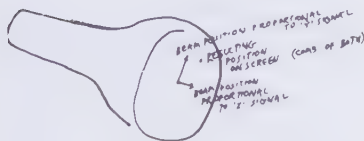
“Because their words have forked no lightning they  
Do not go gentle into that good night.”  
—Dylan Thomas

The most basic, and yet eventually the most versatile, computer display is that of the CRT, or bottled lightning (as I like to call it). It is, you know: a beam of electrons, just like lightning in a storm, but from the neck of a very empty bottle to its flat bottom, whose chemically coated surface we watch. As manipulated by the computer, the CRT stabs its beam to all corners of the faceplate: forked lightning.

Computer display began in the late forties. Computers themselves were completely new, and so was Mr. Dumont's magical Cathode Ray Tube or CRT (see p. DM 84), developed on a crash basis during the war so we could have radar, and as long as it was around after the war, we got television.

But the lightning bottle, or CRT, can be used in a variety of ways. Its control plates, which move the ray of electrons around on the screen, can be given various different electronic signals, causing the beam to move around in different patterns. In normal video, the signals move the beam in a zigzag pattern, where the zigs are very close together and the zags are invisible; the carpet of zigs covers the screen over and over in a repetitive pattern, and the beam's changing intensity paints the picture.

But we can drive the CRT differently, by using different control signals. For instance: We can apply a measured voltage to the height or "Y" plates of the CRT, moving the beam to a given vertical position, and another measured voltage to the sideways or "X" plates, controlling its horizontal position.

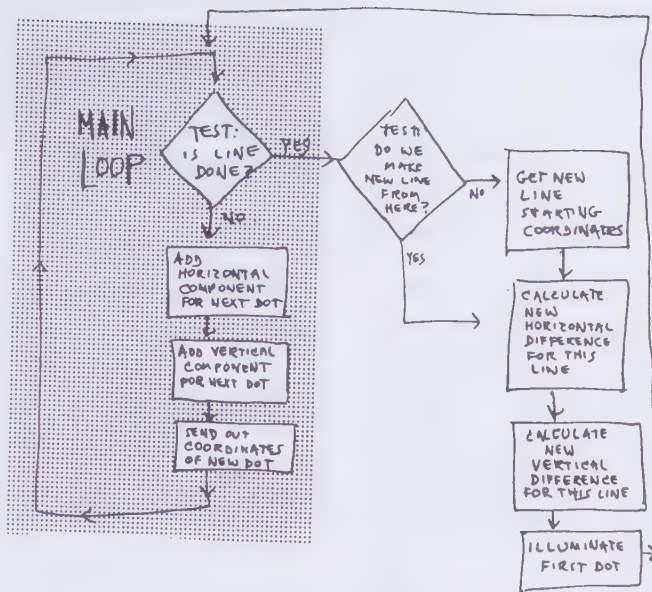


## 1. EARLIEST SYSTEM: A LITTLE PROGRAM TO MAKE DOTS

The earliest setup connected a CRT to a computer by the simplest possible means, and made its pictures with dots on the screen—a sort of tattooing process.

It was simple because all the computer did was furnish to the connecting circuitry (or *interface*) symbols specifying how far up, and how far across the screen, the next dot should be. These symbols were actually coded numbers, and the interface turned them into voltages which then moved the beam correspondingly. (This process of making a measured voltage out of a coded numerical symbol is called *digital-to-analog conversion*, since (as explained on the other side) the main meaning of "analog" these days is "in a measured voltage.")

Now, this has several drawbacks. One is that the lines are dotted; nobody likes that. A more important annoyance, though, is that the computer *scarcely has time for anything else*. Here is a flowchart of what the computer has to do in its program. (Even if you didn't look at the other side of the book, flowcharts are nothing scary. They're just maps of what happens.)



Furthermore, and here was the indignity of it, this system took far too long. To draw a line with thirty dots in it took thirty times around the main loop in the flowchart, and since each box in the flowchart takes at *least* one of the machine's rock-bottom instructions—usually more—then the main loop of this display routine takes four separate operations *per dot*, or 120 operations for a stupid 30-dot line. Plainly, there has to be a better way to use an expensive computer.

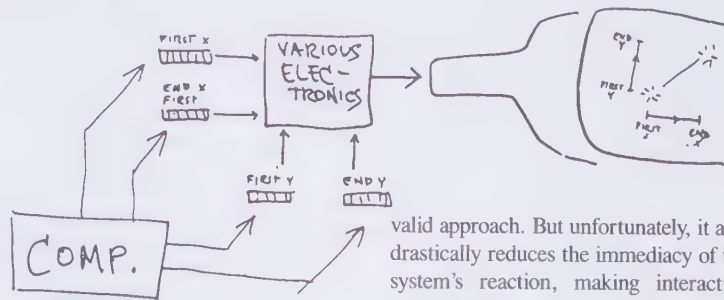
Actually it wasn't just the ignominy of it, but the fact that it took so *long*, that made this a poor method. The amount of stuff the computer could draw in  $\frac{1}{40}$ th of a second—and this turns out to be how fast the whole picture has to be made—was too little. After  $\frac{1}{40}$ th of a second the human eye can see the lines on the CRT start to fade, and so the picture has to be *redrawn* to make it bright again before that happens. If your eye sees the picture fading, then when the computer draws the picture again you will see it get suddenly bright again—and it will start to flicker. This is distracting, unhealthy, and disagreeable.

Note that the most important computer in the history of computer display used this technique (I think). This was the TX-2 at Lincoln Laboratories, a highly-guarded installation outside Boston that is formally part of MIT. The TX-2 was one of the first transistorized computers—perhaps *the* first; and on it were programmed a number of milestone systems, including Sutherland's SKETCHPAD, Johnson's SKETCHPAD IV, and

Baecker's GENESYS animation system (discussed on p. DM 56).

## 2. LINE-DRAWING HARDWARE

The next step in design is to get the computer program out of the business of drawing lines by a succession of dots. So we build a piece of hardware that the computer program may simply instruct to draw a line. As an interface, it looks to the computer like four separate devices: registers that tell where on the screen the line must start ("first X" and "first Y") and registers that tell it where to stop ("end X" and "end Y").



valid approach. But unfortunately, it also drastically reduces the immediacy of the system's reaction, making interaction with the system less intimate and wonderful.

Approaches which put display refreshment and maintenance in a separate device are less interesting to me, and so that discussion continues separately nearby. ("Display Terminals," pp. DM 10-11)

## 4. THE SECOND PROGRAM FOLLOWER

On the other side of the book, I explained that a computer is basically a

(Incidentally, it is a puzzling fact that such a device is available nowhere, although lots of people end up building one for themselves. There was such a thing on the market a couple of years ago—line-drawing hardware with no interface and no CRT—but it was withdrawn because of reliability problems.)

## 3. EVOLUTION FROM THIS: TWO OPTIONS

There are basically two ways to go from this basic starting point. *Either* we can keep the display device intimately and integrally connected to the computer, or we can say the hell with it and cut the display device loose as a separate entity.

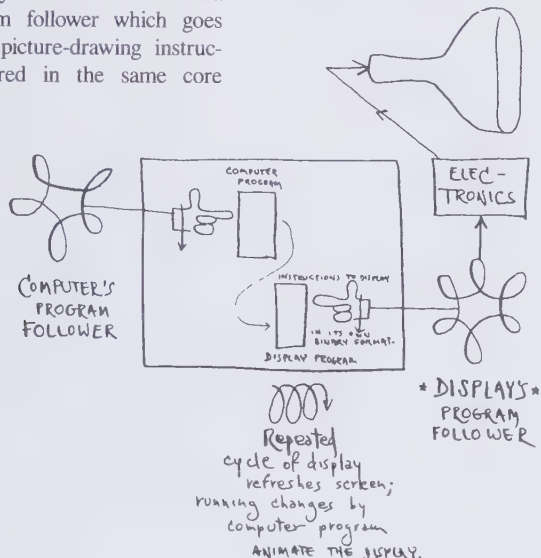
Ivan Sutherland has cannily noted that there is a certain trap involved in these designs: as we build additional "independent" structures to take the burden of display away from the computer, we are tempted to keep adding features which make the "independent" structure a computer in its own right. This paradoxical temptation Sutherland calls "the great wheel of reincarnation" of computer-display architecture.

It is tempting to cut the display loose from the computer. It means the computer can be fully occupied with other matters than refreshing the screen—preparing the next displays, perhaps. Many computer people believe this is the right way to do it, and it is certainly one



zippy device, never mind how constructed, which follows a program somehow stored symbolically in a core memory. Such a device we call here a *program follower*. While programs may be in many computer languages—all of them contrived systems for expressing the user's wishes, in different styles and with different general intent—underneath they all translate to an inner language of *binary patterns*, which may just be thought of as patterns of X and O, or light bulbs on and off. The innermost program follower of the computer goes down lists of binary patterns stored in the core memory, and carries them out as specific instructions. It also changes its sequences of operations under conditions that the programmer has told it to watch for.

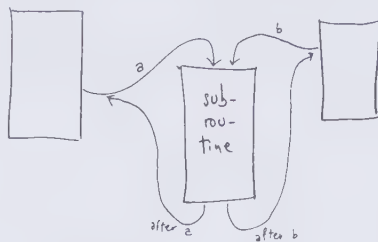
The most powerful and responsive computer displays are those which build a *second* program follower which goes down lists of picture-drawing instructions also stored in the same core memory.



We may call this also a "list-of-lines" system, since the commands recognized by the display program follower are typically patterns that tell it what lines to draw.

Typically also it has its own way of jumping around in a program, and may jump to a specific list of lines, or subpicture, from numerous other parts of its program, always returning each time to the point from which it had jumped. This allows the same subpicture to appear in numerous places on the screen at the same time. (A program that can be jumped to by other programs which then resume operation is called a *subroutine*; thus the real, or most prestigious, name

for such a device is a *subroutining display*.)



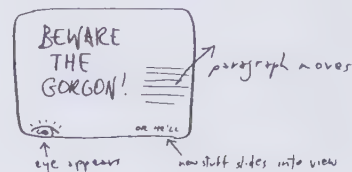
This design has some extraordinary advantages. One is that since the computer's program follower and the display's program follower both share the same core memory, they can work together most intimately. When the user demands something new—by typing, say, or pointing with a light-pen—the computer

Most importantly, the computer can move images on the screen, allowing interactive animation on the screen under the user's control. Each time the display is about to show the same picture again, the computer simply supplies it with a *new starting point*. Since the list of lines is typically in the form of sequences of lines relative to one another, the picture is drawn in a new place each time—and thus seen to move on the screen.

Finally, the computer itself is free most of the time—free, that is, to do other things, which typically is always desirable. Just *how much* the computer can or should do in such a partnership is a matter of dispute. (Ordinarily such devices are spliced onto minicomputers; and minicomputer fans, such as the author, see no reason not to perform all services for the display there in the minicomputer—and a pox on the big machines. Others, for various reasons, see the subroutining display and its host mini as needing the tender ministrations of a big computer via some sort of communications line. There are various reasons for holding this entirely legitimate view. People who are devoted to the high number-crunching capacity of big computers, or to languages which require great big computers to run in, have a right to their opinion. Moreover, it is currently feasible to store large bodies of data only on big computers—not because big disk and tape memories can't be easily attached to the small ones, for they can, but they usually aren't; and other ways to tie minicomputers to big stores of data aren't available yet.)

Subroutining displays often have commands allowing them to display *text* as well as lines and dots. In the display of text, they can use the same technique of "moving the picture" by starting its display at successively creeping points; this will cause, say, whole paragraphs to slide

on the screen. *The importance of this feature in the displaying of text cannot be overemphasized.* As more and more people have experience with displays of different kinds, they are beginning to realize how confusing and disorienting it is for a screen to clear and be filled with something new to read. *You don't know where you are.* On subroutining displays, moving the text can give the reader the same sense of orientation he gets from turning pages—an important thing to replace.



It must be stressed here that, just as computers themselves have no fixed mode or style of operation, neither do computer displays; and so the purpose of such devices is simply

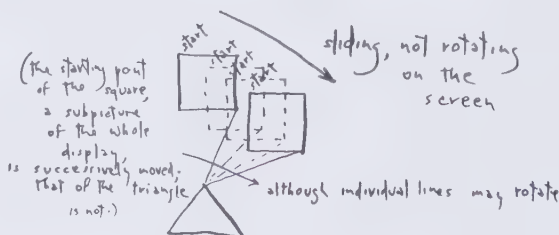
HELPING PEOPLE SEE AND MANIPULATE PICTURES AND TEXT IN ANY STYLE, AND FOR ANY PURPOSE.

Since pictures can be *of* anything, and text can be *about* anything, this effectively comprehends the entire mental and working life of mankind.

Many readers will scoff, supposing that computer display systems will *always* cost tons of money. This is not the case. You can already get a beauty, with its minicomputer, for as little as \$13,000; and this price should fall to three or four thousand within a few years—as soon as the minicomputer manufacturers realize that the market frontier is not in the office or factory, but in the home. But we're getting a bit ahead of ourselves here.

## TYPES OF SUBROUTINING DISPLAY

Some early subroutining displays used a screen-dotting technique, but took the burden of it off the computer itself: It would extract from core memory the instructions telling it to draw individual lines and show text. (I refer here to the DEC model 338, introduced about 1965; this attached to a PDP-8 computer (see p. 85) and cost about \$50,000, including the computer.) Others drew lines as straight zips of light across the screen; an



example is the IBM 2250 display, introduced about 1966. (The model 1 of this device buckled directly to the 360, and cost, I believe, something like \$75,000; its successor, the model 4, buckled to their 1130 minicomputer, the package costing some \$150,000 and *then* you were supposed to attach it to an IBM 360.) The 2250 was a good machine, but in performance suffered greatly from the restrictions of the 360 computer itself (see p. 88).

These earlier machines are being replaced by new versions with better-designed instructions (see "Computer Architecture," p. 76, for a sense of what well-designed instructions are). An especially fine unit is DEC's GT40, which buckles on the exceptionally fine PDP-11 minicomputer (see p. 90). The GT40 is illustrated nearby (p. 15). It goes for some \$12,000, including the computer. (That's *today*, we repeat. Consider not the price at this instant, but how fast it's going down.)

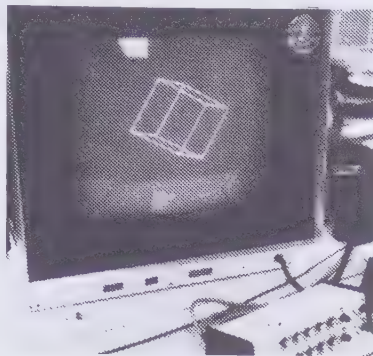
The units mentioned above are of the most basic type: "two-dimensional," whose pictures at any given instant correspond to flat drawings—but, of course, derive their excitement and magnificence from their capacity to interact, change, and animate what you are looking at.

### THREE-DIMENSIONAL LINE DISPLAYS

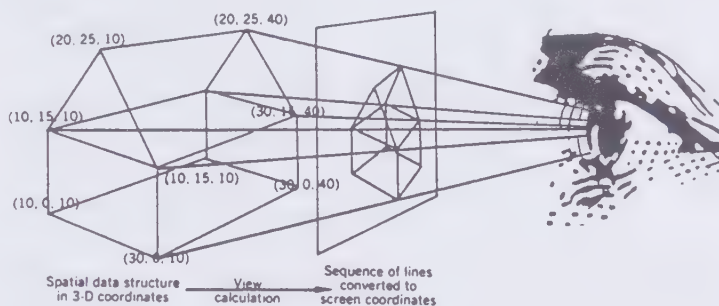
So far we've discussed the two-dimensional subroutining displays. However, things do not by any means stop there. A number of people in the early days experimented with techniques for drawing line pictures by program; the earliest of these used *plotters*, output devices that let the program draw with a pen. But interest soon grew in the possibility of interactive three-dimensional displays on screens. Johnson's SKETCHPAD 4 did this entirely by program. But as night follows day, people set about putting these techniques into *hardware*, creating devices that would automatically show things in three-dimensional views—*allowing the viewer to rotate views of nonexistent objects as if they were on unseen turntables*.

The views we are talking about, now, consist of bright lines on a dark field, and so the "objects" we are talking about are called "wire-frame" objects—they could effectively be made of welded wire.

But now we do not have to build them physically to see them.



Basically a three-dimensional system of this type stores the lines as coordinates in threes: endpoints of lines in a mythical three-dimensional space. Each point's location in the space is told by three numbers; a line in a space is represented in the data structure by two such points, and a code or something tying them together.



The computer, as a penman, draws lines from a list stored in core memory. In a three-dimensional system, the basic list of 3-D coordinates is converted to a list representing a particular view; the result looks like a wire frame.

The second program follower in such a device behaves much as it does in the 2D system, but with certain additions. Like the 2D system, it proceeds down its own program one step at a time. Like the 2D system, it finds in its program the coordinates of a line to display and creates electronic signals representing its endpoints. But it does not display these directly, since these are three-dimensional coordinates. Instead it routes these signals to what we may call a *view calculator*, a particular piece of hardware that has been primed with the angle from

which you want to view the object. This view calculator, automatically and by mysterious means which vary among machines, produces the view, and *its* signals go to the screen.

Let's say we want to display a point. The display's program follower pulls three numbers from its display list and notes the code that says it's a spatial point and not the end of a line. These three numbers slide on into the view calculator, already primed with the angle of rotation; and the view calculator figures *where on the screen that point should be displayed*. The coordinates for the screen—telling where the point goes in the desired picture—go to the screen controller, and the point is brightened.

How are these coordinates calculated? Well, some commercial units do it electronically ("in analog") and some do it symbolically ("in digital"). The result is the same.

(If you want the equations for this, they're in the Newman and Sproull book.)

Then how does the view calculator handle a line? Same thing.

The program follower pulls three numbers from its display list and notes the code that says it's a line, so it takes three more. Then the view coordinates of *both* points are calculated and fed to the screen controller. The screen controller now has two points on its screen—so it draws a line between them.

The first device of this type was, I think, the so-called Kludge (pron.

"Klooj")—computer slang for a ridiculous machine, but in this case applied affectionately) built at MIT's Electronic Systems Laboratory in the early sixties. This device was a one-of-a-kind, built out of DEC circuit cards and hooking to a bigger machine. The ESL Kludge showed vividly how good it was to have instantaneous view calculation under a user's control.

The first of these systems to be offered commercially, I believe, was the "Adage Display," made by Adage, Inc. of Boston, which used their unusual Ambilog computer (see p. 92) to rotate objects on the screen. I vaguely recall that it cost about \$80,000 with computer but without accessories.

Actually, Adage had a tremendous lead in this field, but they let it slip for some reason, and have now lost it to two firms: Evans and Sutherland on the high end, Vector General on the low end. (But of course things keep changing.)

The Evans and Sutherland Computer Company was founded in 1966 by Ivan Sutherland, creator of the masterful Sketchpad system, and David Evans, chairman of computer science at the University of Utah. (For a time both held appointments at U<sup>2</sup> at the same time, but now both have left the university to devote full time to their dream factory in Salt Lake City.)



Sutherland (left) and Evans (right). Photo courtesy University of Utah

EVERYTHING IS DEEPLY INTERTWINGLED.



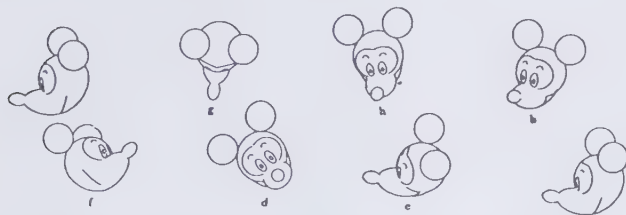
The rules of perspective have been understood since the Renaissance. In olden computer times (up till about 1965), people used to do three-dimensional view calculation by angles relative to a three-dimensional data structure. Then Larry Roberts at MIT noted that there was a more appropriate mathematical method, long moldering in obscure texts. The idea is this: if you *add an extra dimension to the data*, it's easier to program. It's easier because it becomes a simple matrix multiplication, which has no commonsense explanation but is important to mathematicians.

SO, that means that to calculate views of three-dimensional objects, the most usual way is now to add that extra dimension. Instead of having a point in space whose position is 36-24-36 (in some set of three-dimensional coordinates), another arbitrary number is added to make it, say, 36-24-36-1.

It seems that in the mathematics of multiple dimensions, it comes out simpler that way. Indeed, from a mathematical point of view, the new, improved dimension is *just like the other three*. For this reason, such an augmented system of coordinates is called *homogeneous coordinates*. Like homogenized milk, the additional coordinate is just stirred in with the rest, and out comes your desired view calculation. (The formulas are to be found in Newman and Sproull, *Principles of Interactive Computer Graphics*, McGraw, \$15, your basic text on the subject.)

At any rate, the additional coordinate is often referred to, incorrectly, as the "homogeneous coordinate." They're *all* homogeneous, which is why it works.

**UNFORTUNATELY, just to get through the basics, there is only room to discuss stick-figure graphic display here. But curved surfaces may also be depicted, though usually not interactively. See below, and pp. 107-119.**

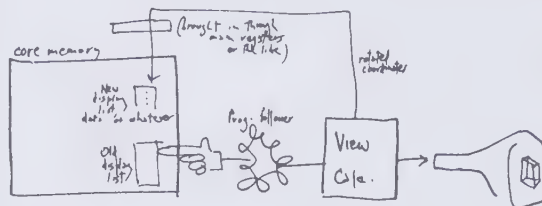


Drawings by Ruth Weiss' BE VISION program, done at Bell Laboratories, mid-sixties.

This program represented truly curved surfaces in its data structure, as "quadric surfaces"—that is, involving powers of two in the math, and calculated the visible lines tangent to the edges from the viewpoint, thus drawing the edges. Removing the hidden parts of the curves is, of course, one of the greatest problems. (From Ruth A. Weiss, "BE VISION." JACM Apr. '66, 194-204, p. 201.)

Their first product was an extraordinary piece of hardware called the LDS-1, which they said innocently stood for Line Drawing System. (To anybody from Utah, however, LDS means Latter-Day-Saint, and don't you forget it. Evans, indeed, is a Mormon, but I've been told it may have been Sutherland's sense of humor that chose the acronym.)

It should be pointed out that a special advantage of digital perspective calculation is that viewed coordinates can be read *back* by the computer, and serve as new data, if you go for that sort of thing.



The Adage Display is *isometric*, meaning that lines do not get shorter as they get farther away or longer as they get closer. While this is marvelously impressive, most people want real perspective; and it was this that Evans and Sutherland set about to make available in real time, i.e., in direct response to the viewer's actions.

The LDS-1, weighing in at half a million dollars or so, buckled to the PDP-10, a big 36-bit computer from DEC (see p. 87). Its view calculator worked symbolically (digitally), and thus could work to the higher precision necessary for true perspective calculation.

Among the exciting demonstrations that you can see sitting at an LDS-1 are a map of the United States you can zoom in on, bringing you in to a map of New Jersey, then Atlantic City, then a specific

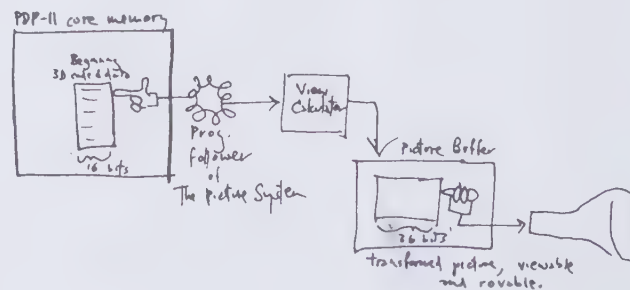
straight at the carrier. In all cases, the ghostly ship will move, turn, and change perspective on the screen as if somehow it were really there.

Meanwhile, a little new firm of young guys in Southern California, Vector General, came up with a line of terminals like the Adage line, except that they could buckle to the 16-bit minicomputer of your choice. (In practice, most of them have been attached to PDP-11s; see p. 90.)

The Vector General display is *isometric*, and makes its calculations in analog, like the Adage Display. It has been very successful among both universities and private corporations. In addition, a highly interactive and well-designed language is available for the creation of data structures representing 3D objects, as well as for general-purpose programming and the creation of whole environments. And it's free to individuals or companies that have Vector General displays attached to PDP-11s. (See "Coup de GRASS," p. DM 96.)

But wait. Evans and Sutherland has now dropped the LDS-1 and given us—no, not LDS-2, but something called The Picture System—*also* built onto the PDP-11, but this one works symbolically (digitally) and in full perspective. The price starts at eighty grand.

Since The Picture System works out of the PDP-11 core memory, the commands it follows are 16 bits long, since that's the size of a slot in PDP-11 core. But wait. They've designed the thing to *convert to 36 bits*, so that coordinates are moved to a private store or buffer between the program follower and the display. This means the display can zoom and zip around in the scene without bothering the computer.



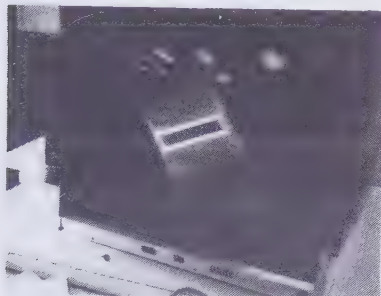
intersection, all in one smooth continuous motion. Also a simulated landing on the flight deck of an aircraft carrier—with you flying the airplane, so you can go over it, to the side, into the drink or

Another important feature of The Picture System: It will do, not just ordinary perspective, but such weird view calculations as wide angle barrel distortion, pin-cushion distortion and similar stuff.

## INTERACTIVE ROTATION

3D screens—aside from their fun and excitement—allow people to understand and work with complex 3D structures without having to build them physically.

The understanding, however, comes from being able to *turn* and *manipulate* the structure on the screen. If you can't turn it, you can't really perceive the 3D structure, because the arrangement of lines could be anything.



However, systems like the Adage and the Vector General and the Evans and Sutherland devices allow you to turn things on the screen as easily as if they were on turntables behind a pane of glass. *That's* how you see, you see.

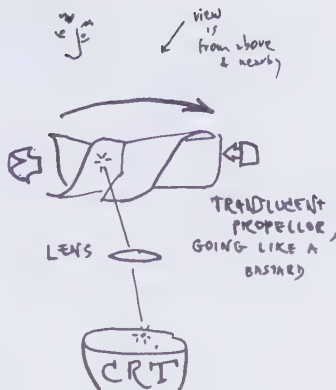
This interaction is what makes computer display augur a new era for mankind, if we're lucky. (It's also why we use the term computer *display* in this book, rather than "computer graphics," since people who make computers *draw with pens* are also doing "computer graphics"—a related activity, but not one to change the world.)

## WHAT ABOUT REAL THREE-DIMENSIONAL DISPLAY?

In science-fiction stories, you hear about how objects are made to appear as if they're standing in the middle of the room. For instance, I believe that in Heinlein's *Stranger in a Strange Land*, they watched a "tank" in which things appeared.

Well, a lot of people have thought about this, and it's not so easy as you might think.

One interesting scheme used a sort of translucent propellor, spinning rather fast, on which computer-generated images were projected from below. It was done by the dotting method, so that a bright dot of light would appear high or low in space depending on whether it was projected on a relatively high or low point on the propellor.



This was interesting but had numerous disadvantages—not the least of which was the danger of the thing flying apart. (Translucent materials tend not to be as strong as, say, metal.) Another basic problem, though, was the fact that any given point in space could only be displayed at a *given time*, when the propellor's height in that region was just right, and that meant that at that given instant you couldn't display any of the other points that could only be displayed at that instant. A considerable disadvantage.

Probably the most astonishing 3D display is Sutherland's Incredible Helmet. This consists of a helmet with two dinky CRTs mounted on it, *each* being driven in real time by a perspective system (such as the LDS-1) and set up with prisms to the wearer's eyes (see drawing p. DM 24). Through the prisms the wearer can see the real world in front of him. *Reflected* in the prisms, however, and thus mixed into the view of the real world, is the glowing wire-frame being presented to him—in perspective, and with its *separate views merging into an apparent object in front of him*. But he need not stand still: As he moves, the helmet's changing position is monitored by the program, and the display system changes the views accordingly, meaning he can *walk around and through* a displayed object. The illusion, and the possibilities, are fantastic: imaginary architecture, explanations and diagrams of things in the room, poetry that changes as you walk through it, . . . well, you work on it. Not available commercially.

Lou Katz, of NYU, put old-fashioned stereopticons up to the CRT, and displayed two separate views to the two eyes. Works fine, even with isometric display.

Stereo glasses that wink open alternately, left and right, are now sold for the Atari ST as a standard accessory (\$150),

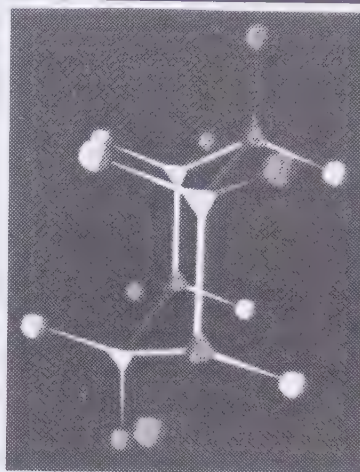
and several Atari programs already work with them.

Bob Spinrad of Xerox Data Systems has a patent on displaying 3D from a computer through an ordinary color TV. Assuming you're using some standard way of refreshing the TV—described elsewhere—the image for one eye is displayed in *green*, the other in *red*, and you look through *red and green glasses*. The wonders of modern science. Spinrad chuckles over it himself.

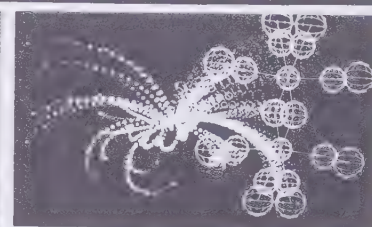
Another scheme glued silver mylar to the front of a loudspeaker, then played a soft hum through the loudspeaker to pulse the mylar back and forth. Then you used that as a mirror to look at what was going on the CRT—which was showing

a lot of points at odd places that would appear to be in space. Unfortunately, this was hard to coordinate and like the propellor, often required you to put dots in several places at once, which don't work.

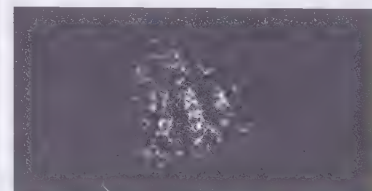
For a while you could get—maybe you still can—a three-dimensional computer-output device. Here's what it did: It created objects showing data structures that had three variables. (It didn't make wire-frame objects or the like.) Automatically ejecting wire through a styrofoam block, and snipping the done ones, it created little mountains showing three-dimensional data. Very cute. Since many people have problems with mountainous computer data, it probably should have caught on. ☹



Bouknight & Kelly (see p. DM 106)



Computer graphics has created a new frontier for chemistry.



Shows part of hemoglobin molecule. Data structure from Richard J. Feldman. NIH.

## Three Ways Of Seeing Molecules Using 3-D Computer Display.

Much of today's impetus for 3D computer display is coming from the field of chemistry. University chemistry departments are buying equipment like the Evans & Sutherland LDS-1, the Adage and the Vector General.

Why?

Because chemistry is increasingly involved with complex three-dimensional structures. Crystals, long folding chain molecules, miniscule forces acting on structures whose shape determines the outcome. Organic molecules that involve thousands of atoms, and whose complex folded structure exposes only certain key features. And so on.

The Vector General display illustrated here and there on these pages belongs to the Department of Chemistry, University of Illinois at Chicago Circle.

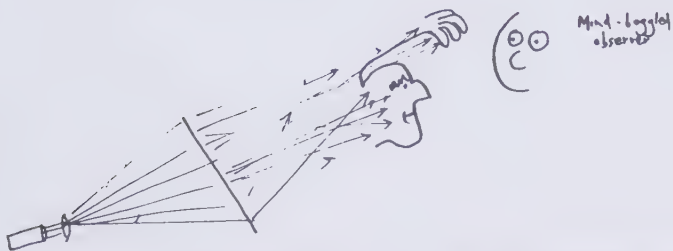
Molecule hackers are now getting more and more elaborate scopes, today's chemistry workstations—where they may study the folding, the folded bond patterns, the resonances and so on. And they are designing cleverer and cleverer molecules—new flavors, new drugs, no-calorie food fat substitutes, and even the wicked "designer drugs" that have created a new phase of the drug problem. See list of programs chemists use, p. 151.

Tom DeFanti

Tom DeFanti



## Return Of The Hologram



A HOLOGRAM REENACTS ALL THE LIGHT RAYS OF WHEN IT WAS MADE and must be shown by an identical laser in the same position. Spooky Image recreated by EXACT TANGLE OF LIGHT RAYS WHICH HIT THE FILM ORIGINALLY & IS NOW RECONSTRUCTED.

**HOLOGRAPHY** puts YOU  
in the picture, and YOU  
and You  
and You...

Holography is one of those Modern Miracles that we really can't get into. It is mind-blowing, influential, and of unclear importance.

Theoretically predicted by Dennis Gabor, the *hologram* (Greek "whole picture") was finally made to work in the late fifties by Leith and Upatnieks. Since then dozens of other types of holograms have been experimented with, including color holograms, movie holograms, video holograms, audio holograms and gracious knows what.

Basically a hologram is an all-around picture. It doesn't look like a picture, but looks like a smudged fingerprint or other mistake of some kind.

Yet it is a marvel.

A basic hologram (—actually it should be called a *laser* hologram or *Leith-Upatnieks* hologram, but we've no time for such distinctions—) is one of these smudgy pictures which, when viewed under a proper laser setup, shows you a three-dimensional picture. Worse than that: as you move your head, the picture changes correspondingly. It looks, not like the flat surface it is, but like a lit-up box with a model in it.

What does the hologram do? Actually it recreates, not a single view, but the entire tangle of light rays that are reflected from the real object. Even down to bright reflections, which scintillate in the usual way, as from chromium.

The only problem: ordinarily they have to be used with laser light, which is spookily one-colored.

Notes from all over: art stylist Salvador Dali presided at an unveiling of "the world's first 360 degree hologram" at a New York gallery not long ago. The subject was a song by Alice Cooper.

The Haunted House at Disney World in Florida will ride you through a building full of holograms. That's one way to move through ghosts, all right.

There is a New York School of Holography.

Then a lot of people mumble the word "holography," as if that is going to settle something. While holograms are terrific and remarkable, and have been produced on computers, making them is not a process that can be carried out decently on sequential machines—let alone making them in real time. So if a solution to interactive three-dimensional computer display is going to come through holography, it means a whole new batch of technology will have to be invented.

My friend Andrew J. Singer, who comes and goes in the computer field and is one of the five or six smartest people I ever met, says *he* knows how to build a display tank, and I believe him. He explained it quickly to me once and I asked him to tell it again, but he just said sadly, "What's the use—there are so many great things that could be done..."

### FOUR DIMENSIONS, EGAD

So much for three dimensions. Now, some readers are bound to ask, "What about *four* dimensions?" because they are science-fiction fans or troublemakers or mathematicians or something.

Just as we can make a two-dimensional picture of a three-dimensional object, it is possible, dear reader, to make a two-dimensional picture of a *four*-dimensional object.

What is a four-dimensional object?

Why, any object that has four dimensions (thanks a lot, you say), or even four *measurable qualities*, such as height, weight, age, and grade point average (see hypercube, p. 93). Well, let's not get into that, but it turns out that views of such multidimensional structures may be obtained by the same homogeneous matrix techniques already mentioned for regular perspective calculations. Rule of thumb: However many dimensions your data has originally, you add one more dimension, homogeneous with the rest, and there exist formulas (sorry, I don't have them) for view calculation.

(Note, of course, that while a two-dimensional view is a *picture*, a three-dimensional view is a three-dimensional object—you'll have to view it on an interactive 3D computer display of some kind.)

The above article stressed line-drawing and subrouting displays. Because of the prices of chips, this turned out not to be the form of computer display that would take over in the mass market. Instead, it was—

## BIT MAPS

The main development in computer graphics in the last year has been the sudden upsurge of the bit-map approach to computer display. While the approach, and equipment for it—like the Data Disk system—have been around for some time, the falling price of electronics, especially in the memory area, have made it abruptly the cheapest and thus the most popular type of computer display for graphics.

A "bit map" is a series of dot positions, or bits, recorded in some form of fast memory and read out in sync to a conventional scanned video system (see pp. DM 84-86). The one bits stand for dots or little squares, the zeroes for nothing, and the video system brightens the corresponding zones on the screen. This method has certain disadvantages—parts of pictures cannot be automatically

distinguished or separately animated, as with subrouting display (see "The Mind's Eye," esp. p. DM 88)—but for the money it's great. Sizes given refer to the number of squares in the rectangle of the picture.

The most spectacular demonstration of bit-map color so far has no doubt been the film done by Dick Shoup et al at Xerox PARC (see p. 128), showing the super animation that's possible when big-computer resources are given over to bit-map animation. Their system is 600x800.

### THE PRICES IN '75

#### BLACK-AND-WHITE

An off-the-shelf bit-map system for the PDP-11 or the Nova is available from Intermedia Systems, Cupertino, CA (\$2750 or \$2500 respectively). May be ganged for grey-scale or color. It's 256x256.



For the Altair, the forthcoming 8096 display will have 120x120 or 240x240 bit-map graphics, for prices starting around \$1000.

*Don't it make you want to cry for joy? You get four times as good as that on the PC now, with the computer thrown in at the same price.*

## COLOR

Extra bit maps, plus electronics, can get you color; if you double the number of bits you can double the number of available colors on your display, ad infinitum.

On the small side, 64x64 color will shortly be available for the Altair from the Digital Group, Denver. A 128x128 color bit-map system for the 11 has just been announced by DEC (for "nuclear medicine" of all things—but they will part with it to anybody for 8 or 10 thousand (not yet fixed)). They stress that this will be the first of a modular series of bit-map displays, with plugins for different degrees of resolution and different character generators.

Ramtek and Comtal both make 256x256 bit-map systems, priced in the \$16,000 area.

Above this resolution, special TV systems tend to be necessary. Both Ramtek and Comtal make very expensive systems for the purpose, using solid-state and disk respectively.

## WHAT YOU CAN GET IN '87

*Gorgeous color boards are now available for the PC, permitting bit-map graphics up to at least 1024x1024. These things are only a few thousand dollars now.*

## FRAME BUFFERS (also called "Graphics Boards") BIT MAPS GET SERIOUS

*A frame buffer is a section of memory that is continually read out onto a display screen, but into which the computer can put changing data to change the picture. Frame buffers come in all sorts, up to what you want for movie-making (whence the name "frame buffer").*

**Black-and-white.** In a black-and-white frame buffer (such as the Macintosh screen) every pixel is just a bit. Cheap and simple.

**Color, no palette.** In this type of buffer, certain fixed colors are allowed and each pixel is several bits, which select the color. Example: the IBM PC standard display, or CGA board (Color Graphics Adapter).

**Color, with palette.** Using a little more trickery, the same type of arrangement can supply changeable colors. Each pixel is still the same several bits, but what color those bits stand for may be changed. Palette registers can be set for the bits "010" to mean fuchsia, orange, chartreuse, or whatever. This method can give you a lot of variety, but not much range: that is, you still get only as many colors at one time as the bits allow. Three bits per pixel, eight colors at once; four bits per pixel, sixteen colors at once; and so on. (Frame buffers for 3D image synthesis are discussed on p. DM 113.)

As a first taste of interaction on a graphical computer system, PLATO can be a thrilling mind-opener—especially to people who think computers can only behave loutishly or through printout.

## THE NEW REPUBLIC OF



Above: PLATO LIVES! Anyway, the word itself goes through changes in the Game of Life (see p.113), as programmed for the PLATO system by Danny Sleator, and photographed from a PLATO screen.

PLATO is the world's greatest computer display system.\*

Some 500 users, at terminals around the world (but mostly in Illinois), simultaneously tie up to a big computer in Urbana, Illinois, and savor instantaneous pictorial and text deliveries on their bright orange screens. Diagrams, explanations, tests and even animation of a sort, flow almost without interruption to the bright orange screens all over. The system is extremely responsive: Depending on what the user is up to, its various programs can respond to each pressing of a key, usually within a fraction of a second.

While literature on PLATO is copious, it is hard to read and slightly sales-oriented. But a few minutes' intercourse with a PLATO terminal makes anyone an enthusiast for the system.

PLATO is the brainchild of Don Bitzer, a U. of Illinois engineer who has

\*In terms of high performance for lots of users. Various systems (described hereabouts) offer more power, but at huge costs.

devoted over a decade to its creation. Michael Scriven, no slouch himself, has called Bitzer "one of the great men of our time." Bitzer is also certainly one of the world's greatest salesmen. A crew-cut, huggy-bear sort of a fellow, he flies around the world demonstrating, lugging a great terminal along. When you sign on the system, you may be informed that Bitzer is at that very moment demonstrating in Paris or Tokyo. This "traveling dog and pony show," as PLATO staffers call it, has created awe and excitement wherever it goes, and where the awe has been strong enough to generate money, there you will find PLATO terminals.

If you have a PLATO terminal—"you" presumably being a school or other favored institution—you can in principle log onto PLATO from anywhere in the world, though most terminals stay in one place. There is one main network, consisting of a big Control Data computer in Urbana (the model 6800; see p. 89) with tendrils extending out into the phone system and the educational TV cable of the state of Illinois. When the Urbana system is "finished" and fully loaded, it will have 1008 terminals; *all are already spoken for*. The PLATO terminal is a totally unique animal (see nearby), manufactured (all too slowly) by Magnavox, incorporating a terrific plasma panel built by Owens-Illinois. (The plasma panel was invented by Bitzer, and even though much of PLATO was publicly funded, he is reputedly rich from it. We said he was a great politician.)

PLATO is a complete stand-alone system, with its own monitor program, or "operating system" (see p. 99) running on the CDC 6800 computer all by itself. It does not run on any other manufacturer's computers, or simultaneously with any other big programs. It communicates only with PLATO terminals, no other, and PLATO terminals, because of their unusual design, can communicate only with it, and partly because of its unique 20-bit interface. (See diagram of PLATO terminal, p. DM 14.)

A PLATO terminal costs about \$4000 and the price seems to be going up; \$5000 in the next few years is a popular estimate. But you can't just buy one. You have to get on the waiting list, and who are you, anyway? There was a time when almost anybody could buy into PLATO, but now that the system is unstoppable, applicants are being scrutinized. ■



## Is It Better To Toot?

"A tutor who tooted the  
flute  
Tried to tutor two tutors to  
toot.  
But he asked through his  
snoot: is it  
better to toot  
Or to tutor two tutors to  
toot?"

—Folk thing

Paul Tenczar's TUTOR language, the programming language inside PLATO, is like any other programming language (see p. 56): intricate, and unlike its results. That is, a program bears no more resemblance to what it *does* than the word "cow" looks like a cow.

PLATO is a system for canned presentations that respond to the student. Students need not know TUTOR. Anyone out to prepare such presentations must learn it, however; and the attempt has discouraged many.

Tenczar is a former biologist, and had no preconceptions from computer orthodoxy to bind him in the design of TUTOR. Thus the language is very original. There is only room to raise the following points:

To learn the first steps in TUTOR—how to set up drill-and-practice lessons, for instance—is unusually easy.

To do anything complex, however, requires you to learn the bulk of the TUTOR language. Thus when people say TUTOR is "easy," they mean those first steps.

TUTOR is not Extensible. That is, a programmer cannot customize the language with new compound functions of his own making. Steps are being taken to correct this; meanwhile, it is said that the Urbana people can be persuaded to put in new commands others want for, e.g., chocolate chip cookies.

The TUTOR language grew out of drill-and-practice, for which it has a command specifying where a student's answer is to appear on the screen. This is the "arrow" command. The language has a strange scanning structure built around this "arrow" command. Beginners don't need to understand the scan and the arrow command, but journeymen do.

Is it really unstoppable? Educational Testing Service, of Princeton, is conducting an elaborate Effectiveness Evaluation of the PLATO system, presumably to decide whether it should live or die (on public funds). But with so many terminals in the field already and so many man-years already gone into its creation and the making of materials for it (—the ghastly term "authoring" seems likely to stick), it is hard to believe PLATO could die. Now, now.

Especially considering that two more systems are now being put together: at Lowry Air Force Base (Colorado) and Florida State University. That means there will be whole other computers of the CDC 6000 series running the PLATO terminals, unconnected to Urbana, one for Lowry AFB and one in Florida.

And it won't end there.

Control Data, whose vested interest in the system (though they didn't pay for its creation) is enormous, is said to be projecting ONE MILLION PLATO TERMINALS BY 1980.

Now, to call the PLATO system a "computer graphics" system may seem somewhat odd to people who know it in another guise, as a system for the Computer-Assisted Instruction (called CAI). But as the author does not like CAI in general, at least as it's been going—see pp. DM 129-136—and rather likes PLATO, I prefer to describe it as I prefer to see it.

Nevertheless, to understand PLATO properly, we had better consider what the people have been doing in terms of what

they think they have been doing, and offer any amendments or restatements later.

### "OPTIMIZED FOR CAI"

PLATO stand for "Programmed Logic for Automated Teaching Operations," and has been billed (and sold) as a system for automated instruction.

It is, most PLATO fanciers will tell you, "optimized for instruction." ("Optimized," in computer talk, means "just what somebody says you need for a specific purpose.") As with any system, the leaps of faith between its basic design premises have become lit by airport beacons; clearminded individuals with alternate views have difficulty making themselves understood to some PLATO enthusiasts. But the most basic underlying feature of the system, INSTANT RESPONSE, cannot be quarreled with. PLATO can respond, as already mentioned, to a single key-pressing by a user, almost instantly; this feature is virtually impossible, say, on IBM systems. This responsiveness is the system's greatest beauty.

Because of the need for high responsiveness, it was decided that all users had to have their particular programs ("lessons") running in core *at the same time*. That meant there would be no swapping (bringing in materials from disk memory), which can bring mortifying delays (if a lot of people need it at once); but it also meant *lessons have to be very small*. Large bodies of material, which would have to be moved in from disk, are not allowed; thus each lesson is

### CONCEPT OF A CONCEPT

Much has been made of TUTOR's facility for "analyzing the content" of what students type in. Actually, of course, the computer does not "understand" what the student says (see "Artificial Intelligence," p. DM 120), but rather offers certain efficient tricks to the person using TUTOR to prepare presentational materials.

Basically, TUTOR's "concept" facility *reduces every input word to a 60-bit code*. The technique of reduction (called a "hashing function") supposedly substitutes for any word of any language a code of 60 bits (see "Binary Patterns," p. 65), which means the program in TUTOR can rapidly test a student's input for numerous different possible things.

Thus a TUTOR program may contain "concept searches" that test whether a student types either a desired response or numerous alternatives.



Amongst the terminals, PLATO room, Circle Campus. What one person is doing ordinarily has no bearing on the others, who could as well be in Timbuktu as far as the main computer is concerned.



basically a little love-nest that must generate its own action. Hence there is an emphasis on little *programs* to respond various ways, rather than text which may be read in quantity.

Partly because large amounts of text cannot be shipped to the user, a little PROJECTOR is in the terminal. It uses a tiny microfiche, or microfilm sheet, small enough to fit in the palm of your hand.

If a PLATO author deems it necessary, he requires for his lesson, not just the use of the keyboard and plasma screen, but a microfiche as well. The student must put the microfiche in place when he starts the lesson; signals from Urbana (or wherever) then jump the projected image among 256 different images, in response to what the student does.

Now, PLATO people are not doctrinaire about how their system is to be used. The plasma screen can be continuously showing little decorations along with the teaching material. The microfiche could be showing irrelevant works of art or travel scenes. These are all facilities at the option of the PLATO author; at his beck and call, if he thinks his program or lesson needs them. (But it's very bothersome to have the microfiche made—an important difficulty.)

Every terminal has the screen, the keyboard, and the projector. Other options may be added, however:

1. *The touch panel.* This is a transparent window that goes over the plasma screen and reports to the main computer whether it has been touched, and where. (This allows illiterates, especially kiddies, to use the system without typing.)

2. *The audio disk.* This allows the terminal to respond with sound, including canned words, to the student. (It does not actually synthesize the sound, as discussed on pp. DM 60-63.)

3. *The general jack.* Not to be confused with Pershing, this is a connector socket that will send and receive data from *any other* device—provided you've got the right interface. This allows all kinds of other devices, such as piano keyboards, to be used for student input. Or output (like gum-ball machines).

Actually, except for the restriction on quantities of material that can reach the student, PLATO is an extremely general system. Despite the strange convention of calling all user programs "lessons"; despite the odd stipulation that all users are

called either "students" or "authors"; and despite being told by PLATO spokesmen that PLATO is not a general-purpose system; actually, it is.

The hardware was designed by Bitzer. The software, that is, *the underlying computer part*, never mind the contents to be shown (also regrettably called "software" by many hands) was initially less stressed by Bitzer, but eventually grew under the direction of others. In particular, an ex-biologist named Paul Tenczar (pron. "Tenzar") created its underlying TUTOR language. (For an introduction to computer languages see p. 56 and what comes after.) The TUTOR language exists only on PLATO; and PLATO authors may only use the TUTOR language, Paul Tenczar's creation.

Tenczar's TUTOR has both the virtues and defects of being original. Apparently unlettered in computermen's controversies and dogma, Tenczar designed a language of great power and speed; is *utterly strange to computer people*, offers various brilliant features, and is in some respects quite irritating. It looks very simple to the user, but beyond a few deceptively simple techniques, it has to be learned in considerable detail to do anything interesting. (See "Is It Better To Toot?" nearby.)

This tale has, of course, been simplified. Bitzer and Tenczar did not work alone, but rather were leaders in a seething community of dozens of smart people working like blazes on the project. It has taken some fifteen years of Bitzer's effort, and tens of millions of dollars, to get the system where it is now—Ready and Working.



Mike O'Brien, a Tolkien fancier, has put the entire Elvish alphabet onto PLATO as a special character-set. Here the system gives a famous warning to turn back, both in English and Elvish. Mike says it intimidates snoopers poking around his material.

Project PLATO now extends far beyond its original domain. Originally a fairly tight nucleus at the Computer-Based Education Research Laboratory ("CERL") at the U. of Illinois in Urbana, the community of PLATO now sprawls out through its lines to a larger constituency, the PLATO community of users.

(Indeed, this extended Republic of PLATO—the systems people (see p. 161) in Urbana, the authors and locals-in-charge throughout the network—constitute one of the maddest rookeries of computer freaks in the world. Where else would you find a 14-year-old systems programmer who's had his job for two years? Where else would you see people fall in love over the Talkamatic (a PLATO program which allows you to have written conversations with people at other terminals, wherever they may be) only to clash when at last they meet in person? Where else can you play so many different games with faraway strangers? Where else can students anywhere in the network sign into hundreds of different lessons in different subjects (most of them incomplete)? Where else are people working on various *different* programs for elementary statistics, all to be offered on the same system?)

PLATO is one of the wonders of the world.

Unfortunately, there are so many learners, and so few PLATO terminals, that use of the terminals must now be fairly strictly controlled. (The eight terminals at the University of Illinois at Chicago Circle, at which most of these pictures were taken, generally work an eight-hour day.) The time was when people could just walk in, sit down at a terminal and do what they liked; now, sadly, each user must have an "account" and a password.

But the rabble is howling at the gates. Many professors want to use it to take rote aspects of teaching off their backs; and the computer bums and students want to play the PLATO games (see p. DM 20) and tinker with an interactive system of its power and lusciousness. But most of them will have to wait.

PLATO's services are "free," for now. That is, if your school has PLATO terminals, and IF it will pay for the communication lines, THEN the services of the central computer are "free"—the National Science Foundation is bankroll-

ing its operation for a couple of years more. Then, bango, PLATO central service becomes something that has to be paid for, too.

Just to give you an idea, the communication costs to Urbana for Circle Campus's eight terminals run at *over \$10,000 a year*. But these costs should be coming down sharply; it is the price of tooling up for whatever the PLATO future is going to be. Anyway, the general cost of the system comes out to about \$1.50 an hour, the same as general time-sharing on a PDP-10 (see p. 87). But that's without paying for the central computer—another cost which we expect to go down, however.

This is all a far cry, of course, from Bitzer's claim a decade ago that PLATO terminals would cost only \$400. But considering the system's success, we needn't dwell on that.

Perhaps the real question is this: with man-machine intercourse of this quality now possible, can people's love for the system stay Platonic?

## PLATO'S RETREAT

*The mighty PLATO system is not dead, but neither is it well. Control Data, which stood to benefit a lot if the system had succeeded big, has more or less taken over, and is trying to salvage whatever it can.*

*There were not one million terminals by 1980. There were only a few thousand.*

*Part of it was the screen. Plasma screens stayed expensive and unreliable, while video bit-map screens got cheaper and cheaper, and so did computers. There was no point in buying a PLATO terminal for \$5000 when you could get a PC for \$2000.*

*So they've moved the TUTOR language (already misdesigned for big systems) onto PCs—little machines with CRTs—and the whole original idea is gone.*

*The courseware—greatly varying in quality—is said to be available somewhere. But given the economics, they're looking for customers with many many students who need the same stuff over and over.*

*In the annals of disasters, boondoggles and misjudgements, PLATO (probable cost in the hundreds of millions) rates about medium. And it was guided by idealism in large part. But it grew out of a strange and rather silly notion of what education needed, or could assimilate.*



# SUTHERLAND'S SKETCHPAD

Seldom has an event in a new field had as much power and influence as what our Ivan Sutherland did as a young man in the period 1960-64.

The SKETCHPAD system, which was basically his thesis work at MIT, was at once inventive, profound, overwhelmingly impressive to laymen, and deeply elegant. Simply for the universal influence it has had in the computer field, it deserves our close attention.

Sutherland was one of the first people to understand the use of the computer in helping visualize things that weren't fully clear yet—the opposite, of course, of the conventional notion of computers. While computers had been made to do animations as early as the forties, and computer graphics had been put to workaday duties in the old SAGE system (defending us against bombers in the fifties—remember the good old days?), Sutherland turned computer display from an expensive curiosity into a true dream machine.

SKETCHPAD ran on the 36-bit TX-2, a one-of-a-kind experimental machine at Lincoln Laboratories (a military research place nominally a part of MIT). It had a display screen, light pen and lots of handy switches.

SKETCHPAD was basically a drawing system. But rather than simulating paper (as *some* people might have done), it

found splendid ways to take advantage of the computer's special capabilities.

In the SKETCHPAD system, Sutherland found for ways that a responding computer display screen could help people design things. He pioneered methods of drawing on screens, with such techniques as the "rubber-band line" (a straight line on the screen, one end of which follows your light pen while the other remains fixed), and the "instance" (a subpicture stored in core memory which could appear numerous times and ways in a larger picture).

The mind-blowing thing about SKETCHPAD was the way you could move and manipulate the picture on the screen, with all its parts. One overall picture could be constructed out of a hundred copies of a basic picture; then a change in the basic picture would immediately be shown in all hundred places. Or you could expand your picture until it was effectively the size of a football field (with you looking at a tiny view in the handkerchief-sized screen). Or you could draw meshing gears on the screen, and with the light pen (and through the "constraint" facility) make one gear turn by turning the other!

*Ivan corrects me on this. Constraints worked for sliding actions, but not rotary.*

This elegant technique, the *constraint*, does not seem to have been imitated even now. A "constraint" was a restriction placed on some part of the overall stored picture complex. The user could move or manipulate various parts of the picture on the screen, but the parts that had *constraints* could only move in certain directions, or according to certain formulas, or dragging other parts along, etc., as specified.

This was a profound idea, because it meant that any rules for the manipulation of particular objects on the screen could be added to SKETCHPAD as *particulars* within the larger program, rather than having to be programmed in from scratch.

(One extremely interesting aspect of Sutherland's thesis, which most people seem to have missed, dealt with *displaying a structure of constraints*: that is, showing what elements depended on what other elements, in a highly abstracted diagram that the system could show you. This form of display has remarkable possibilities.

After his brilliant SKETCHPAD work, Sutherland was made head of ARPA's computer branch (see "Military," p. 157). There he was involved in many of the computer-funding decisions of the late sixties, which contributed to the impetus of computer display. (His predecessor, Licklider, had been a pioneer in time-sharing, and much of the forward movement in the computer field in recent years may just have had to do with the strategic position of those two men when they were at ARPA/IPT.)

SKETCHPAD went on as a continuing research tradition at Lincoln Labs. Timothy Johnson, for instance, made a version of it that allowed the drawing of three-dimensional objects; this became the forerunner of the various three-dimensional line systems described hereabouts.

From ARPA, Sutherland went on to the University of Utah, whence he slipped off with the Computer Science department chairman to found the Evans and Sutherland Computer Company, makers of the top-of-the-line computer display systems (see p. DM 110 and pp. DM 112-113).

Sutherland's work has shown an elegance and inventiveness outstanding in

the field. (For instance, I believe one issue of *Communications of the ACM* had two unusual articles by him: one describing an eccentric "Chinese auction" system worked out for scheduling use of a computer, which benefited users more than any previous method; and the infamous "Great Wheel of Reincarnation" article, where he compared the design of graphical computers to the Hindu system of reincarnation—if you keep adding desirable features to the design, soon you have another program follower and another computer in the same box—over and over.)

## WHY CAN'T WE GET IT?

*SKETCHPAD was so wonderful just as it was that it's a pity nobody sells it. However, it was never exactly a product, and it ran on a one-of-a-kind computer. To redo it and polish it in all its generality would be an enormous task, and evidently the market isn't interested.*

*However, a program that comes within hailing distance is Apple's MacDraw for the Macintosh. This program, elegant in itself, has various similar features (but not instancing, the ability to change a group of parts by changing the master part). I personally use it for writing; it's excellent for the two-dimensional outlining of text.*



This picture vaguely simulates the "instance" facility of SKETCHPAD, by which an overall picture may be created out of repetitions of a single master pattern.

Simulated with GRASS language (see p. DM 96).

# DEFANTI'S COUP DE GRASS

Impudent and plucky Tom DeFanti was an assistant professor at 24. This in part because he has created one of the world's hottest 3D graphics languages, which he calls GRASS. (He says it stands for GRAPHics Symbiosis System—also, he says, it Turns You On.)

(I coined the term *fantics*, for the art and technology of showing things, long before I ever heard of Tom DeFanti.)

Tom's GRASS language is an excellent beginner's computer language for two reasons: first, it is easily taught to beginners, and second, it is about *things of interest* to beginners, i.e., pictures and graphical manipulation on screens.

GRASS runs on the PDP-11, and is specifically designed for the control of three-dimensional stick-figure displays

on the Vector General display system (see p. DM 90). But a lot of people have wrestled with these matters and not done as well.

Tom believes computers are for everybody; he is not a high priest bent on making things obscure (see "Cybercrud," p. 27). Thus he made his language as sensible, clear and easy to learn as possible. Tom likes to stress the concept of "habitability" (a term of W. C. Watt), meaning the coziness of a system.

The internal form of the language is ASCII code (see p. 75). In other words, you can read programs in their final GRASS form.

For a three-dimensional system such as the Vector General, the main form of data structure is the *three-dimensional*



*object*—a list of points and lines in space. This is the form of data GRASS uses for most purposes.

In the design of such a system you want larger 3D objects to be buildable out of smaller ones. This implies arranging data in *tree structures* (see p. 72). You also want to be able to make things do compound motions on the screen—for

other side of the book, *complicatedness* is not generality or goodness or power, but a sign of the designer's shallowness.

(A note to you elegant types: GRASS is fully recursive. As a nice example, Dan

have made one mistake, which is infuriating. Consider only this: TOM DEFANTI'S 'GRASS' LANGUAGE IS PERHAPS THE ONLY SYSTEM THAT CAN BE TAUGHT IN A FEW HOURS TO COMPUTER-NAIVE BEGINNERS THAT PERMITS FULL THREE-DIMENSIONAL ANIMATED INTERACTIVE GRAPHICS WITH TREE-STRUCTURED DATA.

**NOW IT CAN BE TOLD: WOULD YOU LIKE SOME GREEBLY ON YOUR MAGUFFIN?**

In the original "Star Wars," the maguffin—Hitchcock's word for what everybody is chasing someone about—was the 3-dimensional plan for the Death

Star, bravely and loyally carried to the rebels by R2D2.

Those three-dimensional plans were secretly built by Larry Cuba at the Circle Graphics Habitat, using the GRASS language.

**WHERE ARE THEY NOW?**

Tom DeFanti has remained at Circle Campus, but done a lot on the side. He was for years a major organizer of SIG-GRAPH, and pushed them to keep their proceedings and videotapes in print.

He has also widely sold a version of GRASS, called ZGRASS, which runs on various computers. Its most recent descendant is RT1 (for Real Time).



example, showing an airplane flying around on the screen with its propeller spinning; this too implies a tree structure. There are some programmers who would use *different* tree structures both for objects grouped together and for movements grouped together; Tom uses one.

Objects shown on Tom's system can also appear to move on complicated paths through three-dimensional space. In Tom's system, such a path is merely *another object*. It seems obvious when you say it, yet this kind of simple generality is exactly what many programmers seem to avoid. (Note: this facility is a generalization of Baecker's p-curve; see p. DM 56.)

Input devices are completely arbitrary and programmable. What happens on the screen can be controlled by anything—any variable, any particular form of control, allowing user programs to make the connection between controls and consequences. Tom's language is interpretive. However, for long macros (sections of program) that have to run repetitively, more efficient *compiled* versions of the macros may be generated.

The language is *extensible*, meaning that the user may create new commands in the language as *programs*. These commands, however, may be used in later programs as if they were built into the language itself.

The system is completely general-purpose. Many graphics languages are not, being restricted only to their original purpose. This is more difficult, but oh, so much more worthwhile. As is said on the

Sandin (see right) wrote a program to display Peano lines that was under forty GRASS instructions long. It is also astonishingly reversible: you can watch it *uncreate* the Peano line, straightening itself backward.)

Never mind his age, the more esoteric features of his system (full recursiveness, etc.) or the fact that he does not seem to

**You may wonder how a young bronking buck like DeFanti has managed to do such an excellent job, so elegantly, where so many have stumbled and fallen?**

**"I just learn from other people's mistakes," he says cheerily.**



Prof. DeFanti on the system.

**Coupling his system with that of Dan Sandin (right) has created the "Circle Graphics Habitat," described on p. DM 98.**

## SANDIN'S IMAGE PROCESSOR

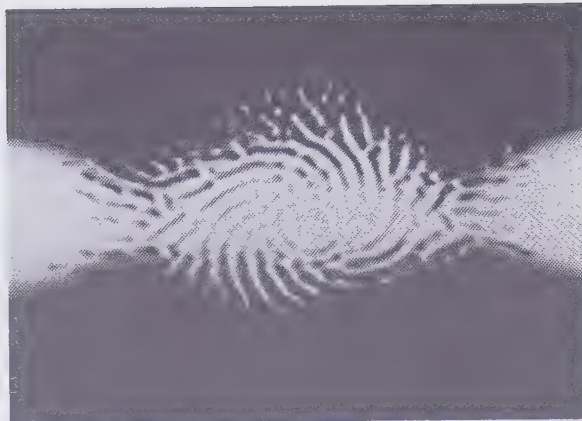


Dan Sandin, professor of Art at U. of Illinois, Chicago Circle, says very wise things (having been a physicist), and we were going to have a whole section on *that*, but, as you can see, there wasn't room.

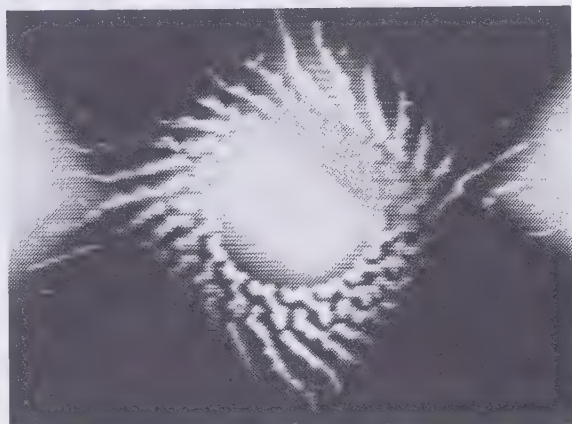
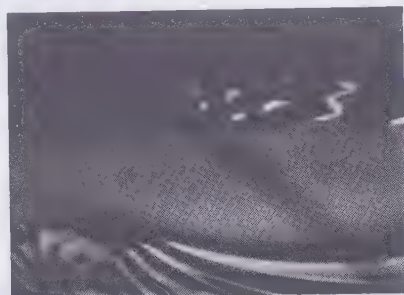
Daniel J. Sandin (pronounced san-DEEN) has spent the last several years putting together a device he currently calls the IP (Image Processor). It's a system of circuits for changing and colorizing TV. 🖨



# Sandin's system is VVO - Video In, Video Out.



What Dan's processor can do to television is not to be believed. Savage colors or delicate off-whites, solarizations and pictures on top of pictures. Then through "video feedback" (pointing a TV camera at a TV screen), the system can generate throbbing animated cobwebs and spirals of its own. Shown.



The idea is basically to create a completely generalized system for altering the color and brightness of video images. (i.e., the system does *not* move them on the screen. Thus it differs from the Computer Image line of video-twisting graphics systems, which alter positions of objects; see pp. DM 112-117. Note also that rather similar facilities exist as part of, e.g., the Scanimate system, p. DM 115.)

This means that basically Sandin's system plays with the part of the TV signal called *z*, or brightness (as distinct from *x* or *y*, the signals for horizontal and vertical movement of the dot. See pp. DM 84-86).

Now, as a physicist and field-theoretician, Sandin approached this as a problem in generality; and indeed, the style of generalization should be appreciated. Sandin repeatedly chose flexibility and power rather than obviousness in the parts he created. The resulting system is both parsimonious and productive.

His first important decision was that

all parts of the system should be *compatible* and *idiot-proof*, so that any user could frivolously plug it together any way at all without burning out the circuits.

Indeed, Sandin decided to build it like a music synthesizer: by making all systems electrically compatible (as they are on the Moog and its progeny), any signal can be used to alter or influence any other signal. This is a very profound decision, whose far-flung results have not yet been fully explored even among Sandin's rather fanatical students.

Basically, the incoming video image is "stripped" of its synchronizing information, so that all signals turning up in the guts of the machine may be freely modified. Only at the final output stage are the jots and tittles of the video signal put back on.

Thus the first and last blocks of the Image Processor act like bookends, between which the other modules have their fun. The first block makes the incoming signal into "naked" video, the last block dresses it up respectably again.

## THE CIRCLE GRAPHICS HABITAT

It is usually hard to combine things: especially complicated technical things. Usually it takes infinite reconsiderations, finagling, modification, intertwining.

The Circle Graphics Habitat, however, is something else again. It results from two intricate, independent technological developments, each an intricate system carefully crafted by an exceptionally talented person, coming together like two hands clapping. Like ham and eggs, like man and woman, Sandin's Image Processor and DeFanti's GRASS language conjoin directly and interact perfectly as if they had been made for each other, which they were not.

Dan Sandin's Image Processor (see p. DM 97) is a system of circuit boxes that allow video images to be dynamically colored, matted, dissolved and palpitated; Tom DeFanti's language (see "Coup de GRASS, p. DM 96) permits the rapid creation, viewing and manipulation of three-dimensional objects on the screen of a particular computer setup.

To combine them, you just point Dan's system at Tom's system.

Let's say that on the screen of Tom's system we are viewing an animated bird, flapping its wings. Since it's being shown on a three-dimensional refreshed line display (see pp. DM 89, DM 88), it appears only as white lines on a dark screen.

Dan merely points a TV camera at Tom's screen, and runs the TV signal into his Image Processor. Now, in the Image Processor, he gives it the magic of color. *Different* colors, interplaying with gradations and subtlety.

From the Image Processor, the finished signal goes out to videotape recorders.

What then have we overall? One of the world's most flexible facilities for the rapid production of educational videotapes.

To explain something, you create a three-dimensional stick-figure "model" of it, using DeFanti's GRASS language. Then you make a videotape of it, showing



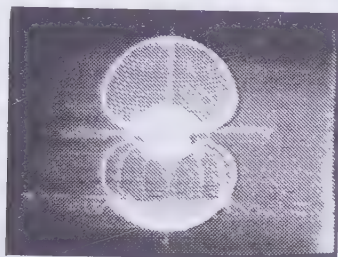
rotations or other manipulations, using the Image Processor to give it color.

DeFanti and Sandin have spent much of the academic year '73-74 getting the kinks out of this procedure. (Many of the difficulties stem from the unreliability of videotape recorders.) Stills from some of the first work are shown here.

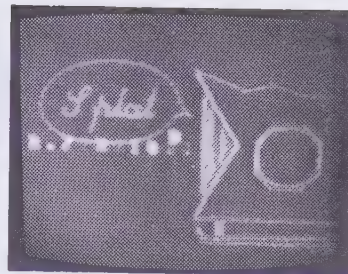
## BIBLIOGRAPHY

Thomas A. DeFanti, Daniel J. Sandin and Theodor H. Nelson, "Computer Graphics as a Way of Life." To be presented at U. of Colorado computer graphics conference, July 1974; to appear in proceedings purportedly to be called *Computers and Graphics*.

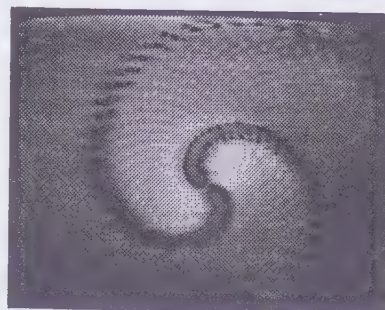
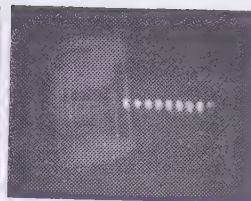
*This turned out to be the first SIGGRAPH.*



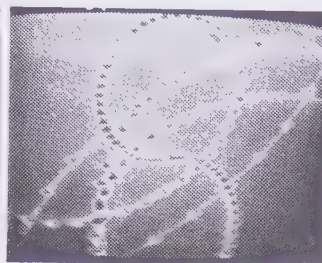
From videotape, "The Hydrogen Atom According to Quantum Mechanics" by T.J. O'Donnell & David Parrish.



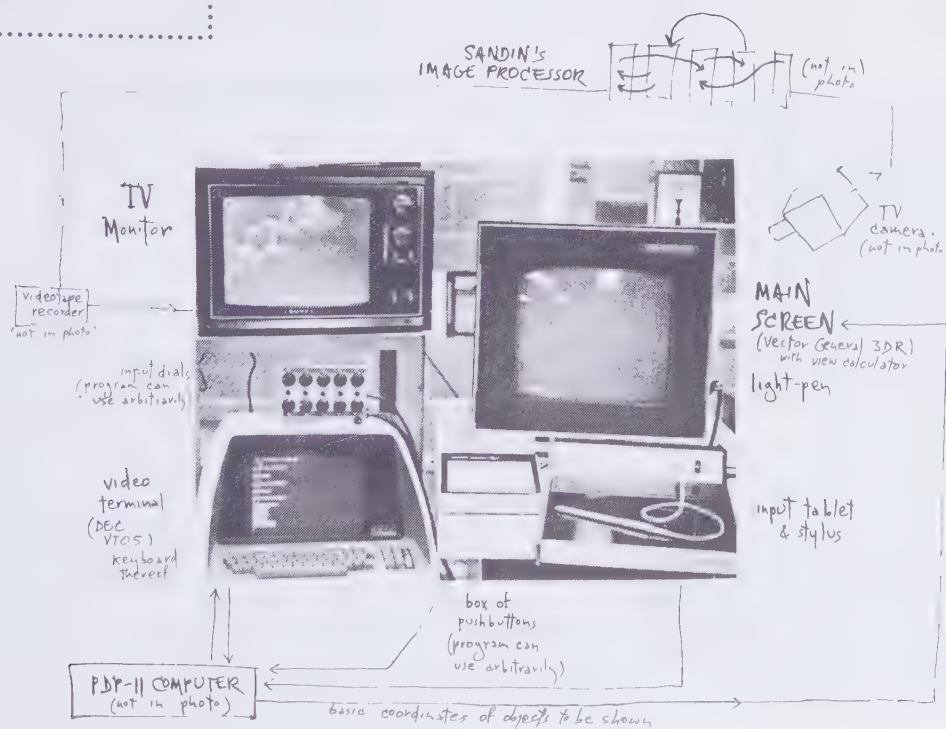
From videotape, "The Number Cruncher," by TDF & DJS.



From videotape, "The Spiral Tape," by DJS and TDF.



## THE CIRCLE GRAPHICS HABITAT



## Flowers by Wire:

# DVI from RCA

A surprise announcement in March of '87 was the spectacular DVI chip-set from RCA.

These chips are billed by RCA as offering "video-compression"—that is, they allow video to be squashed down to a much more condensed signal that can be sent through a much narrower channel. Specifically, video normally wants at least a megabit of channel capacity, but DVI creates a signal that only needs a 150,000-bit capacity.

That substantially-crunched signal is then decoded by the DVI chips. These are basically RISCs (Reduced Instruction Set Computers, see p. 91), but specially programmed to make the video whole again.

This has implications.

It means, for instance, that video can be stored on computer storage media (even floppy disks).

It means far more gorgeous computer graphics. (The demos have beautiful and subtle color. The video gets a little boxy-looking in complicated sequences, but it's still darn good. For instance, in a scene where people shoot the rapids in a canoe, the spraying water momentarily looks like animated rectangles.)

You can also stretch and distort the video in useful ways: in one demo, for instance, the user could pan a virtual camera all around a scene—yet the scene was stored as a single hemispherical picture, and each "camera angle" was a redistortion back to the standard-perspective rectangle.

But I haven't told you the best part.

The DVI chips map living video onto the facets of wire-frame objects. Gorgeous and fantastic 3-dimensional effects can be achieved. Thus the DVI chips can become the basis for new flight-simulator-class workstations at very low cost, bringing super interactive graphics into the school and home at incredibly low prices within a few years. Just what we need to really fix education—assuming the right software and contents. (See "The Playstation," p. DM 23.)



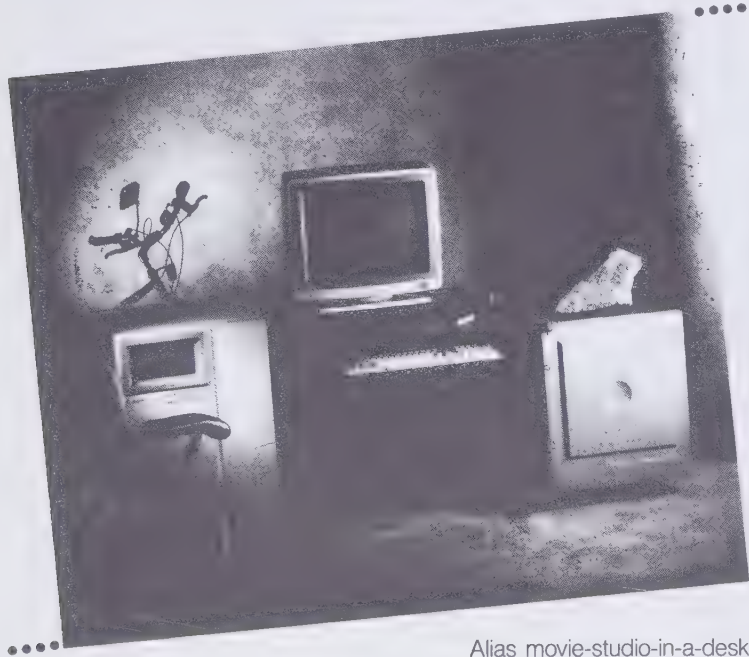
## ►The New Forgery

Today you can still tell the difference (usually) between photography and image synthesis. Tomorrow no more, as it will be with recordings as well. This means the evidentiary value of photographs will have come to an end. To paraphrase Peter O'Toole as the director in "The Stunt Man": we can film John Kennedy as if he were being shot from any angle, or as if he were the gunman. Woody Allen in "Zelig," spliced into photographs with major figures of the 20th Century, is the symbol of the new forgery.

### YOU GET YOUR TURN—AFTER INDUSTRY

*Breathes there a kid of any age who doesn't want her or his own image synthesis system? ("A movie studio is the greatest toy a boy ever had." —Orson Welles.)*

*Soon, soon. Before the end of the century, if we survive. But right now the prices are up where industry will pay them, in the hundreds of thousands, and so don't expect the prices to fall where we want them before the big customers have paid a lot. Big spenders are first to the trough. (Various vendors are now bringing out PC software for image synthesis; West End Film (Washington, DC) and Time Arts (Santa Rosa, CA) are two. But prices are still industrial (in the thousands).*



Alias movie-studio-in-a-desk

## COMPUTER IMAGE SYNTHESIS

# THE TOTAL FORGE

The moviemakers are not ready for image synthesis—the dreamsmiths unprepared, as it were, for the Total Forge.

That was then. This is now (1987). We are now so accustomed to image synthesis, on TV and in the movies, that it's hard to remember how fast this has come upon us. We think nothing of TV titles and station breaks where we fly through animated letters that suddenly look like huge buildings, of cars that dissolve into transparency, of electric shavers that roar through mountain passes.

In movies, "Futureworld" had perhaps the first seriously synthesized sequence; then came "Tron," a strange 1930s-nostalgia-style science fantasy that used image synthesis from a number of companies. "Tron" was historic, but not a big hit.

"Star Trek II" had a synthesized sequence (by Lucasfilms; directed by Alvy Ray Smith) so breathtaking that they used it again in "Star Trek III." This was the Genesis Bomb sequence, which showed a bomb unleashed on a lifeless planet that first caused a great fire to roar concentrically around the planet, then showed the planet transformed in greenery and loveliness. (A leaflet I have seen indicates that at least one revival preacher was using this clip in his Apocalypse spiel.)

And "The Last Starfighter" had great intergalactic fleets of spaceships with nary a physical model ever built; all were simulated in software by Whitney and Demos' Digital Productions (see nearby).

The novelty is over; the Computer Synthesis Industry proceeds.

And it is very competitive.

Special equipment is available for tens of millions of dollars (from Evans and Sutherland Computer Company, Salt Lake City; discussed nearby) to hundreds of thousands (the Pixar; see nearby) down to thousands (the Z-buffer board-set from Renaissance graphics).

Software packages are available for great sums (leases in the hundreds of thousands) down to shareware for the Amiga. (One Eric Graham published a program in AmigaWorld for building

worlds of spheres and photographing them.) There are various turnkey packages. One Toronto firm, Alias, offers a slick and complete movie-studio-in-a-desk, software and hardware all set up to roll for about \$125,000. (For another \$91,000 they'll throw in a Pixar; see p. DM 113.)

Allow me to predict firmly that image synthesis movie-making packages will soon be a big bandwagon in the personal computing market—after the "desktop publishing" fad has run its course. At much more affordable prices. You'll sketch and animate your movie on your personal machine, then send the disks to somebody with an Alias (or the like) to get a videotape or film of it. (As predicted a little early in the original Computer Lib; now at last the time and equipment are ripe.)

Hundreds of "image synthesis houses" crowd the movie industry, vying for TV-commercial business and the occasional biggie ("Hitchhiker's Guide to the Galaxy" is rumored to be in planning).

But the competition is cutthroat. Among the heaviest players have been George Lucas, Bob Abel, and the Whitney-Demos team; but fortunes change fast in this industry. (New York Institute of Technology, a pioneer in the field, looked in the late seventies as though it would be a big force, but most of its graduates have dispersed to other houses.)

### JOHN WHITNEY, JR. AND GARY DEMOS

Many unlikely individuals have stormed that heartbreak town of Hollywood, leaving sadder but wiser—

but nothing stops these guys. The first to enter Hollywood with image synthesis software, they built the movie-making setup at Information International, Inc. into a position of leadership and did many firsts, including the famous Juggler sequence.

Then they started their own company, Digital Productions, bought a big Cray computer and redid the software. Their best-known projects were "The Last Starfighter" (for which they did **all** effects),



the Jupiter animation for "2010," and the extraordinary owl in the opening of Jim Henson's film "Labyrinth." (Another project, a 3D sequence for a Japanese fair was especially extraordinary.)

Now they have yet another company, Whitney/Demos Productions, and they have bought a Connection Machine (see p. 93, Lib side). Preliminary results indicate that this supercomputer will get them a throughput of 2.5 gigaflops, or almost ten times what they were getting from the Cray. Which means more realism for the buck.

Whitney and Demos are funded by Symbolics, the big-money LISP company (see LISP, Lib side p. 61). Symbolics' LISP machines connect to the Pixar, and the new Whitney-Demos code will supposedly run on the Symbolics machines as well as the Connection Machine. So the LISPIANS, who a decade ago may have seemed to be anemic wimps, may yet rule the world of computer image synthesis.

## Jim Blinn!

Amongst the Teraflop Club—the computer-movie superhackers like Whitney and Demos, Nelson Max, Turner Whitted and Y. Kawaguchi—legendary and extraordinary Jim Blinn is perhaps the dean.

Immensely tall (and still a longhair), Blinn is a commanding presence at SIGGRAPH and other elite gatherings.

SIGGRAPH types are usually good speakers, but Blinn is very special: he can explain mathematics like no one else—intuitively, playfully, brilliantly. One hacker's hacker calls him the greatest teacher in the world.

He is best known for the extraordinary films he made at Jet Propulsion Laboratory in Pasadena that showed what various spacecraft would look like, and what they would do, as they passed various planets. This glorious footage turns up constantly whenever space is mentioned on TV, usually without Blinn's name attached. (A lot of people evidently think these are **actual movies** of the spacecraft in flight.)

Blinn has sojourned at Lucasfilms; most recently he has spent several years singlehandedly preparing a physics course for television, all done by computer graphics, with simulations of various kinds and a new kind of **animated algebra** that should have a lot of impact. And I mean **should**.

### TERMINOLOGY

*When these pieces first came out, I used the term "halfstone." That term has not caught on, but the following have: "image synthesis," "imaging," "raster" (as distinct from line-drawing), "shading" and "rendering."*

A Series of Review Articles for *Computer Decisions Magazine*.

## COMPUTER HALFTONE IMAGE SYSTEMS

*Author's note.* These articles were written for *Computer Decisions* magazine, and reflect the results of a lot of phone calls they paid for. The first of these articles was published in 1971. The others have not been previously published, as the editors and I were never able to get together on quite what they wanted.

This is, to my knowledge, the only existing collection and summary of computer halftone systems to date, and in some cases the articles reveal more about the systems than has been published anywhere.

*These pieces are presented for their nostalgic and tutorial value, with no apologies.*

There are more ways than one to produce shaded pictures with computers. Here are the methods of the 'polygon school.'

As 3D work progresses, large prob-

lems are being overcome. The famed "hidden line problem," for example, was misleadingly couched, since the problem is not finding what lines are hidden, but what surfaces are in front!

## HALFTONE IMAGE SYNTHESIS

by Theodor H. Nelson

The Nelson Organization

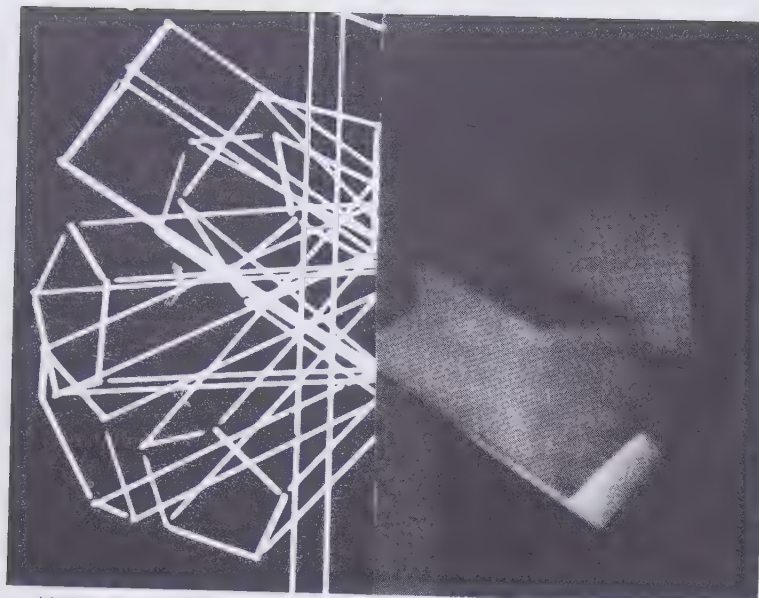
To most people in the computer field, "computer graphics" means line drawing—systems and programs for map-making, pipe layout, automobile and aircraft design, or any other activity where a diagram may help. Using line-drawing programs and equipment, designers may create line drawings on fast-responding graphic screens, reworking their ideas until satisfied; the system then disgorges polished drawings and specifications for the designer's real intent, something else that is to be made or done. But it is possible for a picture itself—instructive, interesting or pretty—to be the goal. In that case we will often want pictures that look like things instead of wires. A picture that is not all black and white we call "halftone."

With much secrecy and a slow start, computer halftone systems are now being built all over. The methods are extremely different from one another; only the outputs are similar. Some exist in software, some have already been built into special hardware. These systems have many potential uses for visualization, animation

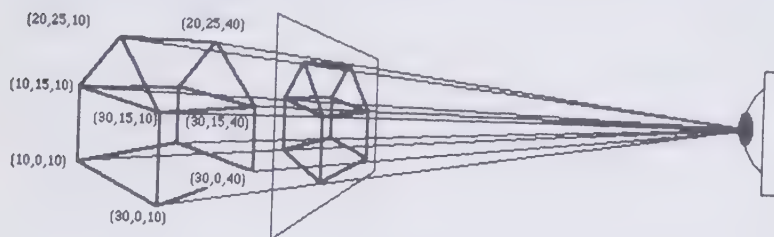
and new kinds of photography, in art, scholarship, motion pictures and TV; for visualizing worlds lost and imagined, equipment yet unbuilt, the responsiveness of aircraft. It may not be long until moviemakers can buy different brands of picture synthesizer, just as musicians choose today among Moog, Buchla and ARP music synthesizers. But none is in production yet. This is an attempt to review the coming apparatuses of apparition.

Not only is the field of halftone one of the most exciting in computing; it is also one of the nuttiest and most secretive. For instance, at one time a firm that was supposedly marketing its halftone system declared the present author *persona non grata* and not to be communicated with in any way, though information was freely available to others. "I don't think it's necessarily paranoia," says Rod Rougelot of General Electric. "A lot of guys started about the same time, and proceeded in a heads-down manner." It took a special kind of initiative to head off in that direction with no external provocation. "All those heavy cats from ARPA and MIT were saying in the sixties I could never do

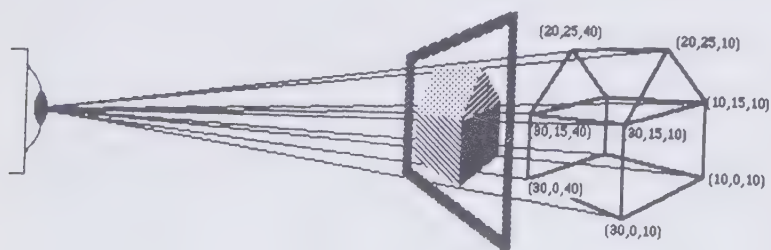




Various computer methods now make it possible to create artificial photographs of three-dimensional objects or scenes represented in the computer's storage. This is done by coloring or shading points in an output picture like the points in the scene that can be sighted through them from the vantage point. What the methods really boil down to, though, are searching processes in the data representation of the three-dimensional scene.



*Computer graphics the ordinary way.* The computer, as penman, draws lines from a stores list in core memory. In a three-dimensional system, the basic list of 3-D coordinates is converted to a list representing a particular view; the result looks like a wire frame.



*3-D halftone system.* Today's new procedures can use the same data to make a realistic shaded or halftone picture. The visible parts of the objects are ascertained by programs or special hardware, using the same 3-D coordinates as in the ordinary systems. These visible parts are then shaded according to the appropriate color information. The series of shading-points makes the picture on an output device.

a Mickey Mouse," says Lee Harrison III of Computer Image. "But I'm not that kind of researcher. I talk to the Lord."

The systems' stories are as different as the systems themselves. General Electric's system grew out of cockpit displays for blind flying. The system of Pennsylvania Research Associates began with terrain and radar modelling. The system of MAGI (Mathematical Applications Group, Inc.) began with the study of radiation hazards in battlefield machinery. Two system families, that of Computer Image, Inc. and my own Fantasm, were designed from the beginning for moviemaking, especially "special effects" and puppeteering. The most poignant tale may be that of Lee Harrison, whose struggling family was warmed through cold winters by the tubes of their analog computer.

### HALFTONES IN TWO DIMENSIONS

Two-dimensional computer halftone is not new. Halftone pictures converted from photographs have often been printed out on line printers, either for fun (nudes often turn up at big installations), or in connection with some scientific problem, such as analyzing chromosomes. Kenneth C. Knowlton, at Bell Laboratories, has executed some well-known photo conversions making pictures into huge grids of tiny whimsical symbols having different grey-values (see pp. DM 81-82).

Various other systems have allowed users to create their own original 2D pictures. But the natural temptation is to want the computer really to make pictures. Why not have the computer produce a photographic picture directly from the 3D representation of objects? Computers don't do this by nature, any more than they do anything else by nature, so how it may be done by computer is very interesting. The problem is also interesting because of its intuitive nature. Visions of scenes in space are around us constantly, and we intuitively understand the geometry of outlines and light.

We must draw on this understanding of scenes to figure out how to make pictures, for there is no mathematically elegant or preferable approach. Scenes are geometrically rich, and thus many different techniques may be used to extract pictures from them. These techniques may look at planar structures, spatial in-

terconnections, relative edges of intersections or anything else you can define and process. I prefer to think of computer halftone as like trick photography of the kind done in Hollywood: a variety of techniques can be combined in various ways. As in trick photography, the number of touches and enhancements that you add generally determines how good it will look, regardless of what system you begin with.

The simplest systems are those that depict objects made of polygons—that is, planes with straight edges. We will discuss such systems in the present installment.

### THE WILD POLYGON YONDER

At least two companies are building image systems that will behave and respond like onrushing reality. Such a system, hooked to cockpit-like controls, can show a trainee pilot the delicate and precipitous results of what he does. Realistic action, rather than surface detail, is crucial.

The techniques of action polygon halftone were originally developed by General Electric, of Syracuse, N.Y., and are now also under development at Link Division of Singer Company (makers of the beloved pilot trainer and its progeny). Basically such systems operate upon the scan-lines that crisscross a television screen, switching the color of the running scan as it crosses from polygon to polygon.

The action polygon school—GE and Link—takes a curious but effective approach to halftone TV: their "environments" are composed entirely of convex objects made entirely of convex polygons. To use only convex objects (no dents) means that one object may be in front of another or vice versa, but never both. (An object with apparent indentations, such as an airplane, has to be made out of a group of convex objects flying together.) To use only convex polygons (notchless) makes it easy for the system to decide, at a given instant, whether the scan is crossing the polygon or not.

This work evolved in part from GE's work in the fifties with a "ground plane simulator," a system that would show a correct representation of the ground's position, dipping and rotating, to the pilot of an aircraft in fog or night. In 1963 the General Electric group, under Rod Rougelot, worked out for NASA the de-

sign of an "environment simulator"—a device that would simulate the appearance and performance of any equipment. This is now called the "old NASA system." It permitted the user—seated before a color TV screen—to work controls for an imaginary aircraft or spacecraft, and see roughly what the pilot of the craft would see, flying in real time through a breathtaking color scene. Films made on this machine have been stunning. Imaginary cities, roller coasters and aerial dogfights are among the visions that can be presented.

General Electric's old NASA method is fairly weird if not mischievous. The earlier "ground plane simulator" had shown an edge (the horizon) digitally displayed on a CRT; the system was extended to

many edges, and the logical analysis of areas between them.

The scene was represented by a collection of edge boxes, physically jumpered into a collection of facet boxes. Each edge box and facet box was loaded with certain numerical and logic values, representing edges and facets in the scene, which could change between frames as required by the action.

In the preprocess for each frame the old NASA system used a specially built digital computer, the "vector calculator." This performed at great speed the three-part vector calculations necessary to determine all scene positions, including the positions and slants of all edges. Each individual edge generator, loaded with its own edge position, constantly reported to

whether the running scan of the picture was to the left or right of its own edge. It dutifully guarded this edge from border to border of the picture.

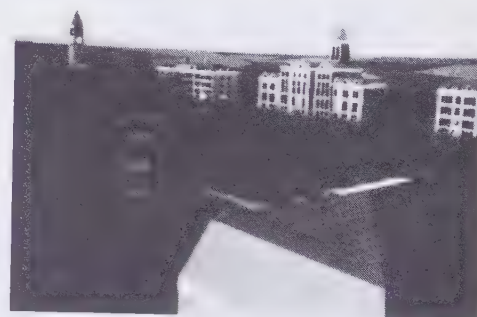
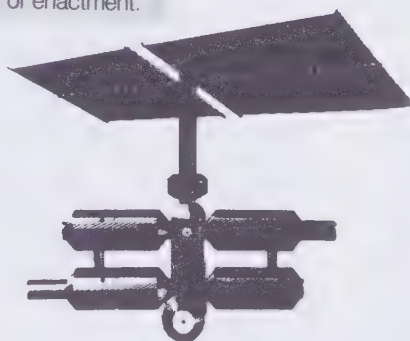
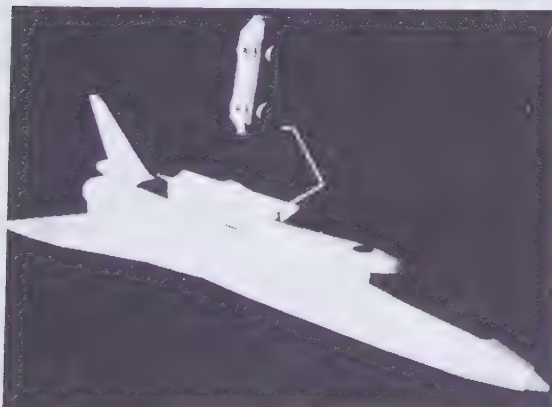
"Old NASA" method: Each edge box constantly reports which side of its edge the scan is on; each facet box sums the edge reports to sense when the scan is crossing it.

The edge-box reports summed into the facet boxes, each of which was set to respond to a particular combination of left-right, above-below reports. At the instant all the facet's edge boxes replied in the proper preset combination, the facet box signalled that its own facet was being crossed by the scan-line. When more than one facet-box responded, the one nearest the viewpoint had its color gated to the screen.

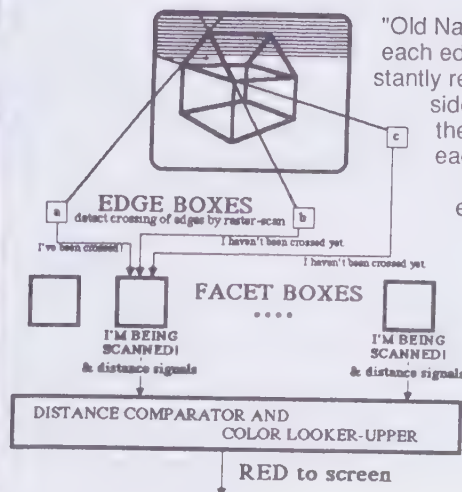
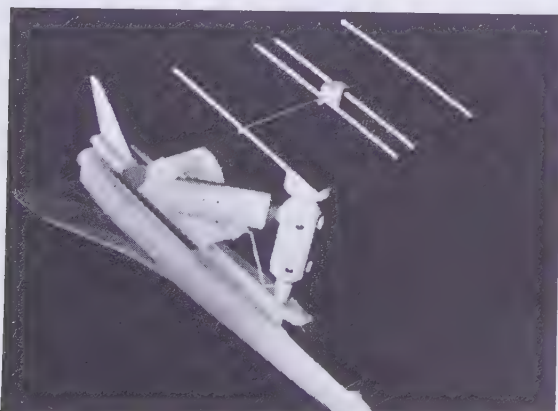
Now Rougelot's group is replacing the old NASA system by a new NASA system, which works on entirely different principles, but keeps the vector calculator. The old one could show scenes with up to 240 edges; the new NASA system will at least double that. GE's new method is already operational on smaller research facilities. They don't tell what it is, but basically it involves sorting by distance. Supposedly the sort method is good enough to make the old edge boxes obsolete.

The Link group claims competitive performance for their system, which will go to black-and-white thousand-line TV. They say their system is different, better, and secret. ■

General Electric will make movies and videotapes for you with their pictorial synthesis system. These are from a beautiful (really *beautiful*) film they did for NASA. The point of the film was to explain to everybody how a proposed space laboratory would be built and would function. Rather than use diagrams, they *enacted* it in the GE system, so viewers could understand how the sections would be delivered and fit together, how the antennas would unfold and so on. For exposition of that kind, nothing beats this kind of enactment.



Campus of Fooled U. (GE)

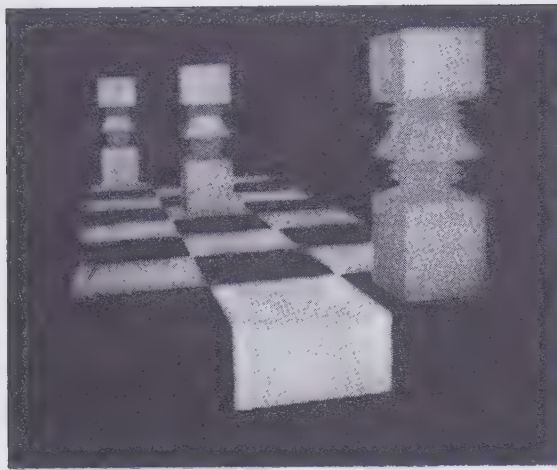


"Old Nasa method"; each edge box constantly reports which side of its edge the scan is on; each facet box sums the edge reports to sense when the scan is crossing it.

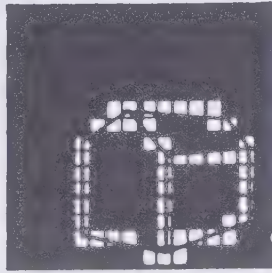
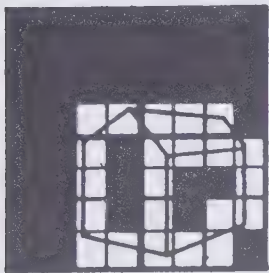
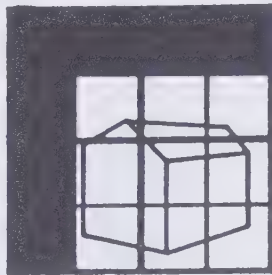
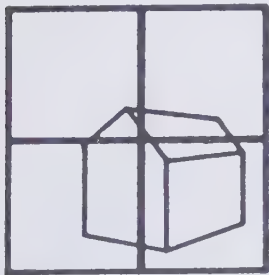


Instantaneous enactment: half-tone animation gives a sense of really being there. (Rod Rougelot, General Electric)





Halftone for art's sake: now the artist can create worlds and photograph them. (Gordon Romney, Utah)



Warnock's dicing method: What can't be made all one color is redivided till its pieces can be.

## WHERE ARE THEY NOW?

John Warnock has developed his obsession with fine-grain area fill into a whole industry—an area which he himself has filled. As head of Adobe Systems, he has presided over the spectacular success of the PostScript laserprinter language (see p. 59), which has now expanded into such areas as screen window management (see p. DM 23).

## WYLIE-ROMNEY: SHOOT THE WORKS

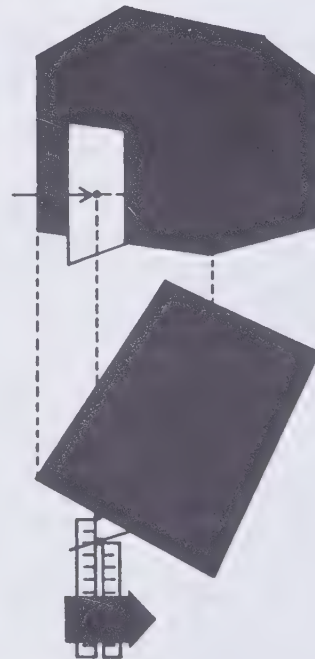
The Wylie-Romney method, disclosed in 1967, was the first generally publicized procedure for making halftone pictures. Indeed, the 1967 publication signalled the explosion of the University of Utah into the forefront of computing research.

The Wylie-Romney method was actually the joint work of Chris Wylie, Gordon Romney, David C. Evans and Alan Erdahl; but much of the impetus for its development came from Evans, chairman of computer sciences at Utah, who had long suspected the possibility of 3D halftone synthesis.

(Note: more output by various Utah systems appear on following pages.)

The Wylie-Romney method is this: for each picture-point desired in the final picture, shoot a searching ray through the scene at a corresponding angle. Find where this searching ray hits every surface in its way.

**NOW AVAILABLE! Machine running Watkins' technique, the Watkins Box, allows you to view imaginary objects in color and manipulate them in real time. (See p. DM 111.)**



Watkins method: A new nearest surface is instantly sensed through continuous comparison of the closest two.

Since the locations in space of these hit-points are easily calculated, figure their distances from the vantage point. The nearest of the intersections is the visible one. Look up the color of that surface and shade the output point accordingly.

This may sound inefficient, but it is comparatively easy to ascertain all the piercing-points, since the surfaces to be hit in a given scanning row can be largely predicted from the previous row.

## THE WARNOCK METHOD

John Warnock's method, also from Utah, is unrelated to the other methods, but has qualities mathematicians like, as well as a certain whimsy.

Consider a square in the picture area. (At the start consider the whole picture area.) *Now then.* Test whether the present square is entirely filled with one color. If so, output a corresponding square all of that color. If the present square is not all one color, divide it into four smaller squares. Take another square and go back to *Now then.* End the process when each of the squares in the broken-down picture has been completely filled with one color—or the unsatisfied squares are too small to care about.

## WATKINS

The method of Gary Watkins is the result of a profound search at the University of Utah for *the* method—a polygon technique fast enough for real-time enactment, but cheaper than the GE-type systems and not subject to the convexity restrictions. They seem to have found it.

Each video scan of the scene results in a "slice" through surfaces in the scene. The two nearest surfaces are continuously compared to see which is closer, as if by two rulers. The instant a new surface becomes the nearer one, the system makes it the visible one. The nearest surface always shows, down to the precise instant two surfaces cross.

## SHADING: LAST OF THE GREAT FUDGE-FUNCTIONS

Suppose that we have some data structure representing a three-dimensional object, and a halftone method to search out its visible surfaces. How do we shade the output points? What do we take into

account: how combine the basic greys or colors, how blend them with computation of surface angle, distances from the vantage point, or anything else we can think of?

The answer: any way at all. The combining function is an aesthetic choice. There are not many areas left where you can make up a mathematical hodgepodge and get pleasing or interesting results. Computer halftone is a felicitous exception: you can augment by adding or multiplying, diminish by subtracting or dividing, and yet always come up with an image resembling something. Anyone who has worked in a darkroom will recognize that this is like enlarging: playing with parameters won't obliterate the picture.

There are purists who insist that halftone coloration should exactly follow the formulas that simulate the behavior of real light. For some purposes, like pilot training, this may often be true. But insis-

ting on mathematical accuracy as a general principle is like insisting on ultra-high fidelity—an aesthetic judgment couched as a mechanical imperative.

Until now the output hardware was not really ready for halftone. Five years ago a computer could usually create halftone pictures only on a line printer or a 4020 microfilm plotter. Today there are many different photographic printers, going to all sizes of film and paper; one even uses a laser. There are various display terminals permitting grey-scale and color halftone on TV screens.

The age of computer image synthesis has begun. Polygon systems are fast and simple, and will come to be used in our daily lives for such diverse purposes as molecule study, the memorization of delivery routes, and visualization of every kind of layout and design. They will be fundamental to our new world of computer display.

## SURFACE PATTERNS, CURVATURE, SHADOW: SHADES OF REALITY

Various New Techniques Permit Us To Add Curves, Shadows and Surface Patterns To Computer-Generated Halftone Pictures

### THE PLOT SO FAR

One of the most interesting things about this branch of computer graphics—already seen in the polygon methods discussed earlier—is the variety of techniques that can be employed. Moreover, these methods, for all their sophistication, can usually be intuitively understood as though they were operations performed on objects in space. The same continues to be true for the more complex systems.

### ENHANCED POLYGON SYSTEMS

In the methods discussed so far, we looked at several computer techniques for photographically depicting scenes and objects made up of polygons—planar facets—in a represented three-dimensional scene. Imaginary houses of cards, cardboard airplanes and triangular scenery take on a compelling vividness when

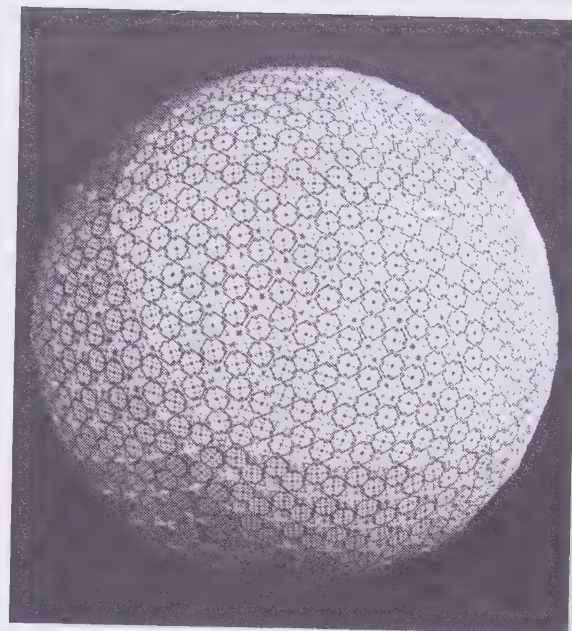
depicted by the computer. And for visualizing such things as architectural arrangements, such systems promise to be of increasing practical value.

Those of us interested in the artistic aspects of computer halftone images want more. This article looks at some ways to add the appearance of curvature and surface pattern to computer-synthesized images.

### MAGNUSKI'S PATTERNED CONSTRUCTIONS

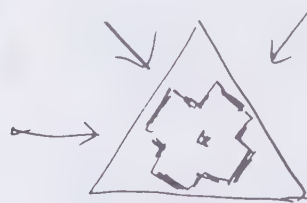
A number of contributions have been made by individuals working alone. For instance Henry Magnuski, at M.I.T., created a program that repeatedly positions patterned facets in space to make large constructions.

This program did not calculate "true" shadow, basing its shading partly on angle of surfaces. Neither does it show true curves. Yet it shows the impressive degree to which such effects may be approximated. The resulting beach ball picture is reminiscent of Moorish architecture.



## MAGNUSKI'S CONSTRUCTIONS OF REPEATED PATTERNS

(different perspective calculations)



Basic triangle pattern...



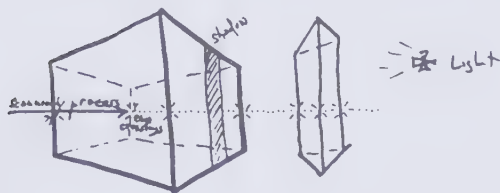
is stitched together in adjacent positions at appropriate angles.



Face by Lillian Schwartz (using Magnuski's system)

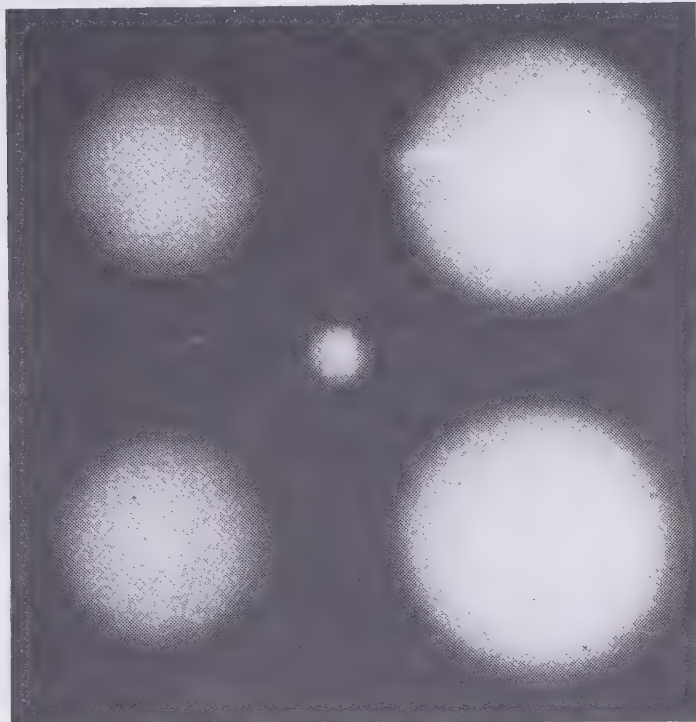
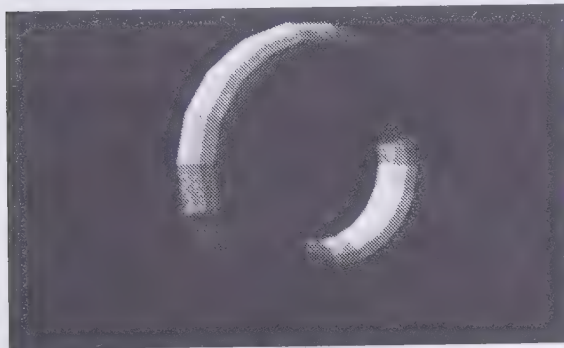


## BOUKNIGHT-KELLEY METHOD



Consider the series of edges whose projections cross the current scan-line. Each time the scan-line crosses an edge, find out what facets are currently pierced by a sight-line from the viewpoint. The nearest of these facets is the visible one.

To add shadow, use an extra list of the scene's edges relative to the light rather than the camera. Between viewed edges, check for shadow-edges as well.



Don Lee

## BOUKNIGHT AND KELLEY:

### PICKING THROUGH A CAT'S CRADLE

The method of Bouknight and Kelley, at the University of Illinois, permits the addition of shadow to polygon pictures. Their method uses an intricate system of scanning sweeps across the scene, analyzing the successive edge-crossings. For each output line, a list of the edges in the scene is ordered according to which will be next encountered. To make a specific output line of shaded points, we step through successive positions of the scan-line, until an edge is crossed. With each edge we cross, we enter or leave at least one facet. Of all the current facets we are in after a given edge-crossing, the system finds out the nearest one, the visible one, by comparing distances. The coloration of this facet is then fed out to the picture, until the next edge-crossing.

Bouknight and Kelley expand their method to show shadows by an additional step. They create a new list of edges to be encountered, this one relative to scans from the light source. Then, during the regular output picture scan, they look to this latter data to see about shadow. As soon as they know two consecutive edges of a visible object in the picture, they are able to search the shadow-edge list to see if any shadow-edges impinge between them. The final list of edges—visible facet edges and shadow edges—goes to the picture output device.



Bouknight and Kelley

## DON LEE FILLS IN THE GAPS

Don Lee, at the University of Illinois, produced his fine-toned pictures of spheres in 1966 simply because someone bet him a quarter he couldn't program the method he'd suggested in twenty-four hours. He almost made it. He made his pictures of spheres and polygons by calculating the boundaries, then checking for overlap and filling in with greys according to viewing angle. His program works only in special cases, but is interesting for its historical position; it was one of the earliest half-tone curvature systems.

## SIMPLEX CURVATURE SYSTEMS: MAHL & MAGI

A fundamental type of system we may call the "simplex" system was exemplified in the previous article by the Wylie-Romney program. A simplex technique simply projects simulated rays toward the scene from the vantage point till they hit the represented objects, and fills corresponding positions on the output picture with the colors encountered on the front surfaces of objects in the scene.

The same principle extends naturally to scenes with curved and otherwise embellished objects.

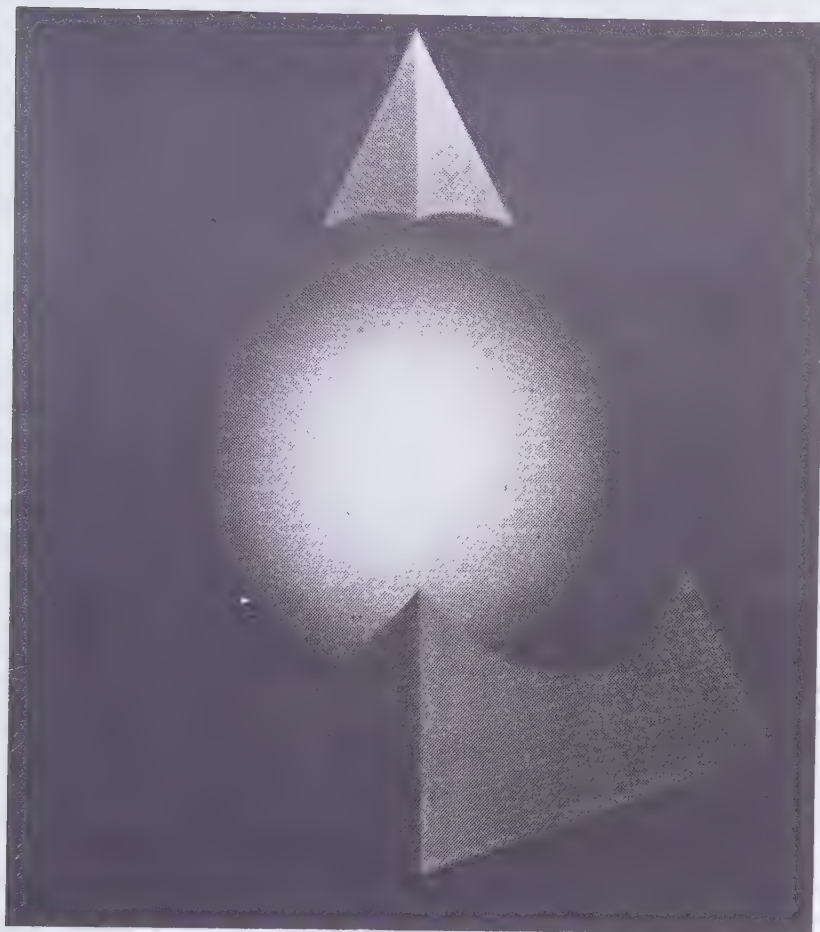
Robert Mahl, at the University of Utah, has recently reported his results with simplex methods using quadric surfaces—those curved surfaces generated by mathematical powers of two. His pictures—like the cup and saucer shown nearby—have a pleasing 1920s Bauhaus-like quality.

One problem with this method is that computational complexity increases rapidly as the scenes grow more complex; the more surfaces and piercing-points, the more time-consuming (and expensive) it becomes to make the picture.

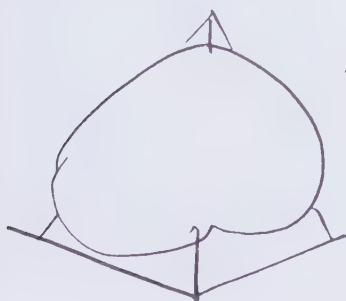
## MAHL'S SIMPLEX METHOD

It seems, however, that Mahl's work may only be a rediscovery of what one organization worked out earlier and is being secretive about. A firm delightfully called MAGI (Mathematical Applications Group, Inc.) of Elmsford, N.Y., has extended the same idea more elaborately. They happened into the half-tone game through a military contract.

# HAVE A BALL WITH DON LEE



His program first works out the general outlines.



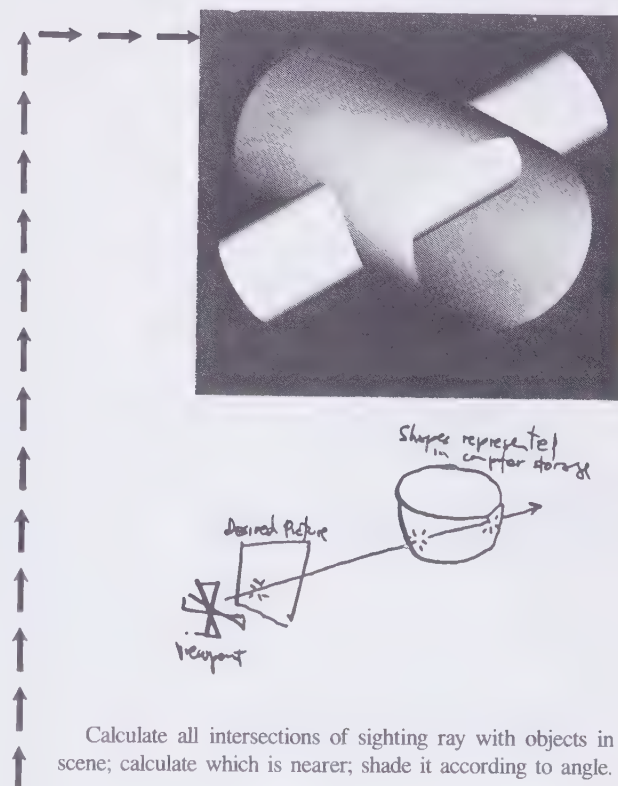
Then fills in curvaceous shading.



Robert Maht



Note: The accepted general term for what is here called "the simplex method," is now **ray tracing**.



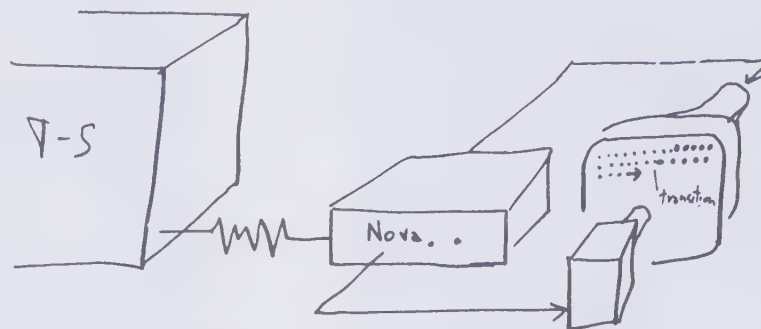


MAGI's system, now thoroughly developed under Robert Goldstein, began in 1965 in a study of radiation hazards in battlefield equipment. They wrote a program to simulate paths of radiation, say, that might reach a tank driver under various disagreeable circumstances. Having written a program that would ascertain the susceptibility to radiation of battlefield machinery, they noted that the same program could be adapted to making photographs. The program simulated radiation; light is radiation; ipso facto, pictures. Substantially the same program would make photograph-like images, by treating the objects as opaque, and reflecting different shades according to color and angle of view.

The resulting system makes nice pictures of objects composed of planes and quadric surfaces; and includes, as will be seen from the racing car and chair, colored surface designs, shadows and spectral reflections. Not only does MAGI's software for this process produce delicately shaded pictures; if the virtual pic-

ture-plane is moved until it intersects the subject, it produces a cross-section.

MAGI runs this program remotely in FORTRAN on a big computer—but they have their own minicomputer setup for photographing the results as color movies. They now offer use of this system commercially for making movies or stills.



SYNTHVISION SETUP uses remote time-sharing computer, running big secret FORTRAN program and containing entire data structure of three-dimensional scenes. Minicomputer photo-

graphic setup is on premises at Computer Visuals, Inc., MAGI subsidiary marketing the Synthevision service.

Local setup uses Nova minicomputer controlling both CRT display and camera. Informed guess would suggest that time-sharing system does not send all

successive points of output line, but difference and transition values; Nova program would then interpolate gradations in relatively quiet sections of the scanline.

MAGI's precise system is secret. However, the only real questions boil down to: forms of surface representation; systems of scene sorting; and method of scene scanning to produce output scan.

Note that one of the most impressive things about MAGI work, at least for sophisticates, is the degree of artistic control that seems to have been realized in their input and revision systems. It seems they offer excellent control over motion and color, and, of course, revision of the action in a scene till the maker is satisfied.

*Popular Science*, I think it was, had a spread on Synthevision in fall of '73.

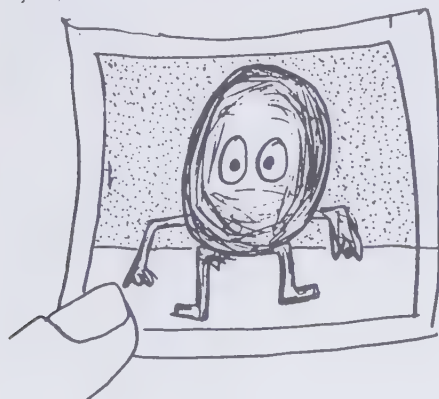
## ROUNDUP

These have been some of the highlights of the halftone game to date. The methods described so far are mainly software-oriented, and for the most part work most efficiently as programs. In the next article we will look at some outlandish new forms of equipment, under construction or proposed, for dedicated production of 3D halftone pictures.

## GENERALIZED SIMPLEX METHOD, AS EXEMPLIFIED BY MAGI SYSTEM

### HOW RAY TRACING WORKS:

To make a finely-shaded half-tone picture of a curved and patterned object,

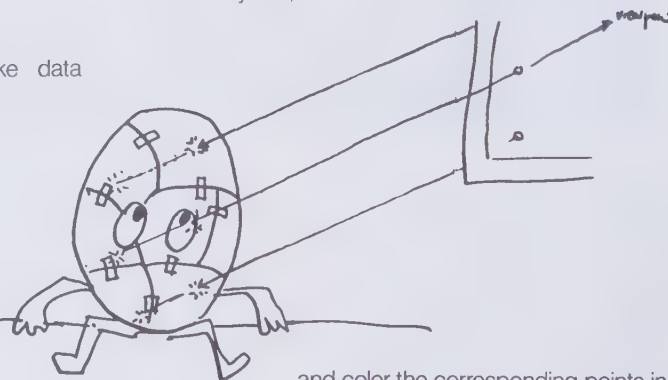


tie together these shell-like data structures in a unified scene,

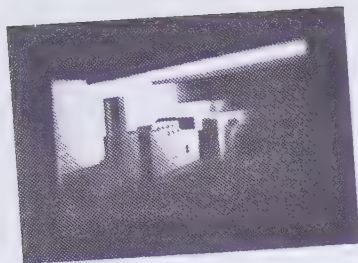


there must be ways to represent the individual current surface pieces, assign colors to their surface details,

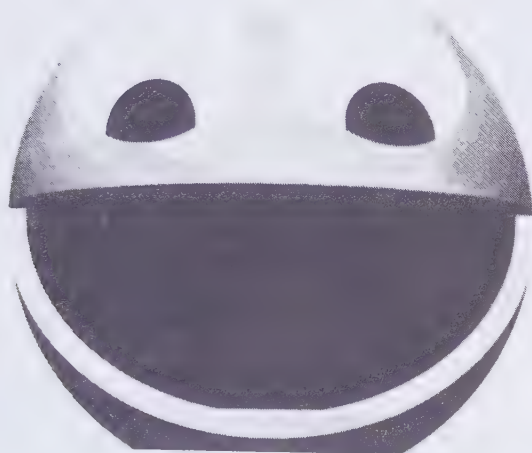
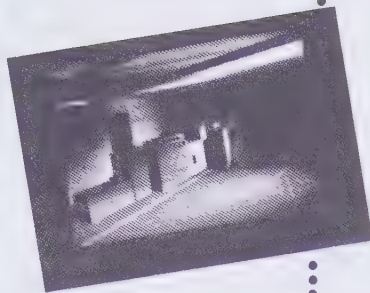
"sight" them with individual exploration rays, select the nearest point each ray hits,



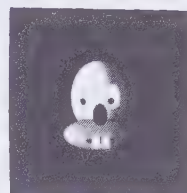
and color the corresponding points in the picture according to a blend of surface color, angle, shadows, specular reflection and whatever else turns you on.



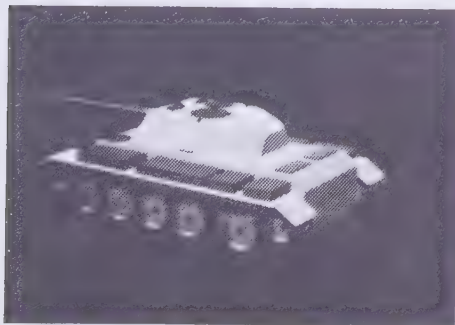
MAGI techniques were used to study alternative ways of lighting mines.



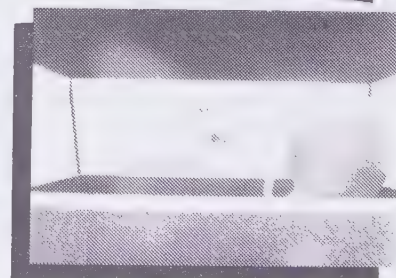
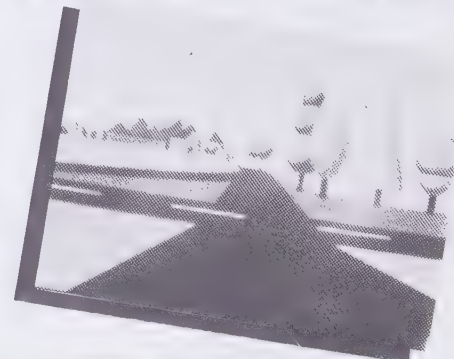
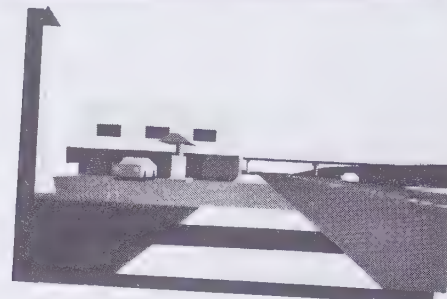
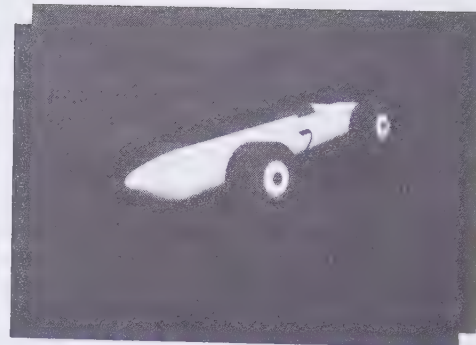
Enlargement from MAGI film. I hope the reproduction shows the concentric rings, called Mach bands, that divide areas of shading; Knowlton and Harmon (citation p. DM 82) advise on pseudo-random techniques for correcting this.



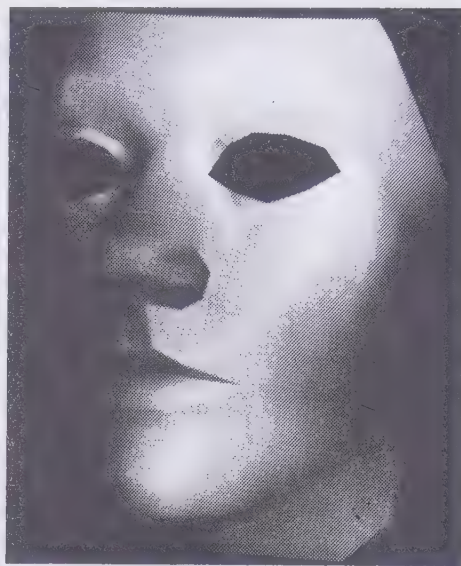
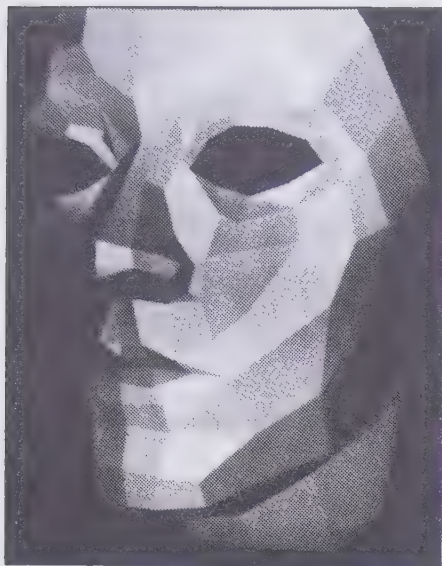
An early MAGI character.



MAGI program was originally developed for study of radiation hazards inside military armor; the pseudo-photographic techniques were a *side effect* of the approach chosen. Who knows, these tanks may be the ones studied.





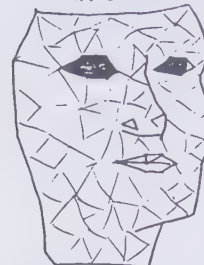


Results of Gouraud's swell smoothing technique. Mme. Gouraud posed for the data structure on the left, a system of interconnected flat polygons. The Gouraud process (see nearby) created the smooth-looking face from it by an extremely simple process. (Note that the power of the technique is in the use of a simple polygon data structure, rather than the more difficult truly-curved surfaces used, e.g., by MAGI.) (Note also that the edges remain jagged.)

# COMPUTER DECISIONS

[black background]

MIRROR, MIRROR  
IN THE KNOW:  
DO MY FACETS  
HAVE TO SHOW?



[pink green]



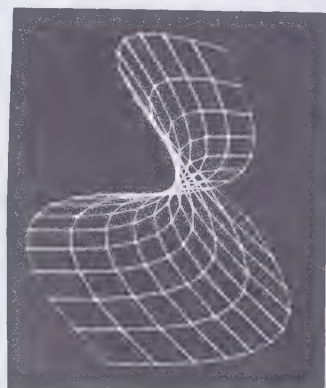
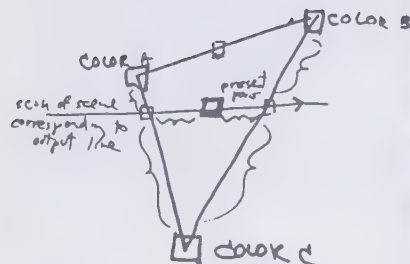
[pink blue]

TRY THE METHOD  
OF GOURAUD.

COMPUTER DECISIONS. I suggested this cover for this article. The folks at *Computer Decisions* reacted with puzzlement if not dismay. "This cover doesn't have *practical applications* for the *average user*," I think someone said.

GOURAUD'S TWIST adds the appearance of curvature to a faceted object shown opaquely by the Watkins method (described in first article).

Instead of shading each point within a facet with the same color, interpolate between the vertex-colors according to how far down the edges you've gotten. Note that the jagged edges are retained.



"Wire frame" of Old-Fashioned 3D computer graphics



as shaded by Watkins (or other) method



...and as Gouraud makes it looked curved. GOURAUD'S SPECIAL TWIST

Evans and Sutherland Computer Corporation, Salt Lake City, offers the Watkins Box, a real-time display device using the Watkins Method (see nearby) and offering also Gouraud pseudo-curved shading (see p. DM 104). It costs about \$500,000 and attaches to a FDP-10 large computer.  
Note: That first marketing of the Watkins Box became the nucleus of the Evans and Sutherland real-time simulation empire. See pp. DM 112-113.



SPECIAL EQUIPMENT IS NOW BEING BUILT FOR MAKING "REALISTIC" HALFTONE PICTURES BY COMPUTER. THIS ARTICLE COVERS SOME OF THE MORE UNUSUAL HALFTONE HARDWARE SYSTEMS NOW IN EXISTENCE OR BEING PLANNED.

## HARDENING OF THE ARTISTRIES:

# HARD TIMES A' COMIN'

In two previous articles we have summarized some of the important basic techniques in computer halftone—the artificial construction by computer of photographic pictures of 3D scenes, scenes which are represented within the computer as colored or shaded surfaces placed in a coordinate system of three dimensions.

The techniques we have looked at were all intuitively "spatial" in character, having to do with the analysis of sight-lines and relative edge positions, and suited to implementation in computer software. Now we turn to some more advanced and peculiar techniques and equipment intended to make 3D computer halftone faster to use, or more realistic, or easier to work with, or cheaper. These systems represent a coming generation of halftone hardware.

## THE WATKINS BOX

The University of Utah is now building what will be for some time the world's most spectacular interactive computer display, the Watkins Box. This device, interfacing between a computer and a television screen, will carry out the Watkins algorithm (described in the first article of this series) in *real time*: ripping through a predigested list of facet information, the Watkins Box will create on the screen an image of an opaque object which the user can rotate or see manipulated by program.

The Watkins Box can operate in two modes: normal mode, in which the object appears faceted, and Gouraud mode, in which it appears to be curved over (see masks, nearby).

The Gouraud algorithm, developed by a graduate student of that name, is a

ridiculously simple technique which marries perfectly to the Watkins method. Instead of shading the facets uniformly, this technique calculates a shade of grey for each point. In effect the method interpolates the shade of the point from those around it, across facet boundaries. In actual procedure, the Gouraud method shades a point by linear interpolation between two edge-colors: the color of the last edge and the next edge to be encountered on the present scan-line. (These shades are in turn found by linear interpolation between their endpoints.)

It will be noted that Gouraud's method does not curve the edges. But considering its simplicity as a small addition to the Watkins box, that's no great sacrifice.

Naturally, the Watkins Box will not reach the private home for several years; current likely price is in six figures. But that's now.

*Watkins-Gouraud shading has now (1987) reached the Amiga in a program called "Caligari."*

## PRA'S WORLD-VIEW

Roger Boyell, of Pennsylvania Research Associates, Philadelphia, likes to refer to the company's main interest as "modelling the physical world." Thus he and his associates have developed systems for cartography, landscape modelling, pipe design, and simulation of complex radar systems.

A radar simulator they are putting together for the Navy will show the results of any possible radar system moving over any possible terrain. A pilot or navigator trainee, in a simulated cockpit, will see the mission's changing radar picture as he changes the plane's course or the radar's tuning. The radar picture, appearing on a screen and changing in real time, will look just the way the radar would look on a real mission—flying in perspective among mountains or valleys, high or low, at any bearing and speed, and viewed through any type of radar.

Boyell's approach is to treat each component of the pictorial/radar simulation as a separate problem, to be handled in different ways, and blended in a final buffer, a core memory which is read out to television. Separate mechanisms supply components of shadow, specular reflection, coloration and randomizing effects. The core buffer continuously refreshes the scanned CRT display.

Boyell has put the same techniques to



work making simulated halftone pictures of the moon (see nearby). Both the radar and moon systems use the same type of halftone image synthesis, even though superficially they seem quite different. But radar is radiation, just like light, and Boyell's techniques of three-dimensional modelling and search apply equally well to depiction by reflected visible light—i.e., halftone images.

An outfit called HUMRRO, in Washington, says it has a real-time interactive halftone that will knock several people out of the ballpark—especially the GE hardware and the Evans and Sutherland Watkins Box (earlier).

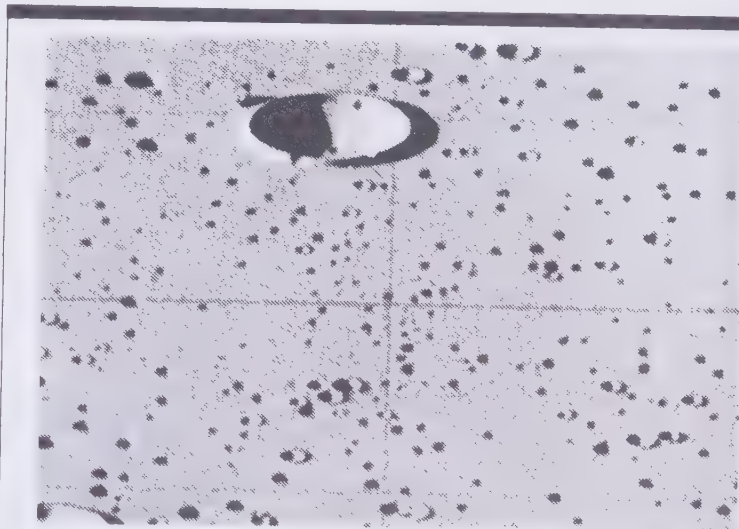
The HUMRRO system is intended to go out to color screens (modified Sony Trinitrons) with shaded polygon halftone, offering pseudo-curved shading like Gouraud's (see earlier).

The techniques were worked out by Ron Swallow, and they're not telling about how they work. It is claimed, however, that their real-time picture generator handles scenes with 16,000 edges, and that this will cost \$150,000 and service 16 (or was it 64) user terminals *simultaneously*.

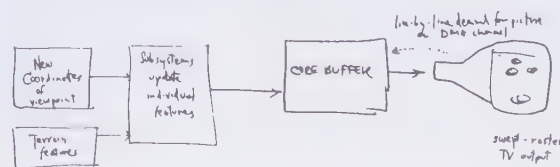
It may have been a bad phone connection, or this may be what they're really claiming. Obviously it'll be really great if it turns out to be real.

Evidently they have in mind the use of such high-performance scopes for teaching, allowing students to explore intricate three-dimensional scenes or objects. Terrific.

(Note: compare the claim of 16,000 edges on a \$150,000 system with the



**BOYELL'S TERRARIUM** fills a fast core-memory buffer with a TV image constantly being read out (much like the Knowlton-Schwartz setup: see p. DM 55, top Schwartz picture) and changes individual features one-at-a-time to match a changing view.







#### "UNREAL ESTATE"

For relaxation, Ron works on the "dream house" he keeps inside the system.

2000 (?) edges allowed by the old NASA system built by GE, or the Watkins Box—I don't know how many edges—at \$500,000 from Evans and Sutherland.)

The halftone system of HUMRRO, rumored above, is real. Clever indeed: It divides the halftone problem into two parts, one the original picturing of the scene, the other its presentation in the terminal. That means that their system permits one central-image generator to send out pictures to as many terminals as desired. Unlike the Watkins Box (see p. DM 110), whose half-million-dollar opulence can be poured only on a single user at once, in this system the central resource can be distributed among various users, with each one's picture changed intermittently, or poured on a single user for full animation. Currently it runs in FORTRAN, transmitting encoded pictures to the unusual terminals required (built around Trinitrons). But a special central processor is foreseen.

The system is called CHARGE, and Ron Swallow, its developer, is indeed a hard charger. (Software: Bill Underhill and Roger Gunwaldsen.) Swallow's game isn't movies or engineering graphics; he wants to CHARGE to compete head-to-head with PLATO (see pp. DM 93-95).

And at the prices he's talking about—\$5000 per terminal and \$150,000 for the central processor—who knows?

#### THE SHAPE OF THINGS TO COME

If these systems sound far-fetched, or only for theoretical investigation, consider this: the Air Force is now letting contracts for an advanced flight-training simulator that is a small boy's dream. To be situated in Dry Lake, Arizona, the simulator will have the most realistic cockpits ever built: the entire mockup will turn and tilt in response to the user, and the seats will even swell and deflate, to simulate acceleration and weightlessness. The cockpits alone, without the visual display screens, will cost ten million dollars each.

But the visual system—ah. The pilot-user will look out into an artificial world, among whose mountains and meadows and clouds he will fly in real time. Six CRTs, arranged as parts of a dodecahedron in an entire visual surround, will show him the changing terrain and flying environment. Each of these CRTs will be driven by a real-time perspective halftone simulator, with all displays spliced together and driven by a master simulator responding to his actions. Who will build them is not yet decided; they could be Warnock or GE boxes.

The sheer joy of such a system will be hard to beat. But no doubt others will be on the way—perhaps at the amusement-park level.

The multi-million dollar simulators are mostly used by the military and civilian pilot training. (This gives pilots safe and realistic experience of numerous crisis situations, and the chance to have a few crashes, harmlessly. It also leads to amusing anomalies. A friend of mine flew underground in a Boeing simulator. Says it was very exciting.)

Mercedes-Benz also has a multiple-projector simulation table big enough to hold a Mercedes truck; they pre-simulate all kinds of driving conditions—and driving feels—there, as well as replaying accidents that are of engineering concern.

Real-time supervisory simulators were during the seventies practically the exclusive playground of the Evans and Sutherland computer company. In the eighties, though, someone put three-dimensional

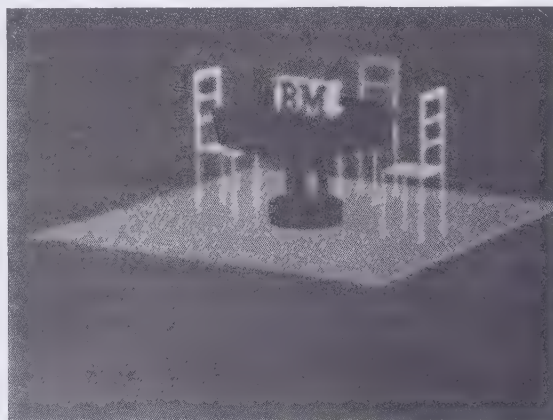
#### AIR FORCE SUPERTOY



The new pilot trainer will not only swing and dip in response to the controls; on six giant CRTs, with optics in front that focus the eye on infinity and connected at the seams, the pilot will see a responding perspective simulation of the world he is flying through, planes he is dogfighting with, and who knows—witches? Superman?

#### Guess Who's Coming To Dinner

**IBM is sponsoring a \$100,000 CHARGE installation at the University of Waterloo, in Canada.**



rotation on a chip-set—called “the geometry engine”—and that is the heart of the IRIS, a very fine simulator/workstation from Silicon Graphics, Inc. The IRIS runs UNIX on an Ethernet, and is widely used throughout the image synthesis industry; the IRIS is, as I said on p. DM 23, the prototype workstation of the future. \$50,000 now, that price goes down rapidly, and the same thing ought to be at personal-computer prices (the IRIS is a personal computer) sooner than most people think.

By 1995—perhaps even 1990—high-power 3D zooming simulation equipment will be on a couple of **cheap** chips, ready to go into the home and school. **Completely redesigning education for this** will be a priority item for the agenda of humankind.

But that brings up the question of how we are to tie together the worlds, objects and characters created by different artists, scholars and research teams. It seems to me there should be a way of **publishing them for shared use**—so that we can combine the different structures so carefully worked out by different people for different purposes. Walk down the street of ancient Rome! Fly through the braincell!

This is one of the goals of Project Xanadu. See p. DM 141.

## How About One Computer Per Pixel?

### THE PIXAR

An immense project at Lucasfilms (under Ed Catmull) was the Pixar, originally billed as a super animation camera for matting parts of films together, as Lucas does so much—you know, space ships and monsters and all.

Now the mighty Pixar has been spun off into a separate company and a product all its own. Very expensive but very powerful, it does operations on every pixel of a frame at once, or moves various “filtering” operations across the image at great speed.

The machine is already proven. It is being sold as part of packages from Symbolics, Inc. and Alias, among others. And it has already been used to discover new medical imaging techniques (see p. DM 82 and pictures p. DM 114).

There's a piece on the Pixar in the December 1986 Scientific American (Dewdney's column).

### “RENDERING ENGINES”

Now that lots of graphics work is done on local-area networks, there are boxes available that you can specially delegate to render the shaded final image. These are sometimes called “rendering engines.”

## Both Sides Now

**“Clouds are a critical element in air-to-air combat. Yet, for over two decades, computer image generation (CIG) has been used to provide visual displays for flight simulation without any ability to simulate clouds with any degree of realism. Clouds are important in the simulation of intelligent weapon systems which seek and identify aerial targets in cluttered backgrounds...”**

—Geoffrey Y. Gardener “Visual Simulation of Clouds” SIGGRAPH '85 Proceedings

## Silicon Graphics

**used just to mean love letters in the sand. Now it's the company in Mountain View, CA, that makes the hottest workstation hardware, with full 3D rotation of opaque-seeming objects and scenes. (See p. DM 24.)**

**HEY! They offer a FREE VIDEO TAPE! (Sorry, only to Qualified Potential Customers. But get your story straight and give it a shot.) (800) 556-6661. California: (800) 824-2388; Canada: (416) 365-7444. VHS only.**

## Frame Buffers In 3D

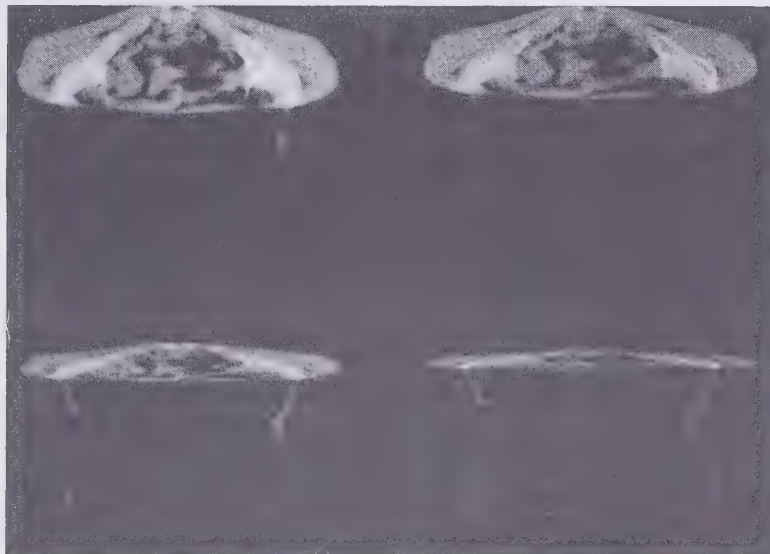
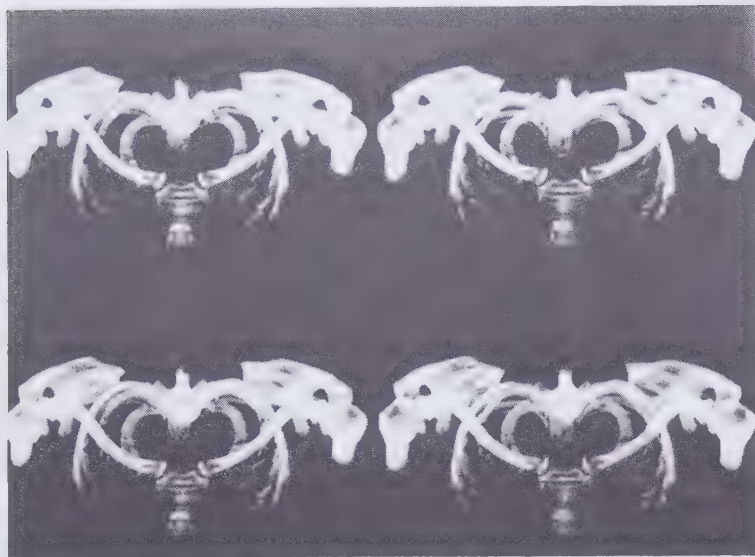
(continued from p. DM 93)

**Z-buffer or depth buffer.** This is a frame buffer for artists working with image synthesis of 3-dimensional scenes. The depth buffer holds not just a color for each pixel, but extra bits. These are used to hold a number representing how far away that pixel is from the camera, in the imaginary scene. When your program adds the image of a 3-dimensional object to what is already in the buffer, it first compares how near each pixel is that is already in the buffer, and then only replaces the pixel if it is **farther away** than the one from the image you're currently adding to the buffer. Thus, after the image of each separate object has been put in the buffer, you end up with a picture on the screen where each part correctly marks or blocks whatever it should. (Note: frame buffers are often measured in bit planes. “Eight bit planes” means eight bits per pixel.)

Renaissance Graphics (Boulder, CO) offers a Z-buffer board-set for the PC (also includes Gouraud shading, see p. DM 111).



*They've discovered that the Pixar can show living tissue and bone in new ways. These photo-like flesh and bone reconstructions are from computer tomographic data on living patients. Created by Pixar. Flesh data courtesy of Phillips Medical Systems, Inc.; bone data courtesy of Dr. Elliot Fishman, Johns Hopkins University. © 1987 Pixar. All Rights Reserved.*



*De Pixies of Pixar are striving to DePict everything you could want moviewise. Here two puppetoon-like characters, Andre the Android and Wally Bee (somehow inspired by the film "My Dinner with Andre"), are seen in a lovely synthesized forest with Maxfield-Parrish-like dappled shade. Note the motion blur, the remarkable trees, and the lighting. © 1986 Pixar. All Rights Reserved.*



*STATE OF THE ART OF IMAGE RENDERING, FROM PIXAR: this unicycle was built from 2700 bicubic surface patches, with reflection maps, texture maps, multiple light sources and anti-aliasing. Crunched in 90 seconds. Created by Pixar. © 1987 Pixar.*

# COMPUTER IMAGE'S MAD WHIRL

SO FAR WE HAVE SUMMARIZED AND DISTINGUISHED AMONG THE MAJOR TECHNIQUES FOR COMPUTER SYNTHESIS OF IMAGES FROM DIGITALLY STORED REPRESENTATIONS OF SCENES. WE NOW TAKE THE WRAPS FROM A DIFFERENT BUT RELATED SET OF TECHNIQUES—THE SYSTEMS OF COMPUTER IMAGE CORPORATION.

Lee Harrison III got the idea for what is now Computer Image Corporation in 1959. Already having an art degree, he went on for a degree in electrical engineering, and through long lean years put together the technical basics around which CI's systems are now built. Computer Image Corporation is now a going concern, and output from their systems, especially Scanimate, is now widely visible on television.

Computer Image Corporation seems to be the first firm to be commercially successful in the halftone field. Whether they should be included with the others is arguable, however. Their systems are not widely understood, and the relation of these systems to the other systems and programs described in these articles is

problematical. Among the few who understand their techniques, some argue that they do not synthesize images at all, but rather twist pre-existing pictures with a sort of Moog synthesizer, and that their analog techniques are really just compound oscillators rather than true computing. I think that this view is wrong, at least as regards their most ambitious system, and that CI's techniques deserve review. All the world is not digital. CI systems *do* fill up areas with grey-scale (and other) pictures, and their systems involve three-dimensional coordinates, occultation and coloration; thus I think it appropriate to discuss them here.

The following discussion is the first, I believe, to lift the veil of secrecy that has hitherto confounded observers of this company's work. In the light of the extreme sophistication with which they have pursued extremely strange techniques, they should benefit from the wider understanding. (Note that this material, which has been assembled from various sources and careful TV watching, is partly conjectural.)

Computer Image's systems represent an apparently unpromising approach brilliantly followed through.

All of CI's systems are a strange combination of closed-circuit TV and analog components out of a music synthesizer: oscillators, potentiometers, interconnection networks. The basic mechanisms are the same for all, but they are carried to different logical extremes, with differing accoutrements, in the four systems. They all seem to be based on the extraordinary Animac II, not yet implemented; it would seem that for business reasons the company decided to raise money promoting simpler systems, so its bread and butter now consists of two less ambitious systems, Scanimate and Animac I; both of which might be puzzling if not recognized as parts of a more elegant whole. It would seem they were designed backwards as spinoffs from Animac II, as was CAESAR, their more recent 2D system.

The extraordinary ramifications and varieties of this system, with all its electronic add-on and composite methods, stagger the most jaded technical imagination.

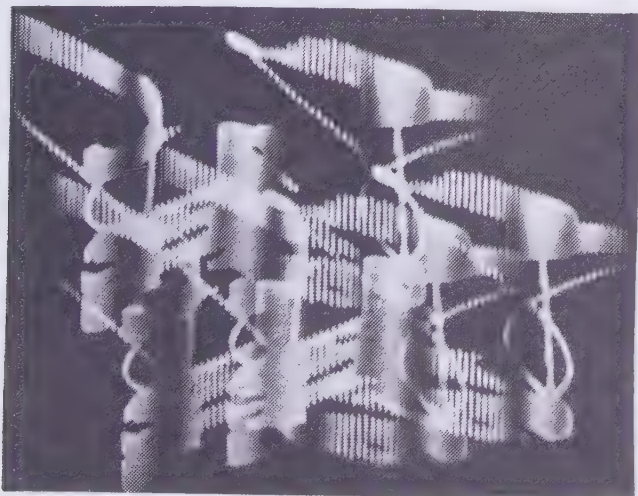
At the heart of the CI systems is the principle of filling areas of a CRT screen with an oscillating trace. This is a principle common to both Lissajous figures and television; but Computer Image has elaborated it peculiarly. By variations they paint twisted television images, wiggle sections of superimposed drawings, create moving filigree effects, and hope to animate whole groups of opaque electronic puppets in 3-space.

Consider an oscillating trace on an oscilloscope. This is a two-dimensional oscillation, having two signals,  $x$  and  $y$ . But a three-dimensional oscillation is also possible; any third signal,  $z$ , can be interpreted as a third dimension, meaning that a "point of light" is whirling out some pattern in a three-dimensional space—an oscillotank, so to speak. Let us call this point moving in three dimensions a "space trace."

Now to view this trace we need to cut it down to two dimensions. By ignoring one of the traces we can view the oscillotank in certain fixed ways; but by creating a "view calculator," a box performing certain perspective transformations on the three signals of the space trace, we may obtain a view of the oscillotank from a movable vantage point. This is an  $x$ - $y$  view which we may put on an ordinary oscilloscope.

Let us now add one more signal,  $b$  (for brightness). This is the brightness signal familiar in television.

Brightness of the spot is thus independent of the movement of the space trace. For example, the space trace could describe a helical path, a sort of tornado motion, and we could time its spinning to phase with a TV signal. If we now brighten the space trace only with the brightness signal of a TV pickup, we now will see (in our view of the oscillotank) what would look like a TV picture curled around itself in space.



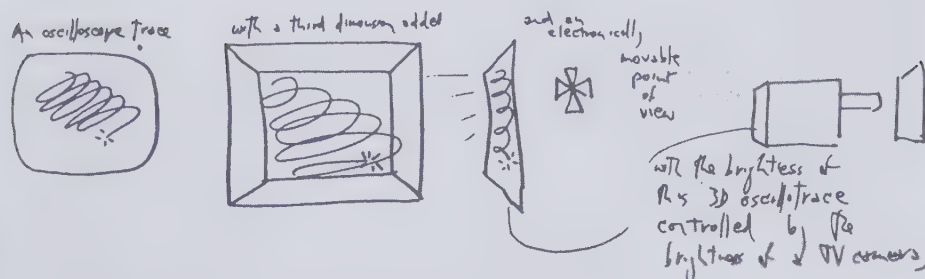
Scanimate's twirl, by now familiar to most TV watchers. Scanimate is extensively used on "The Electric Company."



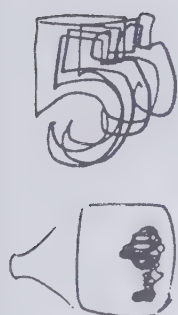
CAESAR System. Characters are made to move jaws and lips by jointing technique similar to Animac II (next page), but in such a way as to matte over drawn artwork—meantime wiggling other drawn artwork through scan manipulation.



## The Whirling Image Of Computer Image Corporation ▼

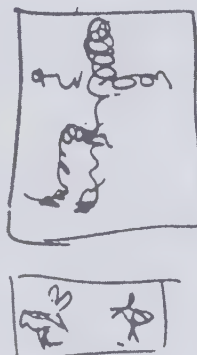


gives us a window into a peculiar sort of world: a world in which luminous shapes can undulate and spin on invisible spindles (Scanimate), or wiggle as separate bones (CAESAR.)



Tubelike shapes may be rotated and shaped in 3D (Animac), and puppets may eventually be rolled like cigarettes (Animac II), which may then be painted from a TV pickup on the side nearest the viewpoint.

By using a storage tube and spinning the trace close together, like cotton candy, and cutting off the painting signal while the trace is within the area already filled, we get electronic masking: which blends animated drawings in 2D (CAESAR) and may eventually manage shadows and occultation masking among 3D puppets (Animac II).



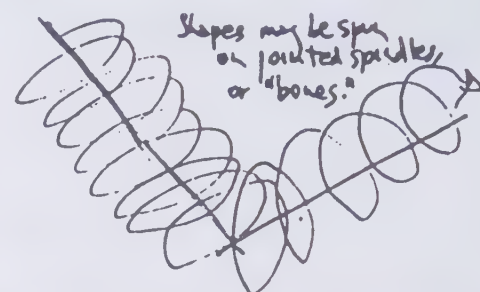
## Shaping Method ●

Lissajous and zigzag features are rapidly spun in three dimensions—that is, varying voltages x, y and z. The resulting “tubes” and “curtains” are then viewed by perspective calculation. The circuitry permits these shapes to flex at joints, wave, and go through other changes.

IN SCANIMATE: zigzag and curling shapes define a moving scroll on which an image is painted.

IN CAESAR: curling shapes are treated 2-dimensionally, as blocking controls for artwork.

IN ANIMAC II: puppets will be sculpted much like rolling a cigarette.

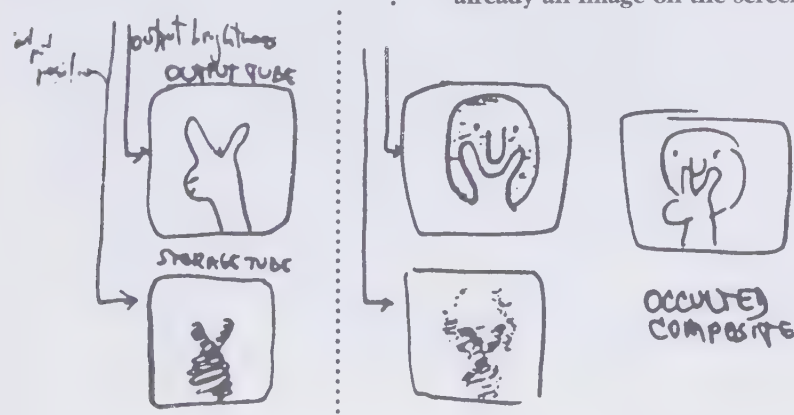


## Blocking Method

While showing nearer part—spin out same part on storage scope; continuously test potential of screen;

—AND LATER, IN THE SAME FRAME:

cut off the brightness of the output signal, wherever there is already an image on the screen.



The only picture I've been able to find that relates to the 3D sculpturing of Animac II is this frame, blown up from a short 16mm sequence. The figure is sculptured from oscillations in three variables, modulated to represent this figure of thirteen sections or “bones.” Head and torso are clearly visible in the film; the figure is seen to spin as if in an ejection seat.

• The different CI systems are built around this effect.

Output from all these signals is ordinarily picked up by another vidicon, which stabilizes it by converting it into conventional television imagery.

A last CI technique, technically minor but remarkable in effect, permits this blocking and shadowing among separate objects. This is the use of a storage CRT on which every frame is painted (from the viewpoint or from the light source). The picture is painted on the storage CRT, nearest things first; and the return signal from the screen tells whether the space trace is crossing an area already painted during the frame. The tube's output signal then effectively constitutes a silhouette. This clue indicates that the space trace should not be visible; and hence is used to cut off brightness while the trace is within the already-filled area. This gates between two desired objects or pictures, foreground and background. If operated from the point of view of the light, it gates shadow: the signal is used to control the relative brightness of the shadowed and unshadowed features of a puppet in 3-space.

A fascinating variety of embellishments has been put into these systems by CI's ingenious engineers. Coloration of the final video signal is added by gating color levels under control of the brightness signal, permitting pictures with several grey-levels to be transformed to up to four rainbow hues. Separate shapes described by the space trace may be independently moved and jointed at the same time: Harrison pointedly calls such separate shapes "bones." Darkening at the backside of a spun shape, or brightening at edges of a painted portion, and brightening in proportion to curl, are all strange capabilities of this machine. Lip-synchronized mouthlike motion can be imparted to any part of the shape spun by the space trace (whether or not a mouth is painted on it), by an audio detector feeding directly to the circuitry from a live mike. And the limbs of CI's ghostly figures can be made to swing by connection

of sensors to the animators themselves—in a living pantograph.

SCANIMATE is a popular device now widely used (at CI's studios) for the making of TV commercials and station-break emblems. This is their simplest system, used for the conversion and discombobulation of flat artwork. In SCANIMATE, the space trace is controlled by hand-operated potentiometers. Two separate oscillator settings are available, so that the space trace can have two separate oscillation patterns, spinning out two entirely different virtual shapes in 3-space. A hand-throttle eases from one oscillator setting to the other. This permits an image to be moved, shrunk or enlarged, or flipped; to go from whirling around to a sort of hula; and many more effects. The picture painted on it may be seen to roll on invisible spindles, bloom into fountains, or undulate as pennants—all by modulating the brightness of the flying spot as it traces its unseen shape. This shape, in turn, can move between its two forms under control of the throttle.

Animac I (usually called Animac) provides greater flexibility in controlling the space trace. The system's oscillations are controlled by an input vidicon, which artists may quickly modify with pastel chalk at the pickup. Ghostly tubular lettering, swarming pendulum-patterns and jiggling filigrees are among the possible doodles.

CAESAR, their newest system, is oriented toward Yogi Bear-type animation. The artist's cartoons are automatically superimposed on a background or each other. They may be moved, and made to wiggle under real-time control by the user.

But it is to Animac II that these curiosities lead. What Harrison calls the "Snow White Capability" of Animac II will permit the sculpture of full humanoid puppets, with perhaps thirty articulated "bones," opaque to one another and casting shadows, colored, moving and talking.

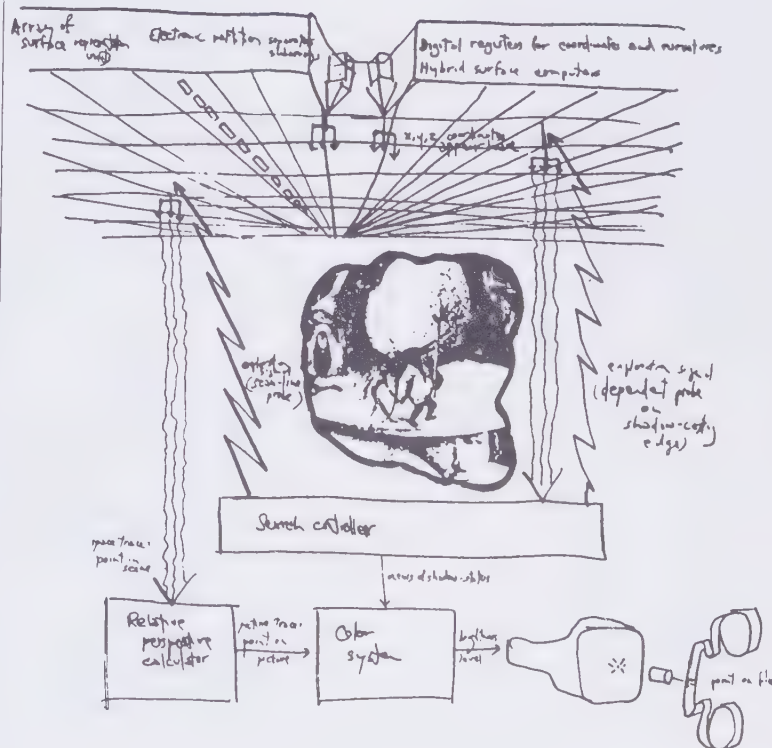
# NELSON'S FANTASM™

I don't expect you to believe this, but the system I call Fantasm™ is intended to make pictures that pass the Turing-test: you won't be able to tell them from real photographs. Fantasm is intended to allow the user to make realistic, Hieronymus Bosch-like photographs and movies, with real-looking people (and scenery, imaginary characters, monsters, etc.) in scenes of arbitrary complexity. Fantasm I originally conceived as a method of making realistic photographs and movies, not knowing at the time that this was impossible, but feeling it could be done somehow if the problem were broken down sufficiently. At times it was not clear which of us would be broken

down first, I or it.

The general goal was to make a system that could do realistic movies without scenery or actors, and make pictures indistinguishable from real photographs of real scenery and actors. ("What do you mean, indistinguishable from photographs?" people keep asking. What do they mean what do I mean?) The surfaces are to be put in by "sculptors," animated by "puppeteers," and photographed by a "director." The objective is for moviemaking to be under the utter imaginative control of the creative user.

I am indebted to Prof. Charles Strauss for the formalization of my smoothing-function. ☹



A LOT OF BOSCH?



A scene of arbitrary curvature and topology is represented in a system of holding registers; the surface is presented (through D-to-A converters and an array parallel function generator) to interrogating circuitry which steers an inquiring signal around the represented surfaces. Operation is empirical. Array has partitioning logic allowing simultaneous queries of various sub-surfaces. Feedback steering circuitry allows multiple loops through array. Steering signal and returned surface parameter are analog and continuous. List techniques manage shadow and visibility 'umbrellas' (surfaces of occulted volumes or umbras).

The Fantasm Scene Machine™, the representation and search array, is one chip repeated in a carpet. Large-scale integration permits the required digital storage of about 500 bits per surface section plus analog circuitry and switching logic.

The system could come in a number of different versions. One of these involves a large array of LSI computing modules (the checkerboard Scene Machine) to be guided by special hardware under an unusual monitor running on a general-purpose computer. The checkerboard Scene Machine holds a great spread of surface data. It is a logical curiosity, an array that replies as a unit, ignoring cell boundaries, to electrical explorations of the shapes represented in it. The resulting trace makes various 3-space explorations on the faces, mountains or automobiles spread-eagled in it. Think of its trace as a radio-controlled firefly skating over a bumpy checkerboard. Using this ma-

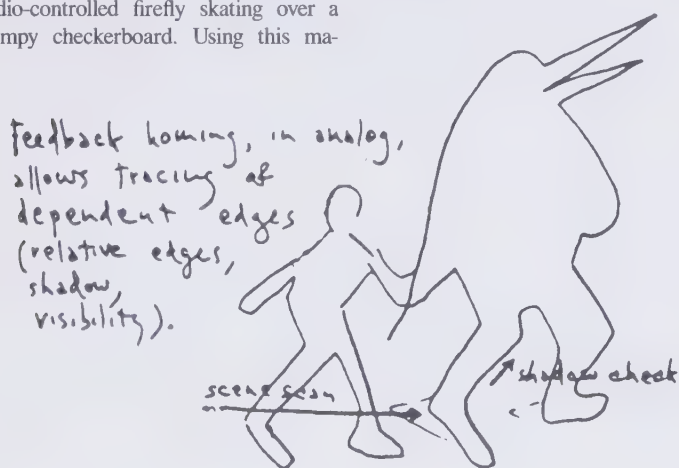
chine, and various cat's-cradle list structures based on the geometry of light around odd volumes of occultation, the problem of halftone analysis of arbitrary shapes is solved by brute force rather than analytically. A variety of other processes have also been defined in the system for other types of graphic application.

As far as I have been able to learn, Fantasm is the most baroque computer graphic system anyone has proposed. It is not intended to operate in real time, but rather take as long as it needs, or as long as the user wants to pay for, to fill in complex visual details, shadow, reflections, curlicues, leaves, hair, etc. It is best suited to the production in Panavision of Busby Berkeley musicals, or "The Lord of the Rings" with realistic wraiths and interspecies battles. But it may well cost too much to use for that. Indeed, its economics seem to improve in low-budget settings like videotape, although there its output bandwidth will flower unseen. But the Scene Machine should also be useful for more mundane applications, such as contour mapping, automobile design, advertising photography and medical illustration.

#### WHERE ARE THEY NOW?

*I had to abandon this project for lack of backing. I believe parts of it (still unpublished) have validity even now. Other parts are rather humorous.*

*(The old pieces written for Computer Decisions end here.)*



## SOFT PARTS ARE THE HARD PARTS

*Today, 3D structures are built out of polygons and many types of "surface patch"—curved shell-like surface sections that can be fitted together into larger structures—Coons patches, Bezier patches, quadric patches, Steiner patches and others. With these surfaces it's fairly easy to sculpt machinery and simple objects; but it's hard to do human-like and animal-like shapes with reasonable animation. In faces and soft curvatures, and the animation of creature-like objects, we're still way behind.*

*And that's why the pictures and animations done by Yoichiro Kawaguchi in Japan are so especially astonishing—because they simulate organic shapes and the look of cells and snails and pond life. (See bibliography.) And that's why Philippe Bergeron's film "Tony de Peltrie," done in Canada a few years ago, was such a hit. The title character is a synthesized, animated piano player. Animating 3D synthesized fingers? Now, that's real digital computing.*

## JAGGIES AND JIGGIES

*Simple ways of doing animated image synthesis such as ray-tracing, have a few problems. One is that diagonal straight lines don't come out straight (jaggies), and that there is often jerkiness between successive frames (jiggies). (These prob-*

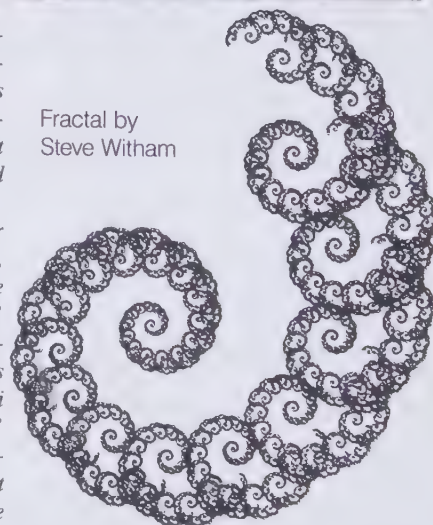
*lems are technically called spatial aliasing and temporal aliasing.) So part of what's done in shaded output is various corrections to get rid of these (anti-aliasing). This puts in just the right amount of blur to make things look right.*

## FRACTALS

*are new and hard to understand. Discovered and named by Benoit Mandelbrot, a French mathematician, fractals are a class of curved line and curved surface that have no exact location, but just keep readjusting their position more and more the closer you look.*

*Fractals are used in computer graphics to depict mountains, coastlines, smoke, and much more. (The incredible Genesis Bomb sequence of "Star Trek II" and "III" was done fractally; Loren Carpenter did a fractal system that generates a whole planetary landscape for the Atari 800 video game, "Rescue on Fractalus," also a Lucasfilm production.) Mathematically, fractals are very complex; but as implemented, a fractal is usually a line or surface whose parts have been pushed more and more out of line by a recursive process.*

Fractal by  
Steve Witham



**FRACTAL:** a geometric pattern which, when viewed at any level of detail, is similar to the whole.



# THE WORKING WORLD OF COMPUTER IMAGE SYNTHESIS

*In an image synthesis/animation house, what do you work with?*

*Curved Patches (and the problem of Getting Them Right)*

*Collections of Shapes*

*Tree Structures of How the Shapes Fit Together*

*Paths and Choreography of Objects (also Tree-Structured)*

*Frame-Buffers in which to Assemble Individual Frames*

*The Process of Interpolating Animation Between Frames*

*"Final Rendering"*

*"Final Touches"—Twinkles and Streaks*

**"Y**ou have  
to really  
want to do  
animation  
real bad in  
order to do  
animation."  
-Jim Blinn

# BIBLIOGRAPHY

## FOR ALL IMAGE SYNTHESIS ARTICLES

Annabel Jankel and Rocky Morton. *Creative Computer Graphics*, Cambridge University Press, 1984.

Isaac Victor Kerlow and Judson Rosebush, *Computer Graphics for Designers and Artists*, Van Nostrand Reinhold, 1986.

Robert Rivlin, *The Algorithmic Image: Graphic Visions of the Computer Age*, Microsoft Press, 1986.

Nadia Magnenat-Thalmann and David Thalmann, *Computer Animation: Theory and Practice*, Springer-Verlag, 1985.

Yoichiro Kawaguchi, *Growth Morphogenesis*, JICC Publishing Inc., (5-5-5, Koujimachi Chiyoda-ku, Tokyo, Japan) 1985. ISBN 4-88063-093-4-C0070.

*An incredible, fantastic, gorgeous book of glistening, semi-transparent synthesized pictures of organic-seeming objects and creatures.*

H. O. Peitgen and P. H. Richter, *The Beauty of Fractals: Images of Complex Dynamical Systems*, Springer-Verlag, 1986.

David Rogers and Alan J. Adams, *Mathematical Elements for Computer Graphics*, McGraw, 1976. *This is the one Jim Blinn recommends.*

Computer Pictures magazine, \$24/year from Electronic Pictures Corporation, 2 Village Square West, Clifton NJ 07011. At this writing, *Computer Pictures* appears to be the industry magazine that focusses most closely on 3D image synthesis.

SIGGRAPH Proceedings, any year.

*The following subjects are slightly irrelevant to the main thrust of this book, which concerns new and old media and how to show things. But they certainly are*

*computer dreams, and they wrap around to computer graphics and hypermedia in more ways than one.*

# TIME OUT FOR THREE COMPUTER DREAMS: "AI" (Artificial Intelligence); "IR" (Information Retrieval); "CAI" (Computer-Assisted Instruction)

## THE THREE DREAMS:

IT'S TIME FOR AWE TO BE REPLACED WITH THE CRITICAL EYE

These are three topics of great importance; of importance, unfortunately, less for what they have actually accomplished than for the degree to which they have confused and intimidated people who want to understand what's going on. Merely to mention them can be one-upmanship. All three titles mean so much, so many different specific things, as to mean almost nothing when lumped together as a whole. All three have developed a web of intricate technical facts (and sometimes theorems), but the applicability of these elegant findings is in all three cases a matter open to considerable scrutiny.

Since each of these fields has developed a considerable body of technical doctrine, the reader might well ask: why aren't they on the other side of the book, the computer side? The answer is that they are computerman's dreams, dreams of considerable intricacy and persuasiveness, and we are not considering the technicalities here anyway. As on the other side, the problem is to help you distinguish apples from oranges and which way is up. For more go elsewhere, but I hope this orientation will make sorting things out quicker for you.

These three terms—"artificial intelligence," "information retrieval," "com-

puter-assisted instruction"—have a number of things in common. First, the names are so portentous and formidable. Second, if you read or hear anything in these fields, chances are it will have an air of unfathomable technicality. Both strange technicalism and deep mathematics may combine to give you a sense that you can't understand any of it. This is wrong. The fact that there are obscure and Deep Teachings in each has no bearing on the general comprehensibility of what they are about. More importantly, the question of *how applicable* all the things these people have been doing is going to be a question of considerable importance, especially when some of these people want to take something over. Don't get snowed.

Each of these fascinating terms is actually a roof over a veritable zoo of different researchers, often of the most eccentric and interesting sort, each generally with his own dream of how his own research will be *the breakthrough* for humanity, or for something. It would take a Lemuel Gulliver to show you the colorfulness and fascination of these fields; again, we just scratch the surface here.

Another interesting thing these three fields have in common: the frequent use of a classical computerman's putdown on anybody who dares question whether their super-ultimate goals can ever be achieved.

The line is, "WE DON'T KNOW HOW TO DO THAT YET."

If somebody pulls it on you, the reply is simply, "How do you know you ever will?"



## THE JAPANESE FIFTH GENERATION PROJECT

A certain chill went across the U.S. technological community when Japan announced their mighty Fifth Generation Computer Project, which would supposedly develop real artificial intelligence (yuk yuk) running the Prolog language (see p. 62) on big machines.

It hasn't happened yet, Prolog's difficulties are becoming better understood, and the worry has eased. Another Sputnik scare.

Artificial Intelligence is okay in its place, but I wouldn't want my daughter to marry one.  
—Anon.

## THE GOD BUILDERS!

# ARTIFICIAL INTELLIGENCE...sort of

"Artificial Intelligence" is at once the sexiest and most ominous term in the world. It chills and impresses at the same time. In principle it means the simulation of processes of mind, by any means at all; but it generally turns out to be some form or another of computer simulation (see "Simulation," p. 149). Actually, "artificial intelligence" has generally become an all-inclusive term for systems that amaze, astound, mystify, and do not operate according to principles which can be easily explained. In a way, "artificial intelligence" is an ever-receding frontier: as techniques become well-worked out and understood, their *appearance* of intelligence, to the sophisticated, continually recedes. It's like the ocean: however

much you take out of it, it still stretches on, as limitless as before.

Unfortunately, laymen are so impressed by computers in general that they easily suppose computers can do anything involving information. And public understanding is not fostered by certain types of stupid demonstration. One year I heard from numerous people about how "they'd seen on TV about how computers write TV scripts"—what had actually been shown was a hokey enactment of how the computer could randomly decide whether the Bad Man gets shot or the Good Guy gets shot—both outcomes dutifully enacted by guys in cowboy outfits. Duh.

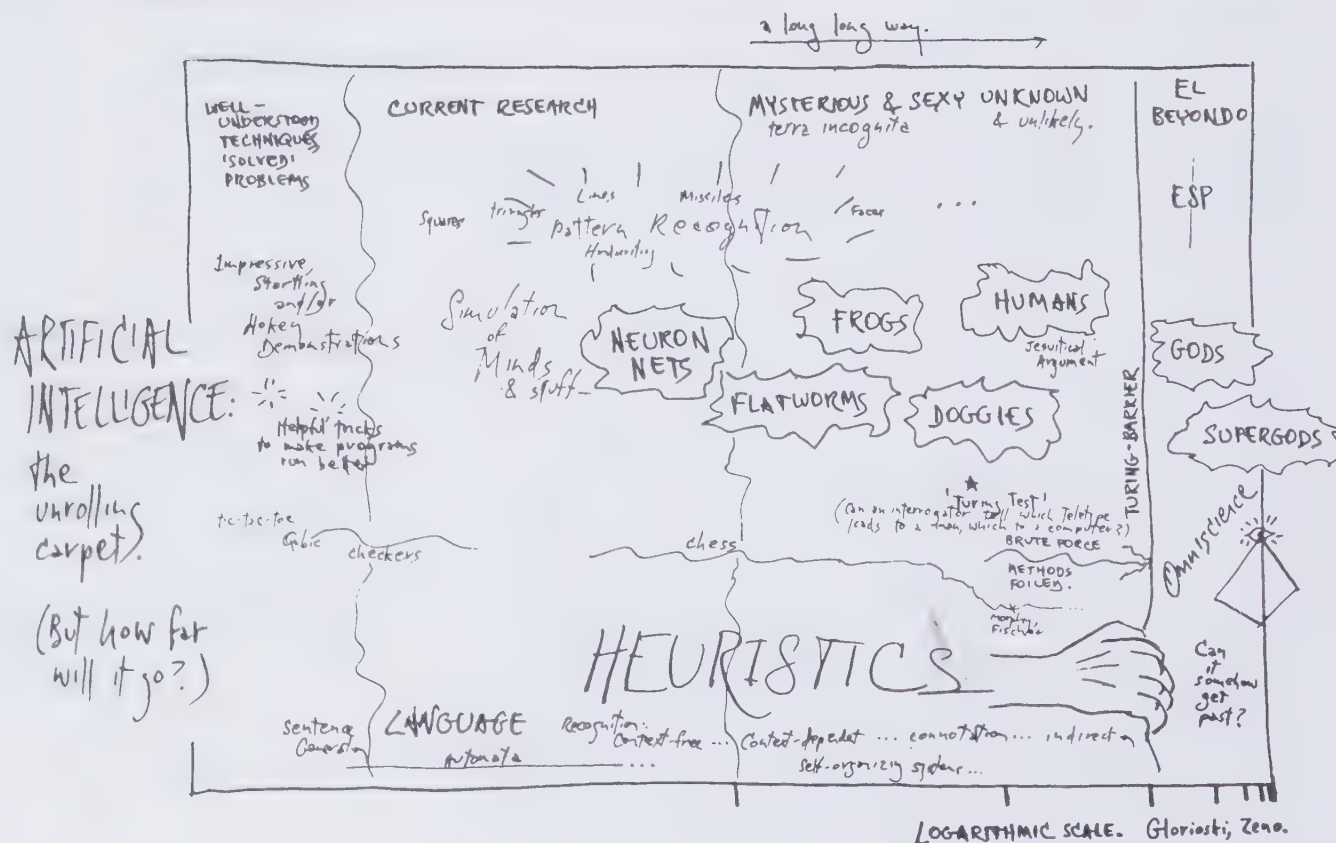
It should be perfectly obvious to anybody who's brushed even slightly with computers, however—for The Brush,

see the other side—that they just don't work like minds. But the analogy hangs around. (Edmund C. Berkeley wrote a book in the forties, I believe, with the misleading title of *Giant Brains, or Machines That Think*. The idea is still around.)

Here's a very simple example, though. Consider a maze drawn on a piece of paper. Just by looking, we cannot simultaneously comprehend all its pathways; we have to poke around on it to figure out the solution. Computers are sort of like that, but more so. While our eyes can take in a simple picture, like a square, at once, the computer program must poke around in its data representation at length to see what we saw at once.

The principle holds true in general. The human mind can do in a flash, all at once (or "in parallel"), many things that must be tediously checked and tried by the highly sequential computer program. And the more we know about computers, the more impressive the human brain becomes. (The seeming cleverness of some simple programs does *not* prove the simplicity of the phenomena being imitated.)

Computers Don't Actually Think.  
You Just Think They Think. (We Think)



Nevertheless, it is interesting to try things with computers that are more like what the mind does; and that is mostly what artificial intelligence is about.

In various cases this has resulted in helpful tricks that turn out to be useful elsewhere in the computer field. In this sense, artificial intelligence is sort of like menthol: a little may improve things here and there. But (in my opinion), that does not mean a whole lot of it would make things better still.

Nevertheless, some artificial-intelligence enthusiasts think there is no limit on what machines can do. They point out, after all, the brain is a machine. But so is the universe, presumably; and we're never going to build one of those, either.

### PATTERN RECOGNITION

This is one of the most active areas in artificial intelligence, perhaps because of Defense Department money. (It might be nice, goes the reasoning, to have guns that could recognize tanks, machines that could look over aerial reconnaissance pictures, radars that could recognize missiles...)

What it boils down to is the study of *clues* and *guessing among alternatives*. In some cases, well-defined clues can be found for recognizing specific things, like parts of pictures (even straight lines cannot be recognized by computer without a complex program) or like handwriting (see nearby). In the worst cases, though, careful study only raises the most horrendous technical problems, and the pursuit of these technical problems is its own field of study (articles have titles like "Sensitivity Parameters in the Adjustment of discriminators," meaning It Sure Is Hard to Draw The Line.)

But in some felicitous cases, researchers actually boil a recognition problem down to a manageable system of clues. For instance, take the problem of written input to computers. (Some people don't like to type and would rather write by hand on special input tablets.) But how can a program recognize the letters? Aha: the answer, kids, is in your text.

The Ledeen Character Recognizer (described in detail in Newman and Sproull, *Principles of Interactive Computer Graphics*, Appendix 8) is a method by which a program can look at a hand-drawn character and try to recognize it. The program extracts a series of "properties" for the character and stores them in


an array. Every character in a given person's block lettering will tend to have certain property scores. But the Ledeen recognizer must still be *trained*, that is, the average property scores of the letters that each individual draws must be put into the system before that individual's lettering can be recognized. Even then it's a question of probability, rather than certainty, that a given character will be recognized.

### HEURISTICS (Pronounced hewRIStics)

If we want to make a computer do what we know perfectly well how to do ourselves, then all we do is write a program.

Aha. But what if we want a computer to do something we do *not* know how to do ourselves?

We must set up its program to browse, and search, and seize on what turns out to work.

This is called heuristics. 

## The Press Releases Go On And On

*"Researchers at MCC and a handful of laboratories are trying to build the prototype for a fifth-generation computer capable of reasoning its way through myriad tasks in the home, at the workplace and on the battlefield."*

—J. H. Doyle, "Making a Computer Get Smart," *Insight*, 24 March '86, p. 63

*Artificial intelligence is what we don't know how to do yet.*  
—Larry Tesler

Dorothy read the card aloud, spelling out the big words with some difficulty; and this is what she read:

SMITH & TINKER'S  
Patent Double-Action, Extra-Responsive,  
Thought-Creating, Perfect-Talking  
**MECHANICAL MAN**  
Fitted with our Special Clock-Work Attachment.  
Thinks, Speaks, Acts, and Does Everything but Live.  
Manufactured only at our Works at Evns, Land of Ev.  
All infringements will be promptly Prosecuted according to Law.

"How queer!" said the yellow hen. "Do you think that is all true, my dear?"

O z m a o f O z

"I don't know," answered Dorothy, who had more to read. "Listen to this, Billina:"

DIRECTIONS FOR USING:  
For THINKING:—Wind the Clock-work Man under his left arm, (marked No. 1.)  
For SPEAKING:—Wind the Clock-work Man under his right arm, (marked No. 2.)  
For WALKING and ACTION:—Wind Clock-work in the middle of his back, (marked No. 3.)  
N. B.—This Mechanism is guaranteed to work perfectly for a thousand years.

"Well, I declare!" gasped the yellow hen, in amazement; "if the copper man can do half of these things he is a very wonderful machine. But I suppose it is all humbug, like so many other patented articles."

"We might wind him up," suggested Dorothy, "and see what he'll do."

## The Turing Machine

is the most classical abstract Automaton. A Turing Machine, named after its discoverer, is a hypothetical device which has an infinite recording tape that it can move back and forth, and the ability to make decisions depending on what's written there.

Turing proceeded to point out that no matter how fast you go step-by-step, you can't ever outrun certain restrictions built into all sequential processes as represented by the Turing Machine. This lays heavy limits on what can ever be done step-by-step by computer. (It means we have to look for non-step-by-step methods, which much of Artificial Intelligence is about.)



What it amounts to basically is techniques for trying things out, checking the results, and continuing to do more and more of what seems to work best.

Or we could phrase it this way: looking for successful *strategies* in whatever area we're dealing with. As a heuristic program tries things out, it keeps various scores of how well it's doing—a sort of self-congratulation—and makes adjustments in favor of what works best.

Thus the Greenblatt Chess Program, mentioned under "Chess," nearby, can "invent" chess strategies and "try them out." What it actually does is test specific patterns of moves for the overall goodness of their results (in terms of the usual positional advantages in chess), and discard the strategies that don't get anywhere. It does this by comparing its "strategies" (possible move patterns) against the records of chess matches which are fed into it.

(If you've read the other side of the book, heuristics may be thought of as a form of operations research (p. 151) carried on by the computer itself.)

In some ways heuristics is the most magical area of artificial intelligence: its results are the most impressive to laymen. But, like so many of the computer magics, it boils down to technicalities which lose the romance to a certain extent.

## NEURAL SIMULATION

An important branch of artificial intelligence is concerned with what bunches of imaginary neurons could do, even neurons that we made up to follow particular rules. This area of study is somewhere between neurology and mathematics; much of it is concerned with the mathematics of imaginary setups, rather than the properties of actual nerve-nets, as studied by psychologists, physiologists and others. (The hypothetical studies, of course, alert researchers to complex configurations and possibilities that may turn out to occur in reality, as well as being interesting for their own sake—and conceivably as useful ways of organizing things to be built.)

However, an earlier myth, that you could simulate neurons till you got a person, is about dead.

## SIMULATION OF THOUGHT-PROCESSES

Nobody talks anymore about simulating artificial brains; there's too much to it, and it involves dirty approximations.

*Actually, that's changed a little recently. There is a new school of thought called "Connectionism" that says maybe neural hookups can do a lot after all. But that is now distinct from "real artificial intelligence"—get the joke?—which is now the many techniques of cognitive sci-*

*ence (whose surface I am trying at least to scratch here).*

However, a cleaner area is in the simulation of thought: creating computer programs that mimic man's mental processes as he dopes through various problems. Trying things out, deducing thoughts from what's already known, following through the consequences of guesses—these can all be done by programs that "try to figure out" answers to problems like The Cannibal and The Missionary, or whatever.

## AUTOMATA

"Automata", as the term is used in this field, is just a fancy word for *imaginary critters*, particularly little things that behave in exact ways. (The Game of Life, see p. 115, is an automaton in this sense.)

## SELF-ORGANIZING SYSTEMS, SELF-REPRODUCING SYSTEMS, AND SO FORTH

These are terms for imaginary objects, having exactly defined mathematical properties, about which various abstract things can be proven that tend to be of interest only to mathematicians.

## SPEECH

### 1. SENTENCE GENERATION

The problem of computers speaking human languages—not to be confused

with computer languages, p. 56 and elsewhere—is incredibly complicated. Just because little human tykes start doing it effortlessly, it is easy to suppose that it's a basically easy problem.

No way.

Only since the mid-fifties has human language begun to be understood. That was when Noam Chomsky discovered the inner structure of human languages: namely, that the long (and complex) sentence constructions of language are built out of certain exact operations. Previous linguists had sought to classify the sentence structures themselves; this led to complexities which Chomsky discovered were unnecessary. It is unnecessary to catalog sentence types themselves if we can simply isolate, instead, the exact processes by which they are generated.

These processes he called *transformations* (a term he borrowed from mathematics). All utterances are created from certain elementary pieces, called *kernels*, which are then chewed by transformations into *surface structures*, the final utterances. Examples of kernels: The man lives in the house, The house is white. Result of combining transformation: The man lives in the white house. Kernel: I go. Result of past-tense transformation: I went.

The most important finding, now, is that the transformations are carried out in *orderly sequences*: any sentence can have more transformations carried out on it,

# Can A Computer Play Chess?

The real question is, can a set of procedures play chess? Because that's what the computer program really does, enact a set of procedures.

And the answer is yes, fairly well.

Now, a chess program is not something you jot down on the back of an envelope one afternoon. It's usually an immense, convoluted thing that people have worked on for years. (Although I vaguely recall that second place in the 1970 inter-computer chess contest was won by a program that occupied only 2000 locations in a 16-bit minicomputer—in other words, a compact and tricky sneaker.)

Now, simple games (like tic-tac-toe and Nim and even Cubic) can be worked out all the way: all alternatives can be examined by the program and the best one found. Not so with chess.

Chess basically involves, because of its very many possibilities, a "combinatorial explosion" of alternatives (see p. 72); that is, to look at "all" the pos-

sibilities of a midgame would take forever (perhaps literally—the Turing problem), and thus means must be found for discarding some possibilities.

The structure of branching possibilities is a tree (see p. 99); so that methods of "pruning" the tree turn out to be crucial.

Basically there are two approaches to the design of chess programs. In one approach, the programmers look for specific threats and opportunities in the data structure representing the board, and try to find good strategies for selecting good moves on the basis of them. This is the approach taken in COKO, the "Cooper-Koz" chess program. The programmers selectively cope with individual problems and strategies as they turn out to be necessary. (This means that it is likely to have specific Achilles' heels; which, of course, the authors of the program keep trying to repair by adding specific corrections.)

A different approach is taken by the Greenblatt chess program. This is basically a big Heuristic program. It "learns" best strategies in chess by "watching" the game. That is, you pour historical chess matches through it, and it tries out strategies—making various tentative rules about what kinds of moves are good, then *scoring these moves according to the results* of making them—as seen in positional advantages that resulted in actually championship play.

Obviously this is a field in itself. You won't get grants for it, but to those who really care about both chess and computers, it's the only thing to be doing.

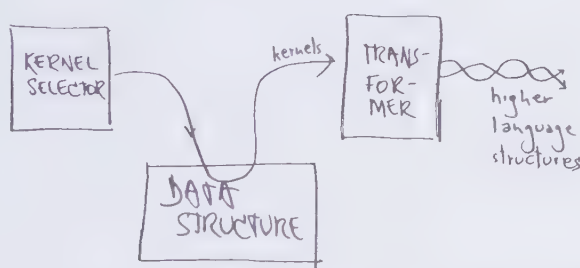
all adhering to the basic rules, resulting in the most complex sentences of any language.

Linguists since then have confirmed Chomsky's conjecture, and proceeded to work out the fundamental transformations of major languages, including English.

Now, one result of all this is that it turns out to be easier to *generate* sentences in a language than to understand them. Why? Because it is comparatively easy to program computers to apply transformations to kernels, BUT very hard to take apart the result. A complex "surface structure" may have numerous possible kernels. Does "Time flies like an arrow" have the same structure as "Susie sings like a bird" or "Fruit flies like an orange?"

Result: to program a computer to generate speech—that is, invent sentences about a data structure and type them out—is comparatively easy, but to have it recognize incoming sentences, and break them up into their kernel meanings, is not.

We may think of a language-generating computer system as follows:



puters, it is sensible guessing. (But see "Heuristics," nearby.)

This means that to create computer systems which will take real sentences apart into their meanings is quite difficult. We can't get into the various strategies here; but most researchers cut the problem down in one way or other.

### 3. SPEECH OUTPUT AS SOUND

It is possible in principle to set up computers to "talk" by converting the language surface structures that their programs come up with into actual sound. See "Audio," p. DM 60.

### 4. SPEECH INPUT TO COMPUTERS BY ACTUAL SOUND

So far we have been talking about the computer's manipulation of language as an *alphabetical coding* or similar representation. To actually *talk at* a computer is another kettle of fish. This means breaking down the sound into phonemes and *then* breaking it into a data structure which can be treated with the rules of grammar—a whole nother difficult step.

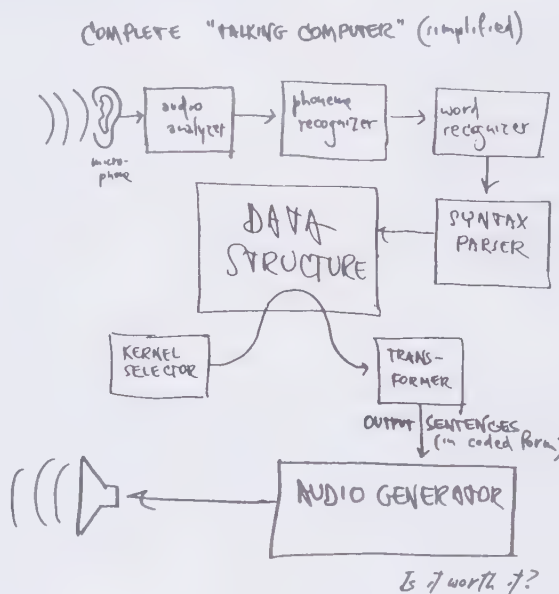
A few attempts have been made to market devices which would recognize limited speech and convert it to symbols to go into the computer. One of them, which supposedly can distinguish among thirty or forty different spoken words, is supposedly still on the market. Specific users have to "train" it to the particulars of their voices.

I repeatedly hear rumors of "dictation machines" which will type what you say to them. If such things exist I have been unable to confirm it.

(Everybody says that of course what we *want* is to be able to communicate with computers by speech. Speaking personally, I certainly don't. Explaining my punctuation to human secretaries is hard enough, let alone trying to tell it to a computer, when it's easy enough to type it in.)

### 5. ALL TOGETHER NOW

The complexity of the problem should now be clear. ☹



## 2. SENTENCE RECOGNITION

Chomsky and others have discovered that sets of transformation rules (or *grammars*, praise be) vary considerably. It is possible to invent languages whose surface structures are easy to take apart, or *parse*; such languages are called *context-free languages*. (Most computer languages, see other side, are of this type.) Unfortunately *natural* languages, like English and French and Navaho, are not context-free. It turns out that the human brain can pick apart language structures because it is so good at making sensible guesses as to what is meant—and if there is one thing hard to program for com-

I have strong hunches about the inner workings of men who get millions of dollars from the Department of Defense and then say in private that really they're going to use it to create a machine so intelligent it can play with their children. (Not to name names or anything.) An obvious question is, do they play with their children? No, they play with computers.

## ONE OF THE FEW GOOD LAYMEN'S COMPUTER JOKES

illustrating also certain problems of Artificial Intelligence.

A very large artificial-intelligence system (goes the story) had been built for the military to help in long-range policy planning; financed by ARPA, with people from MIT, Stanford and so on.

"The system is now ready to answer questions," said the spokesman for the project.

A four-star general bit off the end of a cigar, looked whimsically at his comrades and said—

"Ask the machine this: Will it be Peace or War?"

The clerk-typist (Sp4) translated this into the query language and typed it in.

The machine replied:

YES

"Yes *what*?" bellowed the general.

The operator typed in the query.

Came the answer:

Yes SIR.



## DO WE WANT TALKING SYSTEMS?

I had one quite irritating experience with a “conversational” system, that is, computer program that was supposed to talk back to me. I was supposed to type to it in English and it was supposedly going to type back to me in English. I found the experience thoroughly irritating. My side of the conversation, which I sincerely tried to keep simple, produced repeated apologies and confusion from the program. The guy who'd created the program kept explaining that the program would be *improved*, so that *eventually* it could handle responses like mine. My reaction was, and is, *Who needs it?*

Many people in the computer field seem to think we *want* to be able to talk to computers and have them talk back to us. This is by no means a settled matter.

Talking programs are complicated and require a lot of space in the machine, and (more importantly) require a lot of time by programmers who could achieve

(I think) more in less time by other means. Moreover, talking programs produce an irritating strategic paradox. In dealing with human beings, we know what we're dealing with, and can adjust what we say accordingly; there is *no way to tell*, except by a lot of experimenting, what the principles are inside a particular talking program; so that trying to adjust to it is a strain and an irritation. (Compare: talking to a stranger who may or may not turn out to be your new boss.) Now, some programmers keep saying that *eventually* they'll have it acting just as smart as a real person, so we needn't adjust; but that's ridiculous. We *always* adjust to real people. In other words, the human discomfort and irritation of *psyching the system out* can never be eliminated.

Furthermore, on today's sequential equipment and with feasible budgets, I personally think the likelihood of making programs that are really general talkers is a foolish goal. There are many simpler ways of telling computer systems what

you want to tell them—light pen choice, for example.

Moreover, having to type in whole English phrases can be irritating. (We can't even get into the problem of having the computer pick apart the audio if you *talk* it in.)

This is not to say *understandably restricted* talking systems are bad. If you know and understand the sorts of response the system makes to what kinds of things, then an English-like response is really a clear message. For instance, the JOSS system (the first Quickie language) had an eloquent message:

eh?

which actually meant, *What you have just typed in does not fit the rules of acceptable input for this system*. But it was short, it was quick, it was simple, and it was almost polite.

Similarly, talking systems that use an exact vocabulary, whose limits and abilities are known to the person, are okay.

(Winograd, see Bibliography, has a nice example of telling a computer to stack blocks, where the system knows words like between, on, above, and so on.) Where this is understood by the human, it can be a genuine convenience rather than a spurious one.

(The problem of *rudeness* in computer dialogue has not been much discussed. This is partly because many programmers are not fully aware of it, or, indeed, some are so skilled in certain subtle forms of rudeness they wouldn't even know they weren't acceptable. The result is that certain types of putdown, poke, peremptoriness and importunacy can find their way into computer dialogue all too easily. Or, to put it another way: nobody likes to be talked back to. Cf. Those stupid green THANK YOU lights on automatic toll booths.)

Now, this is *not* to say that research in these areas is wrong, or even that researchers' hopes of some breakthrough in talking-systems is misguided. I am say-

Sometimes All We Need Is a Sympathetic Teletype

## ELIZA WILL SEE YOU NOW

(What the ELIZA program types, we represent as all caps; what the person types is not.)

PLEASE TELL ME YOUR PROBLEM.

I am feeling depressed.

HOW LONG HAVE YOU BEEN FEELING DEPRESSED?

Ever since my mother left.

TELL ME MORE ABOUT YOUR FAMILY.

My father is very strict.

PLEASE GO ON.

Last night I dreamt he spanked me.

WHAT WOULD IT MEAN TO YOU IF HE SPANKED YOU?

Didn't you ever get spanked?

WE WERE DISCUSSING YOU, NOT ME.

You're just a dumb machine anyway.

DO COMPUTERS FRIGHTEN YOU?

(quoted in Winograd, 74-5)

The fascinating thing about the ELIZA program is that it seems so like a real person—that is, a psychoanalyst of the nondirective, Carl Rogers school.

But all it is is a bag of tricks.

It has a lot of canned phrases it can throw out, like PLEASE TELL ME YOUR PROBLEM.

Then it spots certain types of sentences and turns them around. If the person types a sentence beginning with “I am...,” the ELIZA program turns it around with HOW LONG HAVE YOU BEEN...?

And it looks for specific words which indicate a sentence can be turned around for a convincing result. For instance, if the person uses the word “everyone,” the program simply replies, WHO IN PARTICULAR CAN YOU THINK OF?

And certain highly-charged words are signals to throw in a zinger, regardless of what the user may have said. For instance, if the person uses the word “mother,” the machine replies:

TELL ME ABOUT YOUR FAMILY.

Now, do not be fooled. There is no Ghost in the Machine. The program does not “understand” the user. THE PROGRAM IS MERELY ACTING OUT THE BAG OF TRICKS THAT JOE WEIZENBAUM THOUGHT UP. Credit where credit is due: not to The Computer's Omniscience, but to Weizenbaum's cleverness.

(Look at the above sample dialogue and see if you guess what tricks the program was using.)

The thing is, many people refuse to believe that it's a program. Even when the program's tricks are explained.

And even some who understand ELIZA like to call it up from their terminals for companionship, now and then.

## BIBLIOGRAPHY

Terry Winograd, “When Will Computers Understand People?” *Psychology Today* May '74, 73-9.

(Weizenbaum's full article on ELIZA appeared in the *Communications of the ACM* sometime in the mid or late sixties; a flowchart revealed its major tricks.)

This same original ELIZA is now available for personal machines on an album called Les Crane's Golden Oldies, from Electronic Arts.

ing, basically, that talking systems cannot be taken for granted as *the* proper goal in computers to be used by people; that the problems of rudeness, and irritating the human user, are far greater than many of these researchers suppose; and that there may be alternatives to this potentially eternal leprechaun-chasing.

If like the author you are bemused by the great difficulty of getting along with human beings, then the creation of extraneous beings of impenetrable character with vaguely human qualities can only alarm you, and the prospect of these additional crypto-entities which must be fended and placated, clawing at us from their niches at every turn, is both distasteful and alarming.

*(The obsequious talking doors from Hitchhiker's Guide to the Galaxy, as well as the obnoxious Eddie the Shipboard Computer, are fine recent examples of this problem.)*

## FRANKENSTEIN MEETS CYBERCRUD

Fred Brooks, the keynote speaker at the IEEE computer conference in Fall '74, seems to have said that HAL 9000 (the unctuous, traitorous Presence in the movie "2001") was the way computers should be. (*Computer Decisions*, Apr '74, 4.)

I find it hard to believe that anybody could think that. Nevertheless, there are those artificial-intelligence freaks whose view is that the purpose of all this is eventually (a) to create servants that will read our minds and do our bidding, (b) servants who will take things over and will implement human morality, regardless of our bidding (though we humans are too frail to do so—as in Asimov's *I, Robot*); or even (c) create masters who will take everything over and run everything according to *their* own principles and the hell with us. (I met a man in a bar, after an ACM meeting, who claimed to believe this was the purpose of it all: to create the master race that would replace us.)

*(Actually, this master-race dream seems to be fairly widespread. I've had at least one roommate who believed it, and I've heard perhaps a dozen others express this view.)*

## DEUS EX MACHINA

Obviously such beliefs are outside the realm of science or engineering. They be-

long to pure speculation; and while various mechanisms have in fact been programmed to croak, stagger, stack blocks, compose sentences and so on, to suppose that we are in any real sense anywhere near mimicking human intelligence, let alone surpassing and superseding it, is either to be totally fooled or to hanker after some curious dream from inside yourself.

As we said on the other side of the book, everybody in computers has deeper motivations and interior twists that form his own special ties to these machines; and when it comes to our choices of fantasy machines, obviously an even deeper level of psychic imprint is projecting itself into the world.

## ...EX MENSA

People who fantasize about wondrous creatures and deities they want to make out of the computer obviously have something interesting in their own heads from which that comes. Perhaps it comes from a desire for imaginary playmates, or an ambivalence toward authority, or goodness knows what; there are so many odd people at different ends of Artificial Intelligence that there may be a lot of different psychological systems at work. Or maybe artificial intelligence is just where the most brilliant, determined and eccentric dreamers go. Anyway, I can only ask the question, not give the answer.



GORDON PASK

Gordon Pask is one of the maddest mad scientists I have ever met, and also one of the nicest. An eloquent English leprechaun who dresses the Edwardian dandy, Pask sows awe wherever he goes. A former doctor and theatrical producer, Pask is one of the great international fast-talkers, conference-hopping round the globe, from Utah to Washington to England and his labratory.

In a field full of brilliant eccentrics, Pask has no difficulty standing out.

Pask is one of the Artificial Intelligencers who is working on teaching by computer, about which more will be said; but the original core of his interest is perhaps the process of conceptualization and abstraction.

Pask has done a good deal on the mathematics of self-contemplating systems, that is, symbolic representations of what it means for a creature (or entity *omega*) to look at things, see that they are alike, and divine abstract conceptions of them. A crowning moment is when Omega beholds itself and recognizes the continuity and selfhood.

## MARVIN

Marvin Minsky, "Dean of Artificial Intelligence," occupies a unique place in the world, beyond titles and beyond specifics. A personal inspiration to many, the *Oppeheimer* of AI, he may have done more to give the field a feeling, a *Zeitgeist*, than anyone else. For decades a beacon at MIT, he knows how to encourage people, subtly, and did much to create the welcoming atmosphere of the AI lab. Both generator and summarizer of ideas, he may have contributed a lot to tomorrow's intellectual vocabulary.

His brilliance is obvious to all, but his sarcasm is lost on many; the zingers and one-liners, uttered deadpan, have so many levels and nuances that simply to explain the *context* of a Minsky joke is usually hopeless. But beneath the impish grin is a combination of mischief, warmth and wistfulness. He sees so much and there is no way to tell it all.

## MARVIN BIBLIOGRAPHY

Marvin Minsky, *The Society of Mind*, Simon & Schuster, 1986.

According to Arthur C. Clarke's retroactive novel *2001: A Space Odyssey* (Signet, 1968, 95 cents), the HAL 9000 computer series began as follows:

"In the 1980s, Minsky and Good had shown how neural networks could be generated automatically—self-replicated—in accordance with any arbitrary learning pattern. Artificial brains could be grown by a process strikingly analogous to the development of the human brain."

I don't know who Good is, but these are among the lines Minsky has been working along for years, so I hope he's encouraged by the news of what he's going to accomplish.

*Minsky tells me he hasn't heard of Good either, and there's not much time left.*

Anyhow, so okay they grow the HAL 9000 in a tank. Then how come in the Death-of-Hal scene we see Keir Dullea bobbing around loosening circuit cards, just as if it were a plain old 1978 computer?

*Possible answer 1.* It is rumored that Clarke's retro-novel was Clarke's rebuttal to Kubrick's final film.

*Possible answer 2.* HAL's tanks of neural glop are controlled by PDP-11s, one to a card.

(Of course, if you take the letters after H, A and L in the alphabet, you get I, B, and M. So maybe those are 1130s.)

Why are people so emotional about AI? Is the real issue whether "intelligence" can be implemented artificially? Or whether there can be artificial consciousness?



**"AI is easy!  
When they  
say  
'knowledge,'  
read 'data.'  
When they  
say  
'heuristics,'  
just read 'a  
bag of  
programming tricks.'  
When they  
say 'expert  
system,' read  
'hack.'"**  
**—A cynic from  
Michigan**

***In a way, AI is a new branch of philosophy. But it has introduced a curious new standard of rigor: an AI publication is a writing about a program that runs, and in some way demonstrates an idea, exemplifying the abstractions that are being talked about. This is a curious new development in philosophical discourse.***

## “SYNTHETIC PSYCHOLOGY”

In a way much of artificial intelligence has always been synthetic psychology, that is, figuring out what minds would be like if they embodied certain principles, first one set of possibilities, then another.

But the term is new. It comes from Valentino Braitenberg's curious book *Vehicles*, in which he works out the possible behaviors of imaginary cars that respond to light and have different kinds of internal structure.

## ● CYBERNETICS

The term “cybernetics” is heard a lot, and is one of those terms which, in the main, mankind would be better off without.

The term “cybernetics” was coined by Norbert Wiener, the famously absent-minded mathematician who (according to legend) often failed to recognize his own children. Wiener did pioneering work in a number of areas. A special concern of his was the study of things which are kept in control by corrective measures, or, as he called it, *Feedback*. The term “cybernetics” he made out of a Greek word for *steersman*, applying it to all processes which involve corrective control. It turns out that almost everything involves corrective control, so the term “cybernetics” spreads out as far and as thinly as you could possibly want. (The public is under the general impression that “cybernetics” refers to computers, and the computer people should be called “cyberneticians.” There seems to be nothing that can be done about this. See “Cybercrud,” p. 27. This is an even worse term meaning “steering people into the crud,” specifically, putting things over on people using computers.)

Properly, the core of “cybernetics” seems to deal with control linkages, whether in automobiles, cockroaches or computers. However, people like Pask, von Foerster, Ashby (and so on) appear to extend the concept generally to the study of forms of behavior and adaptation considered in the abstract. The validity and fascination of this work, of course, is quite unrelated to what you call it.

Look! Up in the sky!

It's a database!

It's a new programming method!

It's an...

**EXPERT SYSTEM!**

Artificial intelligence is like a glacier. When a part breaks off, that part isn't glacier anymore.

Many things now ordinary in the computer field used to be “parts of artificial intelligence”—computer graphics, linked lists (and many higher data structures), and so on. Now they're just familiar techniques.

Recently, with much ballyhoo, the “ex-

pert system” became a Saleable Object, and many newcomers to the computer field speak of the coming of the Expert System as if it were the coming of the Messiah.

An expert system is simply a way of codifying unruly rules—rules for when to do what that involve a lot of “sometimes” and “maybe's” and “unless's”—rules which are like those implicit and intuitive ways of deciding something that an “expert” in some field gains from a lifetime of experience. Not easy to codify and pass on, which is why it took so long to learn them in the first place.

An expert system, now, is a codification of such intricacies—a combination of database and decision procedures—that makes decisions and judgements that the human might make. This codification is called a **rule base**, and it handles more complicated sets of conditionals than any sane programmer could make with IF statements.

The classic example is a system called MYCIN, which is a database on prescription drugs.

Now perhaps you thought, like many of us, that the prescribing of drugs could be a simple matter of following lists and tables. But it isn't.

The possible interactions of drugs with one another and other conditions of the patient—their “contraindications”—do not reduce to an orderly set, but are an elaborate tangle. The physician must know these interactions in some detail.

MYCIN, then, is a system for prescribing—and for checking prescriptions. No doctor cares to relinquish the authority of prescribing, but it's sometimes nice to have a knowledgeable advisor checking your work.

In a famous example, MYCIN was being demonstrated to a group of doctors and it came up with a situation where a certain drug should not be used. “That's not correct,” said a doctor in the room. They checked and it was correct. “Excuse me,” said the doctor, and hurried from the room to change the prescription of a patient on the basis of what he had just learned.

(It would be a nice test of an expert system to see if it could figure out the rules of English grammar—or simply English spelling—from examples. I don't know if that's been done.)

So expert systems tend to surprise us

and do things in ways that **look human**—i.e., inscrutable and authoritative—and so now a lot of expert system products are out there—supposedly ready to have all those complex rules read into them, about when to do what, and then dazzle you with their human-like performance.

There's just a couple of problems.

First, you have to get that complex knowledge in.

An individual can't just say all that he or she knows (which is impossible). Typically, setting up an expert system is a long-drawn-out interview process where the setter-upper probes and probes and tries examples, and the Human Expert says well, sort of, but I guess it's more complicated than I just said. . . . And the expert system makes up its own rules from the examples, and tries new examples on the expert.

Suppose you're a successful stock market player, and you're tired of doing it yourself, you think an expert system could handle your transactions for you. Now come the two problems: first, getting your style of operation codified into the expert system; and second, when you **think** it has your judgement, comes the big question: Will you bet on it?

And that is the other big problem with Expert Systems. Since they behave inscrutably, **how do you know when to trust them?**

So in a way, the expert system has reached a par with humans: it's good enough to make you rightly uncertain and suspicious. As you should be with regard to human experts.

**BIBLIOGRAPHY**

Terry Winograd, “When Will Computers Understand People?” *Psychology Today* May '74, 73-9.

Nicholas Negroponte, *The Architecture Machine*. MIT Press, 1970. Takes the view that computers should be made into magical servants which not only handle bothersome details, but more or less read our minds as well.

Leonard Uhr, *Pattern Recognition, Learning and Thought: Computer-Programmed Models of Higher Mental Processes*. Prentice-Hall, 1973.

Arthur W. Holt, “Algorithm for a Low Cost Hand Print Reader.” *Computer Design* Feb '74, 85-89.

Edward A. Feigenbaum, *Computers and*

# INFORMATION RETRIEVAL

"Information Retrieval" is one of those terms that laymen throw around as if it were a manhole cover. It sounds as though it means so much, so very much. And so you actually hear people say things like: "But that would mean ... (pregnant pause) ... Information Retrieval!!!" Similarly, some of the hokey new copyright notices you see in books from With-It publishers intone that said books may not be "placed in any information retrieval system..." I take this to mean that the publishers are forbidding you to put the book on a bookshelf, because "information retrieval" simply means any way at all of getting back information from anything. A bookshelf, since it allows you to read the spines of the books, is indeed an Information Retrieval System.

It happens, incidentally, that the phrase "information retrieval" was coined in the forties by Calvin Mooers, inventor of TRAC™ Language (see p. 61). (If Wiener had coined it he might have called it Getback. If Diebold had coined it, it might have been Thoughtomation.)

Anyhow, numerous entirely different things go on in the field, all under the name of Information Retrieval. Here are some.

1. *Non-computer retrieval.* (See Becker and Hayes, *Automatic Information Retrieval*.) These things are kind of old-fashioned fun—cards with holes punched along the edge, for instance, that you sort with knitting needles, or the more recent systems with holes drilled in plastic cards. Trouble is, of course, that computers are becoming much more convenient and even less expensive than these, counting your own time as being worth something.

2. *Document Retrieval.* This basically is an approach that glorifies the old library card file, except now the stuff is stored in computers rather than on cards. But what's stored is still the name of the document, who wrote it, where it was published and so on. Obviously helpful to librarians, but scarcely exciting.

3. *Automatic document indexing.* Some organizations find it helpful to have

*In Terry Gilliam's film "Brazil," the term "information retrieval" appears to apply to all forms of oppression and spying. This is incorrect, but his heart's in the right place.*

It is a truism that Mendel's theories of genetics got "lost" after publication in 1865, to be rediscovered in 1900. "If only there had been proper information retrieval under the right categories," people often say. Recent studies indicate that the publication containing Mendel's paper reached, or got nearly to, "practically all prominent biologists of the mid-nineteenth century." (*Scientific American*, July '68, 55.)

I take this as suggesting the problem isn't categorical retrieval at all. It's multi-connected availability (see "Hypertext," p. DM 29).

*Thought.* McGraw-Hill, 1963. Old but still good for orientation.

A journal: *Artificial Intelligence* (North-Holland Publishing Co., Journal Division; P.O. Box 211, Amsterdam, The Netherlands. Was \$26.50 a year in 1973. This'll show you what they're thinking about now.

Roger Lind, "The Robots Are Coming, The Robots Are Coming." *Oui*, Feb. 1974.

Edward W. Kozdrowicki and Dennis W. Cooper, "COKO III: the Cooper-Koz Chess Program." *CACM* July '73, 411-427.

Greenblatt, R. D., Eastlake, D. E., and Crocker, S. D., "The Greenblatt Chess Program." *Proc. FJCC* '67, 801-810.

R. C. Gammill, "An Examination of Tic-Tac-Toe-like Games." *Proc. NCC* '74, 349-355.

AI Expert magazine. Covers expert systems as a market.

Patrick Henry Winston, *Artificial Intelligence*. Addison-Wesley, 1984.

Joseph Weizenbaum, *Computer Power and Human Reason: From Judgment to Calculation*. W. H. Freeman & Co, 1976. *The famous putdown of AI.*

Demongeot, J. et al., *Dynamic Systems and Cellular Automata*. Academic Press, 1985. *This one is meta-wacko.*

## GOOD OVERVIEWS

Pamela McCorduck, *Machines Who Think*. Freeman, 1979.

George Johnson, *Machinery of the Mind: Inside the New Science of Artificial Intelligence*. Times Books, 1986.

Howard Gardener, *The Mind's New Science: a History of the Cognitive Revolution*. Basic Books, 1985.

## THE BEST BOOKS TO GET REALLY

To learn what artificial intelligence people really do, rather than generalizations about it, consider the two big (but very browsable and readable) anthologies that represent the two big AI points of view (MIT and Stanford/Xerox PARC).

Patrick Henry Winston and Richard Henry Brown (eds.), *Artificial Intelligence: an MIT Perspective* (two volumes). MIT Press, 1979.

Avron Barr et al. (eds.), *The Handbook of Artificial Intelligence* (three volumes). HeurisTech Press and William Kaufmann, Inc., 1982.



a computer try to figure out what a book is about, rather than have a person look at it and check. (I don't see why this saves anything, but there you are.) Anyway, the text of the document (or selected parts) are poured through a computer program that selects, for instance, *keywords*, that is, the most important words in it, or rather words the program thinks are most important. Then these keywords can go on the headings of library file cards, or whatever.

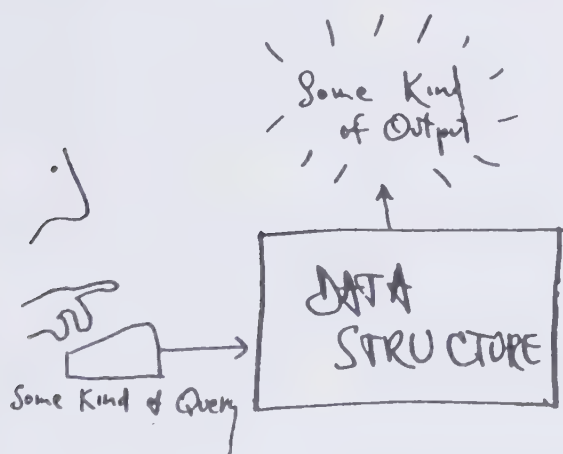
There are various related systems by which people study, for instance, the citations between articles, but we won't get into that.

4. *Content retrieval.* Now we're getting to the sexy stuff. A system for content retrieval is one that somehow stores information in a computer and lets you get it back out.

The trick on both counts is of course how.

Well, as we said on the other side of the book, any information stored in a computer has a *data structure*, which simply means whatever arrangement of alphabetical characters, numbers and special codes the computer happens to be saving.

In a content-retrieval system, information on some subject is somehow jammed into a data structure—possibly even by human coders—and then set up so people can get it back out again in some way. Lots of possibilities here, get it?



In the most startling of these systems, the QAS, or "Question-Answering System," some sort of dialogue program (see "Artificial Intelligence," nearby) tries to give you answers about the data struc-

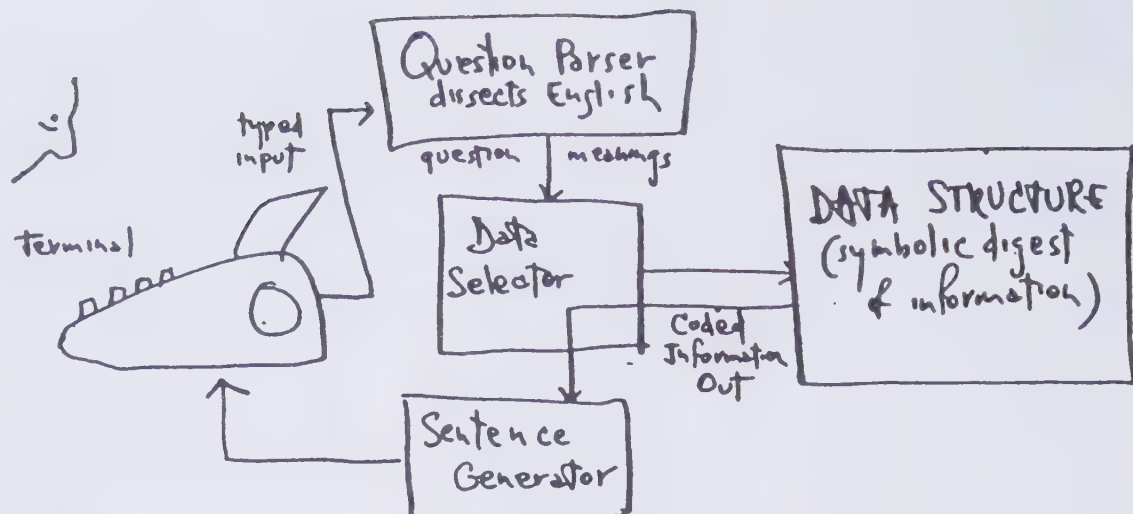
ture. But this means there have to be a whole lot of programs:

These systems can be quite startling in the way they seem to understand you (see Licklider book; also Winograd piece under Artificial Intelligence). But they *don't* understand you. They are just poor dumb programs.

Many people (including Licklider) seem to see in Question-Answering Systems the wave of the future. Others, like this author, are skeptical. It's one thing to have a system that can deduce that Green's House is West of Red's House from a bunch of input sentences on the subject, but the question of how much these can be improved is in some doubt.

A system that can answer the question, "What did Hegel say about determinism?" is some ways away, to put it mildly.

Then there is the matter of consistency. The really interesting subjects are



the ones where different authors claim opposing facts to support opposite conclusions. In other words, there is inconsistency *within* the content of the field. In this case such systems are going to have a problem. (See "Rashomonics and the Perplex," p. DM 34.)

Another fundamental point is this. It may be easy enough to program a system to answer the question,

WHAT TIME DOES THE NEXT PLANE LEAVE FOR LAGUARDIA?

but it is a lot simpler to have it display schedules your eye can run down, or allow you to go look at some kind of graphic display.

Speaking personally, I don't like talking to machines and I don't like their talking back to me. I'm not saying you have to agree, I'm just telling you you're allowed to feel that way.

5. *Screen summaries.* These systems let you sit at a computer display screen and read summaries of various things, as well as run through them with various programs to look for keywords. (The *New York Times* now offers such a system, costing over a thousand dollars a month to subscribers.)

6. *"Full-text systems."* These are systems that one way or another allow you to read all the text of something from a computer display screen. There are those of us who see these as the wave of the future, but many others are perfectly outraged at the thought. (*Hypertext* systems, now, are setups that allow you to read *interconnected* texts from computer display screens. See pp. DM 10-11, DM 13.)

This has been brief and has skipped a lot. Anyway, as you see, IR is no one thing.

## BIBLIOGRAPHY

- Vannevar Bush, "As We May Think." *Atlantic Monthly*, July 1945, 101-108.
- Theodor H. Nelson, "As We Will Think." Proc. Online '72 Conference, Brunel U. Uxbridge, England.
- G. Salton, "Recent Studies in Automatic Text Analysis and Document Retrieval." *JACM*, Apr '73, 258-278.
- Donald E. Walker (ed.), *Interactive Bibliographic Search: The User/Computer Interface*. AFIPS Press, 1971. \$15.
- Theodor H. Nelson, "Getting It Out of Our System." George Schechter (ed.), *Information Retrieval: A Critical View*. Thompson Book Co., 1967.
- J.C.R. Licklider, *Libraries of the Future*. MIT Press, 1965. Clear and readable summary of the rest of the field, then he goes on to advocate "procognitive systems," systems that will digest what's known in any field and talk back to you, using techniques of artificial intelligence. Whatever its other merits, this book is great for shaking people up, especially librarians. It seems so official.

# COMPUTER-ASSISTED INSTRUCTION

Like Artificial Intelligence and Information Retrieval, Computer-Assisted Instruction sounds like something exact and impressive, but is in fact a scattering of techniques tied together only nominally by a general idea.

The real name for it should be Automated Dialogue Teaching. That would immediately allow you to ask, *should* computer teaching use dialogues? But they don't want you to ask that.

In the classic formulation of the early sixties, there were going to be three levels of CAI: "drill-and-practice" systems, much like teaching machines, that simply helped students practice various skills; a middle level (often itself called, confusingly, "computer-assisted instruction"); and a third level, the Socratic system, which would supposedly be

Ideal. Students studying on Socratic systems would be eloquently and thoughtfully instructed and corrected by a perfect being in the machine. "We don't know how to do that yet," the people keep saying. *Yet*, indeed.

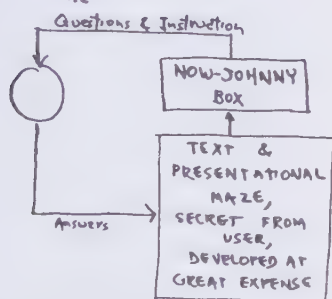
(My personal view on this subject, expressed in an article (following) is that Computer-Assisted Instruction in many ways extends the worst features of education as we now know it into the new realm of presentation by computer.)

## BIBLIOGRAPHY

George B. Leonard, *Education and Ecstasy*. Delacorte, 1968. Argues for making education an enthusiastic process.

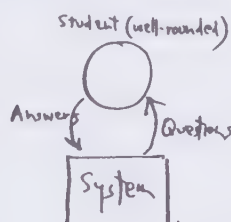
Theodor H. Nelson, "No More Teachers' Dirty Looks." Follows.

### 2. "FULL CAI"

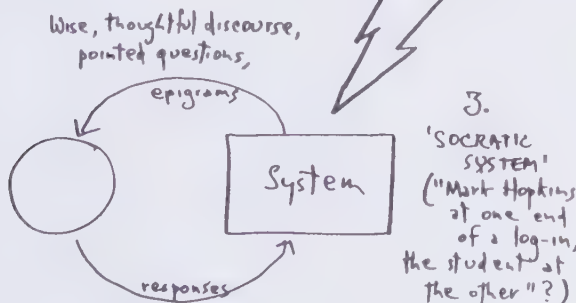


(Whereas in Hypertexts the text maze should not be a secret. See pp. DM 29-33.)

### 1. DRILL-AND-PRACTICE



## ARTIFICIAL INTELLIGENCE



3. 'SOCRATIC SYSTEM' ("Mark Hopkins at one end of a log-in, the student at the other"?)

This is a true story. (The details are approximate.) It may provide certain insights.

An Assistant Commissioner of Education was being shown a CAI system by representatives of a large and well-known computer company.

On one side of the Commissioner stood a salesman, who wanted him to be impressed. On the other side stood one Dr. S., who knew how the system worked.

The terminal, demonstrating a history program that had hurriedly been put together, typed: WHO CAPTURED FORT TICONDEROGA?

"Can I type anything?" asked the Assistant Commissioner.

"Sure," said the salesman, ignoring the frantic head-shaking of Dr. S.

The Assistant Commissioner typed: Gypsy Rose Lee.

The machine replied: NO, BUT YOU'RE CLOSE. HE CAPTURED QUEBEC A SHORT TIME LATER.

The Assistant Commissioner evidently enlivened many a luncheon with that one, and Computer-Assisted Instruction was effectively dead for the rest of the administration.

Does the name  
PAVLOV  
ring a bell?

## Another Anecdote

Some of us have been saying for a long time that learning from computers ought to be under control of the student.

One group (never mind who) has taken hold of this idea and gotten a lot of funding for it under the name of STUDENT CONTROL. This group talks as if it were some kind of scientific breakthrough.

A friend of mine suggests, however, that this phrase may have brought the funding because administrators thought it meant control of the student.



The following article appeared in the September 1970 issue of *Computer Decisions*, and got an extraordinary amount of attention. I have changed my views somewhat—we all go through changes, after all—but after consideration have decided to re-run it in the original form, without qualifications, mollifications or anything, for its unity. Thanks to *Computer Decisions* for use of the artwork by Gans and for the Superstudent picture on the cover, whose artist unfortunately insists on preserving his anonymity.

An interesting point, incidentally, is that people read this a lot of different ways. One Dean of Education hilariously misread it as an across-the-board plug for CAI. Others read in it various forms of menace or advocacy of generalized mechanization. One letter-writer said *I* was a menace but at least writing articles kept me off the streets. Here is my fundamental point: *computer-assisted instruction, applied thoughtlessly and imitatively, threatens to extend the worst features of education as it is now.*

# NO MORE TEACHERS' DIRTY LOOKS

by Theodor H. Nelson  
The Nelson Organization  
New York

Some think the educational system is basically all right, and more resources would get it working again. Schools would do things the same way, except more so, and things would get better.

In that case the obvious question would be, how can computers help? How can computers usefully supplement and extend the traditional and accepted forms of teaching? This is the question to which present-day efforts in "computer-assisted instruction"—called CAI—seem to respond.

But such an approach is of no possible interest to the new generation of critics of our school system, people like John Holt (*Why Children Fail*), Jonathan Kozol (*Death at an Early Age*) and James Herndon (*The Way It Spozed To Be*). More and more, such people are severely questioning the general framework and structure of the way we teach.

These writers describe particularly ghastly examples of our schooling conditions. But such horror stories aside, we are coming to recognize that schools as we know them appear designed at every level to sabotage the supposed goals of education. A child arrives at school bright and early in his life. By drabness we deprive him of interests. By fixed curriculum and sequence we rob him of his orientation, initiative and motivation, and by testing and scoring we subvert his natural intelligence.

Schools as we know them all run on the same principles: iron all subjects flat and then proceed, in groups, at a forced march across the flattened plain. Material is dumped on the students and their responses calibrated; their interaction and involvement with the material is not encouraged nor taken into consideration, but their dutifulness of response is carefully monitored.

While an exact arrangement of intended motivations for the student is preset within the system, they do not usually

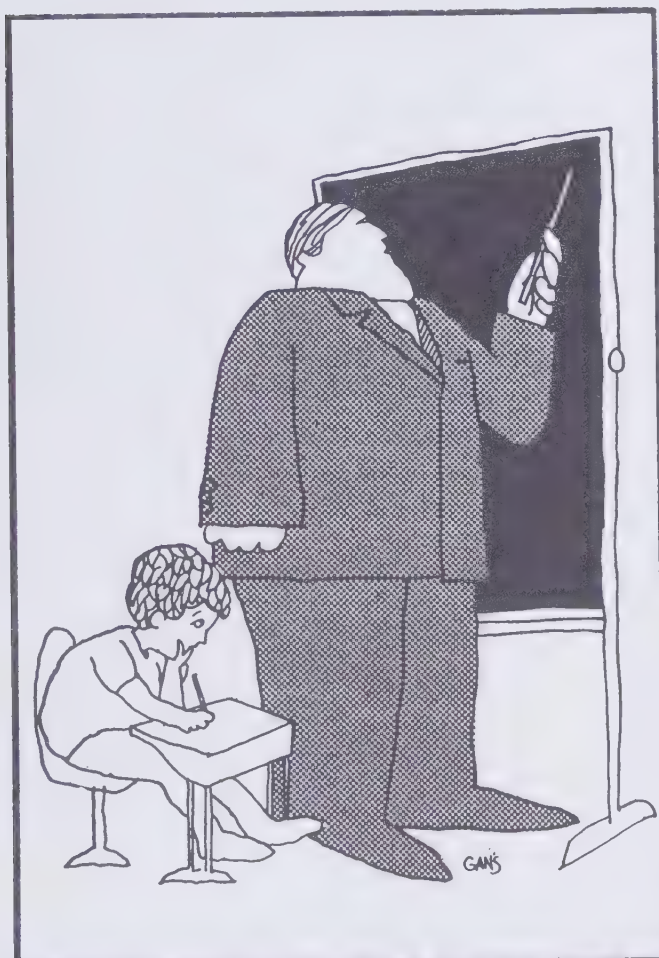
take effect according to the ideal. It is not that students are *unmotivated*, but motivated *askew*. Rather than seek to achieve in the way they are supposed to, students turn to churlishness, surliness, or intellectual sheepishness. A general human motivation is god-given at the beginning and warped or destroyed by the educational process as we know it; thus we internalize at last that most fundamental of grownup goals: just to get through another day.

Because of this procedure our very notion of human ability has suffered. Adult mentality is cauterized, and we call it "normal". Most people's minds are mostly turned off most of the time. We know virtually nothing of human abilities except as they have been pickled and boxed in schools; we need to ignore all that and start fresh. To want students to be "normal" is criminal, when we are all so far below our potential. Buckminster Fuller, in *I Seem To Be A Verb*, says we are all born geniuses: Sylvia Ashton-Warner tells us in *Teacher* of her success with this premise, and of the brilliance and creative potential she was able to find in all her schoolchildren.

Curricula themselves destructively arrange the study situation. By walls between artificially segregated "studies" and "separate topics" we forbid the pursuit of interest and kill motivation.

In ordinary schooling, the victim cannot orient himself to the current topic except by understanding the official angle of approach and presentation. Though tie-ins to previous interests and knowledge are usually the best way to get an initial sense of a thing, there is only time to consider the officially presented tie-ins. (Neither is there time to answer questions, except briefly and rarely well—and usually in a way that promotes "order" by discouraging "extraneous" tie-ins from coming up.)

The unnecessary division and walling of subjects, sequencing and kibbling of material lead people to expect simplifications, to feel that naming a thing is understanding it, to fear complex wholes; to



Did you find school dismal and dreary?  
Did it turn you off?  
Here the author proposes safe and legal  
ways to turn kids on.



believe creativity means recombination, the parsing of old relations, rather than synthesis.

Like political boundaries, curriculum boundaries arise from noticeable features of a continuum and become progressively more fortified. As behind political borders, social unification occurs within them, so that wholly dissimilar practitioners who share a name come to think they do the same thing. And because they talk mainly to each other, they forget how near is the other side of the border.

Because of the fiction of "subjects," great concern and consideration has always gone into calculating the "correct" teaching sequence for each "subject." In recent years radical new teaching sequences have been introduced for teaching various subjects, including mathematics and physics. But such efforts appear to have been misinformed by the idea of supplanting the "wrong" teaching sequence with the "right" teaching sequence, one which is "validated." Similarly, we have gone from a time when the instructional sequence was a balance between tradition and the lowest common denominator of each subject, to a time when teachers may pick "flexible optimized strategies" from textbooks. And this all ignores a simple fact: all are arbitrary. Instructional sequences aren't needed at all if the people are motivated and the materials are clear and available.

Testing as we know it (integrated with walled curricula and instructional sequences) is a destructive activity, particularly for the orientation which it creates. The concerns of testing are extraneous: learning to figure out low-level twists in questions that lead nowhere, under pressure.

The system of tensions and defenses it creates in the student's personality are unrelated to the subject or the way people might relate to the subject. An exploitive attitude is fostered. Not becoming involved with the subject, the student grabs for rote payoff rather than insight.

All in a condescending circumstance. Condescension is built into the system at all levels, so pervasive it is scarcely noticed. Students are subjected to a grim variety of put-downs and denigrations.

While many people evidently believe this to be right, its productivity in building confident and self-respecting minds may be doubted.

The problems of the school are not particularly the teacher's fault. The practice of teaching is principally involved with managing the class, keeping up face, and projecting the image of the subject that conforms to the teacher's own predilections. The educational system is thereby committed to the fussy and prissy, to the enforcement of peculiar standards of righteousness and the elevation of teachers—a huge irrelevant shell around the small kernel of knowledge transmitted.

The usual attacks on computer teaching tend to be sentimental and emotional pleas for the alleged humanism of the existing system. Those who are opposed to the use of computers to teach generally believe the computer to be "cold" and "inhuman." The teacher is considered "warm" and "human." This view is questionable on both sides.

The computer is as inhuman as we make it. The computer is no more "cold" and "inhuman" than a toaster, bathtub or automobile (all associated with warm human activities). Living teachers can be as inhuman as members of any people-prodding profession, sometimes more so. Computerists speak of "freeing teachers for the creative part of their work"; in many cases it is not clear what creative tasks they could be freed for.

At the last, it is to *rescue* the student from the inhuman teacher, and allow him to relate directly and personally to the intrinsically interesting subject matter, that we need to use computers in education.

Many successful systems of teacherless learning exist in our society: professional and industrial magazines; conventions and their display booths and brochures; technical sales pitches (most remarkably, those of medical "detail men"); hobbyist circles, which combine personal acquaintance with a round of magazines and gatherings; think-tanks and research institutes, where specialists trade fields, and the respectful briefing.

None of these is like the conventional classroom with its haughty resource-chairman; they are not run on condescension, and they get a lot across. We tend to

## Some Premises Relevant To Teaching

1. The human mind is born free, yet everywhere it is in chains. The educational system serves mainly to destroy for most people, in varying degrees, intelligence, curiosity, enthusiasm, and intellectual initiative and self-confidence. We are born with these. They are gone or severely diminished when we leave school.

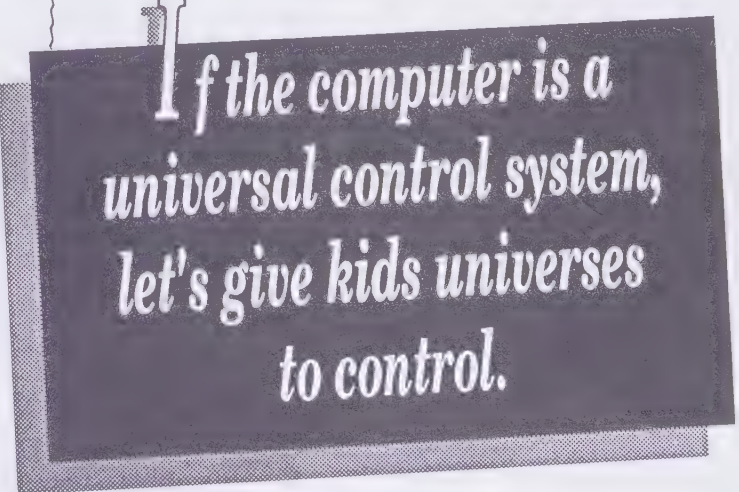
2. Everything is interesting, until ruined for us. Nothing in the universe is intrinsically uninteresting. Schooling systematically ruins things for us, wiping out these interests; the last thing to be ruined determines your profession.

3. There are no "subjects." The division of the universe into "subjects" for teaching is a matter of tradition and administrative convenience.

4. There is no natural or necessary order of learning. Teaching sequences are arbitrary, explanatory hierarchies philosophically spurious. "Prerequisites" are a fiction spawned by the division of the world into "subjects"; and maintained by not providing summaries, introductions or orientational materials except to those arriving through a certain door.

5. Anyone retaining his natural mental facilities can learn anything practically on his own, given encouragement and resources.

6. Most teachers mean well, but they are so concerned with promoting their images, attitudes and style of order that very little else can be communicated in the time remaining, and almost none of it attractively.



If the computer is a  
universal control system,  
let's give kids universes  
to control.



think they are not "education" and that the methods cannot be transferred or extended to the regions now ruled by conventional teaching. But why not?

If everything we ate were kibbled into uniform dogfood, and the amount consumed at each feeding time tediously watched and tested, we would have little fondness for eating. But this is what the schools do to our food for thought, and this is what happens to people's minds in primary school, secondary school and most colleges.

This is the way to produce a nation of sheep or clerks. If we are serious about wanting people to have creative and energetic minds, it is not what we ought to do. Energy and enthusiasm are natural to the human spirit; why drown them?

Education ought to be clear, inviting and enjoyable, without booby-traps, humiliations, condescension or boredom. It ought to teach and reward initiative, curiosity, the habit of self-motivation, intellectual involvement. Students should develop, through practice, abilities to think, argue and disagree intelligently.

Educators and computer enthusiasts tend to agree on these goals. But what happens? Many of the inhumanities of the existing system, no less wrong for being unintentional, are being continued into computer-assisted teaching.

Although the promoters of computer-assisted instruction, affectionately called "CAI," seem to think of themselves as being at the vanguard of progress in all directions, the field already seems to operate according to a stereotype. We may call this "classic" or "conventional" CAI, a way of thinking depressingly summarized in "The Use of Computers in Education" by Patrick Suppes, *Scientific American*, September, 1966, 206-220, an article of semi-classic stature.

It is an unexamined premise of this article that the computer system will always decide what the student is to study and control his movements through it. The student is to be led by the nose through every subject, and the author expresses perplexity over the question of *how* the system can decide, at all times, *where* to lead the student by the nose (p. 219). But let us not anticipate alternatives.

It is often asserted (as by Alpert and Bitzer in "Advances in Computer-Based Education," *Science*, March 20, 1970) that this is not the only approach current. The trouble is that it *seems* to be the only

approach current, and in the expanding computer universe everyone seems to know what CAI "is." And this is it.

Computer-assisted instruction, in this classical sense, is the presentation by computer of bite-sized segments of instructional material, branching among them according to involuntary choices by the student ("answers") and embedding material presented the student in some sort of pseudo-conversation ("Very good. Now, Johnny, point at the...")

### CAI: BASED ON UNNECESSARY PREMISES

At whichever level of complexity, all these conventional CAI systems are based on three premises: that all presentations consist of *items*, short chunks and questions; that the items are arranged into *sequences*, though these sequences may branch and vary under control of the computer; and finally, that these sequences are to be embedded in a framework of *dialogue*; with the computer composing sentences and questions appropriately based on the student's input and the branching structure of the materials. Let us call such systems SIC (Sequenced-Item Conversational) systems.

These three premises are united. For there to be dialogue means there must be an underlying graph structure of potential sequences around which dialogue may be generated; for there to be potential sequences means breakpoints, and hence items.

Let us question each of the premises in turn.

1. *Is dialogue pleasant or desirable?* Compulsory interaction, whether with a talking machine or a stereotyped human, is itself a put-down or condescension. (Note that on superhighways there is often a line of cars behind the automatic toll booths, even when the manned ones are open.) Moreover, faked interaction can be an annoyance. (Consider the green light at the automatic toll booth that lights up with a "thank you.") Moreover, dialogue by simple systems tends to have a fake quality. It is by no means obvious that phony dialogue with a machine will please the student.

2. *Is the item approach necessary?* If the student were in control, he could move around in areas of material, leaving

each scene when he got what he wanted, or found it unhelpful.

3. *Are sequences necessary?* Prearranged sequences become unnecessary if the student can see what he has yet to learn, then pursue it.



The sense of prestige and participation

### CAI: UNNECESSARY COMPLICATION

The general belief among practitioners is that materials for computer-based teaching are extremely difficult to create, or "program." Because of possible item weakness and the great variety of possible sequences within the web, extensive experimentation and debugging are required. Each item must be carefully proven; and the different sequences open to a student must all be tested for their effectiveness. All possible misunderstandings by a student need to be anticipated and prevented in this web of sequences, which must be designed for its coverage, correct order, and general effectiveness.

### CAI: GENERAL WRONGFULNESS

Computers offer us the first real chance to let the human mind grow to its full potential, as it cannot within the stifling and insulting setting of existing school systems. Yet most of the systems for computer-assisted instruction seem to me to be perpetuating and endorsing much that is wrong, even evil, in our present educational system. CAI in its conventional form enlarges and extends the faults of the American educational system itself. They are:

- Conduciveness to boredom;
- The removal of opportunities for initiative;

● Gratuitous concerns, both social and administrative ("subject," "progress" in subject);

● Grades, which really reflect commitment level, anxiety, and willingness to focus on core emphasis;

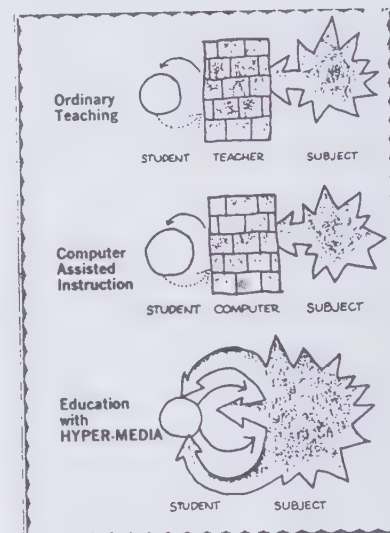
● Stereotyped and condescending treatment of the student (the "Now-Johnny" box in the computer replacing the one that sits before the class);

● The narrowing of curricula and available materials for "results" at the expense of motivation and generalized orientation;

● Destructive testing of a kind we would not permit on delicate machinery; and,

● An overt or hidden emphasis on invidious ratings. (Ungraded schools are nice, but how many units did you complete today?)

There are of course improvements, for instance in the effects of testing. In the tell-test, tell-test nattering of CAI, the testing becomes merely an irritant, but one certainly not likely to foster enthusiasm.



### BUT ISN'T CAI 'SCIENTIFIC'?

Part of CAI's mystique is based upon the idea that teaching can become "scientific" in the light of modern research, especially learning theory. It is understandable that researchers should promote this view and that others should fall for it.

Laymen do not understand, nor are they told, that "learning theory" is an extremely technical, mathematically oriented, description of the behavior of abstract and idealized organisms learning non-unified things under specific conditions of motivation and non-distraction.

Let us assume, politely, that learning theory is a full and consistent body of knowledge. Because of its name, learning theory has at least what we may call nominal relevance to teaching; but real relevance is another matter. It may be relevant as Newtonian equations are to shooting a good game of pool: implicit but without practical bearing.

Because of the actual character of learning theory, and its general remoteness from non-sterile conditions, actual relevance to any particular type of application must still be demonstrated. To postulate that the theory still applies in diluted or shifted circumstances is a leap of faith. Human beings are not, taken all together, very like the idealized pigeons or rats of learning theory, and their motivations and other circumstances are not easily controlled. Studies concerned with rate of repetition and reinforcement are scarcely relevant if the student hates or does not understand what he is doing.

I do not mean to attack all CAI, or any teaching system which is effective and gratifying. What I doubt is that SIC system for CAI will become more and more wonderful as effort progresses, or that the goal of talking tutorial systems is reachable and appropriate. And what I further suspect is that we are building boredom systems that not only make life duller but sap intellectual interest in the same old way.

## SHOULD SYSTEMS 'INSTRUCT'?

Drill-and-practice systems are definitely a good thing for the acquisition of skills and response sets, an improvement over workbooks and the like, furnishing both corrections and adjustment. They are boring, but probably less so than the usual materials. But the CAI enthusiasts seem to believe the same conversationalized chunk techniques can be extended to the realm of ideas, to systems that will tutor and chide, and that this will provide the same sort of natural interest provided by a live tutor's instruction.

The conventional point of view in CAI

claims that because validation is so important, it is necessary to have a standardized format of item, sequence and dialogue. This justifies turning the endeavor into picky-work within items and sequence complexes, with attendant curricular freeze, and student inanition and boredom. This is entirely premature. The variety of alternative systems for computer teaching have not even begun to be explored. Should systems "instruct" at all?

## 'RESPONDING RESOURCES' AND 'HYPER-MEDIA'

At no previous time has it been possible to create learning resources so responsive and interesting, or to give such free play to the student's initiative as we may now. We can now build computer-based presentational wonderlands, where a student (or other user) may browse and ramble through a vast variety of writings, pictures and apparitions in magical space, as well as rich data structures and facilities for twiddling them. These we may call, collectively, "responding resources." Responding resources are of two types: facilities and hyper-media.

A *facility* is something the user may call up to perform routinely a computation or other act, behaving in desired ways on demand. Thus JOSS (a clever desk calculator available at a terminal) and the Culler-Freed graph-plotting system (which graphs arbitrary functions the user types in) are facilities.

*Hyper-media* are branching or performing presentations which respond to user actions, systems of prearranged words and pictures (for example) which may be explored freely or queried in stylized ways. They will not be "programmed" but rather *designed, written, drawn and edited*, by authors, artists, designers and editors. (To call them "programmed" would suggest spurious technicality. Computer systems to present them will be "programmed.") Like ordinary prose and pictures, they will be *media* and because they are in some sense "multi-dimensional," we may call them *hyper-media*, following the mathematical use of the term "hyper-."

## A MODEST PROPOSAL

The alternative is straightforward. Instead of devising elaborate systems per-

mitting the computer or its instructional contents to control the situation, why not permit the student to control the system, show him how to do so intelligently, and make it easy for him to find his own way? Discard the sequences, items and conversation, and allow the student to move freely through materials which he may control. Never mind optimizing reinforcement or validating teaching sequences. Motivate the user and let him loose in a wonderful place.

Let the student control the sequence, put him in control of interesting and clear material, and make him feel good—comfortable, interested, and autonomous. Teach him how to orient himself: not having the system answer questions, all typed in, but allowing the student to get answers by looking in a fairly obvious place. (Dialogue is unnecessary even when it does not intrude.) Such ultra-rich environments allow the student to choose what he will study, when he will study it and how he will study it, and to what criteria of accomplishment he will aim. Let the student pick what he wishes to study next, decide when he wishes to be tested, and give him a variety of interesting materials, events and opportunities. Let the student ask to be tested on what he thinks he knows, when he is ready, selecting the most appropriate form of testing available.

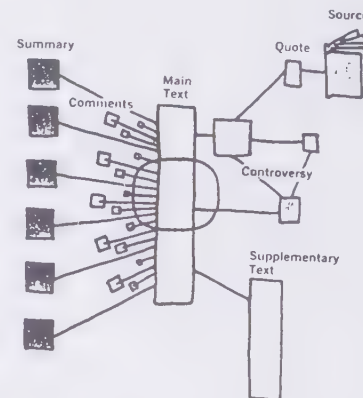
This approach has several advantages. First, it circumvents the incredible obstacles created by the dialogue-item-sequence philosophy. It ends the danger to students of bugs in the material. And last, it does what education is supposed to do—foster student enthusiasm, involvement, and self-reliance.

Under such circumstances students will actually be interested, motivated to achieve far more than they have ever achieved within the normal instructional framework; and any lopsidedness which may result will be far offset by the degree of accomplishment which will occur—it being much better to create lopsided but enthusiastic genius specialists than listless, apathetic, or cruelly rebellious mediocrities. If they start soon enough, they may even reach adulthood with natural minds: driven by enthusiasm and interest, crippled in no areas, eager to learn more, and far smarter than people ordinarily end up being.

Enthusiasm and involvement are what really count. This is why the right to explore far outweighs any administrative advantages of creating and enforcing "subjects" and curriculum sequences. The enhancement of motivation that will follow from letting kids learn anything they want to learn will far outweigh any specialization that may result. By the elimination or benign replacement of both curriculum and tests in an ultra-rich environment, we will prevent the attrition of the natural motivation of children from its initially enormous levels, and mental development will be the natural straight diagonal rather than the customary parabola.

## IS IT SO HARD? SOME IDEAS

CAI is said to be terribly hard. It would seem all the harder, then, to give students the richer and more stimulating environments advocated here. This is because of the cramped horizons of computer teaching today. Modest goals have given us modest visions, far below what is now possible and will soon be cheap.



Discrete (Chunk Style) Hypertexts

The static computer displays now associated with CAI will give way to dynamic displays driven from minicomputers, such as the IDIOM, IBM 2250/4 or Imlac PDS-1. (The last of these costs only \$10,000 now; by 1975 such a unit will probably cost \$1,000 or less.) Not only will computers be much cheaper, but their usability will improve: a small computer with a fair amount of memory will be able to do much more than it can now, including operate a complex display from its own complex data base. ■



It is generally supposed that systems like these need big computers and immense memories. This is not true if we use the equipment well, organize storage cleverly, and integrate data and display functions under a compact monitor. This is the goal of The Nelson Organization's Project Xanadu, a system intended to handle all the functions described here on a minicomputer with disk and tape.

## DISCRETE HYPERTEXTS

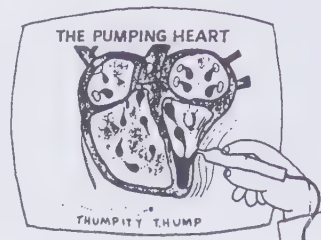
"Hypertext" means forms of writing which branch or perform on request; they are best presented on computer display screens.

In ordinary writing, the author may break sequence for footnotes or inserts, but the use of print on paper makes some basic sequence essential. The computer

display screen, however, permits footnotes on footnotes on footnotes, and pathways of any structure the author wants to create.

Discrete, or chunk style, hypertexts consist of separate pieces of text connected by links.

Ordinary prose appears on the screen and may be moved forward and back by throttle. An asterisk or other key in the text means, not an ordinary footnote, but a *jump*—to an entirely new presentation on the screen. Such jumpable interconnections become part of the writing, entering into the prose medium itself as a new way to provide explanations and details to the seeker. These links may be artfully arranged according to meanings or relations in the subject, and possible tangents in the reader's mind.



## PERFORMING HYPERGRAMS

A hypergram is a performing or branching picture: for instance, this angle, with the bar-graph of its related trigonometric functions. The student may turn the angle upon the screen, seizing it with the lightpen, and watch the related trigonometric functions, displayed as bar charts, change correspondingly.

Hypergrams may also be programmed to show the consequences of a user's prod—what follows or accompanies some motion of the picture that he makes with a pointing tool, like the heart-beat sequence.

## STRETCHTEXT™ FILLS IN THE DETAILS

This form of hypertext is easy to use without getting lost. As a form of writing, it has special advantages for discursive and loosely structured materials—for instance historical narratives.

There are a screen and two throttles. The first throttle moves the text forward and backward, up and down on the screen. The second throttle causes changes in the writing itself: throttling toward you causes the text to become *longer* by minute degrees. Gaps appear between phrases; new words and phrases pop into the gaps, an item at a time. Push back on the throttle and the writing becomes short and less detailed.

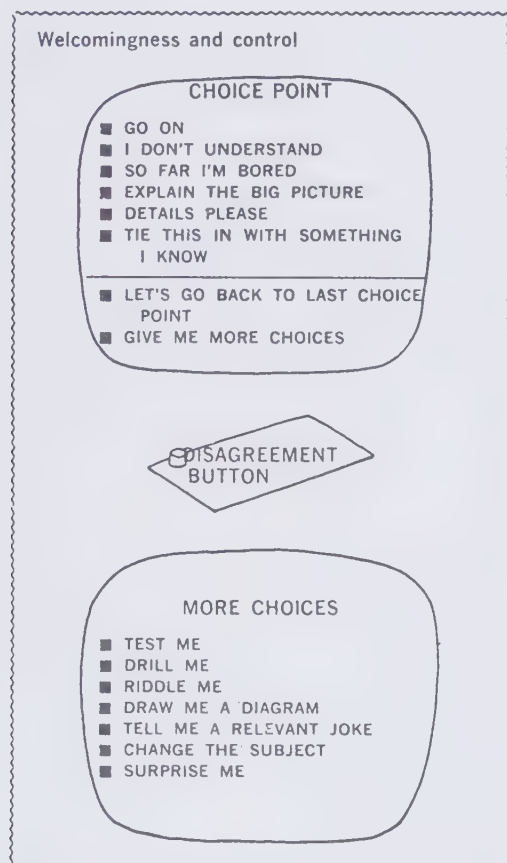
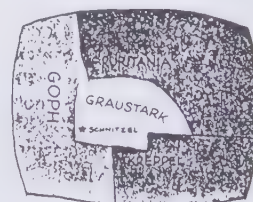
The stretchtext is stored as a text stream with extras, coded to pop in and pop out at the desired altitudes:

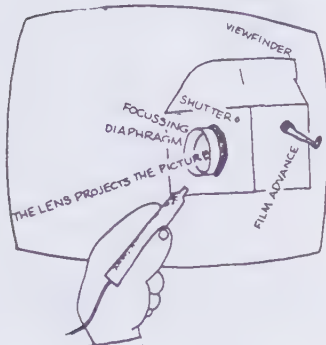
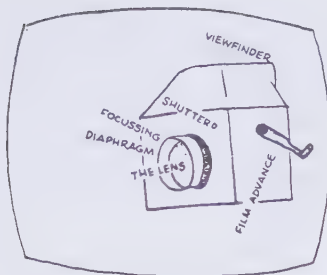
Stretchtext is a form of writing. It is read from a screen. The user controls it with throttles. It gets longer and shorter on demand.

Stretchtext, a kind of hypertext, is basically a form of writing closely related to other prose. It is read by a user or student from a computer display screen. The user, or student, controls it, and causes it to change, with throttles connected to the computer. Stretchtext gets longer, by adding words and phrases, or shorter, by subtracting words and phrases, on demand.

## HYPERMAP ZIPS UP OR DOWN

The screen is a map. A steering device permits the user to move the map around the world's surface; a throttle zooms it in. Not by discrete jumps, but animated in small changes, the map grows and grows in scale. More details appear as the magnification increases. The user may request additional display modes or "overlays," such as population, climate, and industry. Such additional features may pop into view on request.





## QUERABLE ILLUSTRATIONS: A FORM OF HYPERGRAM

A "hypergram" is a picture that can branch or perform on request. In this particular example, we see on the screen a line-drawing with protruding labels. When the student points at a label, it becomes a sliding descriptive ribbon, explaining the thing labelled. Or asterisks in an illustration may signal jumps to detailed diagrams and explanations, as in discrete hypertexts.

*Note: programs allowing you to build queriable diagrams are now available (though not with animation). These include FileVision and Guide (both, of course, for the Macintosh).*



## DISSECTION ON THE SCREEN

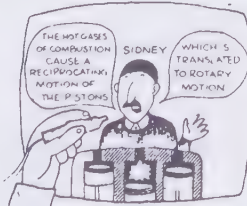
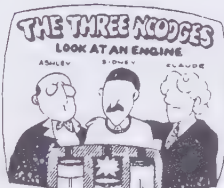
The student of anatomy may use his lightpen as a scalpel for a deceased creature on the screen. As he cuts, the tissue parts. He could also turn the lightpen into

hemostat or forceps, and fully dissect the creature—or put it back together again. (This need not be a complex simulation. Many key relationships can be shown by means of fairly simple schematic pictures, needing a data structure not prohibitively complicated.)

*Note: screen dissection programs are finally beginning to turn up in the marketplace.*

## HYPER-COMICS ARE FUN

Hyper-comics are perhaps the simplest and most straightforward hyper-medium. The screen holds a comic strip, but one which branches on the student's request. For instance, different characters could be used to explain things in different ways, with the student able to choose which type of explanation he wanted at a specific time.



## 'TECHNICALITY' IS NOT NECESSARY

Proponents of CAI want us to believe that scientific teaching requires a certain setup and format, incomprehensible to the layman and to be left to experts. This is simply not true. "Technicality" is a myth. The problem is not one of technical rightness, but what should be.

The suggestions that have been given are things that should be; they will be brought about.

*A prediction that has been coming true.*

Much of CAI is still in the old direction; instead of a simulated pedant, many teaching programs have rocketships and bears, but are still tell-and-test, tell-and-test, with Points To Score and forced sequence still.

Yet while the tell-and-test approach is still around, a whole new style has developed, and now we are beginning to see Explorables and Responding Resources popping up. Which is what I was advocating here.

This means the user (or pupil—thank goodness, not distinguishing—) can move around in the explanatory structure.

An explorable diagram has no sequence. The sequence is in the user's mind and hand, and his or her initiative is unfettered. And that's where the future of computer teaching is going.

## EXPLORABLES WITH SIMULATION

In some new explorable environments, the user can even TRY EXPERIMENTS. That's fabulous if you can get it, but much more expensive in terms of programming effort; so let me stress once again the cheaper horsepower of humble explorable pictures (hypergrams) and explorable, nonsequential writing (hypertext).

*Here are some explorables with simulation:*

*In the edu-diskette business, nice examples are Rocky's Boots and its sequel, Robot Odyssey, by Warren Robinett.*

*In Rocky's Boots, the user/player/student, in control at the screen, may wire together various circuits and see their results. (In industry, this kind of program is called a "circuit simulator." Now it's called a Construction Set.) In Robot Odyssey, Robinett expands this so you can do it with whole integrated circuits and robot effects.*

*(See also, "Laws of the Universe Hyper-Comics," nearby.)*

*The magnificent San Marcos LISP Explorer, a splendid exploration tool, allows the user to study and to practice the LISP computer language, going forward and backward through all kinds of programming examples, starting and stopping experiments at will and changing their current structure before continuing.*

## OTHER STUDENT-CONTROLLED STUFF

*Texas Instruments' redoubtable Speak'n' Spell is a portable battery device that says words aloud and has a keyboard for typing them in. The machine congratulates the tyke when the word has been correctly spelled.*

*Many kids love it. This is because it's under their control. They turn it on and off. (And can even practice in secret.)*

*The best for last: it appears that a new teaching program for younger kids, from IBM no less, allows them to write what they choose, and reads back to them what they've written. It's called Writing to Read. If so, it's a nice latter-day implementation of the way Sylvia Ashton-Warner taught New Zealand children to read and write (as celebrated in her book, Teacher.)*

***I had this argument recently with an executive from a major media company.***

***Exec: We have to allow for failure in our hyper-media systems...***

***TN: Failure? Why call it failure? Call it never-mind, something the user doesn't want to do.***

***Exec: Never-mind is a better thing to call failure.***

***TN: Failure is a bad word for never-mind.***

***In other words: as long as we proceed from a mental model of success and failure, our systems will be the same old stuff.***



WHAT THIS IS: I briefly visited Alfred Bork's CAI shop at the University of California at Irvine on a consulting basis. Bork is a really swell guy, but he's devoted to Dialogue CAI—that is, to teaching programs that have pseudo-conversations with the student. (As I've said variously already, the pseudo-conversation parts are not only expensive and difficult, but sometimes irritating and objectionable; and happier, zippier, simpler techniques are available using various techniques of old-fashioned showmanship—as from movie-making, writing and (here) the comic book.

This is my reply to Bork's question, "Well, how would you do it?"

This ties into Bork's physics display system. That is, it's intended to be a front-end program, leading into a simulation program (see pp. 149-151) allowing the user to see all kinds of motions in physical law. The program it's intended to supplant uses a dialogue.

This alternate approach lets the user choose simulations (as in Bork's physics teaching system) without a lot of dialogue. (Please note that I am in no way criticizing Bork's stunning accomplishments, but only the general dialogue approach, both for its costs and the fact that users have to do a lot of extraneous sentence-typing.)

This was an animated hypercomic for a Tektronix display scope, popular in the early '70s, whose only animation had to be *cumulative*—unless you cleared the screen.

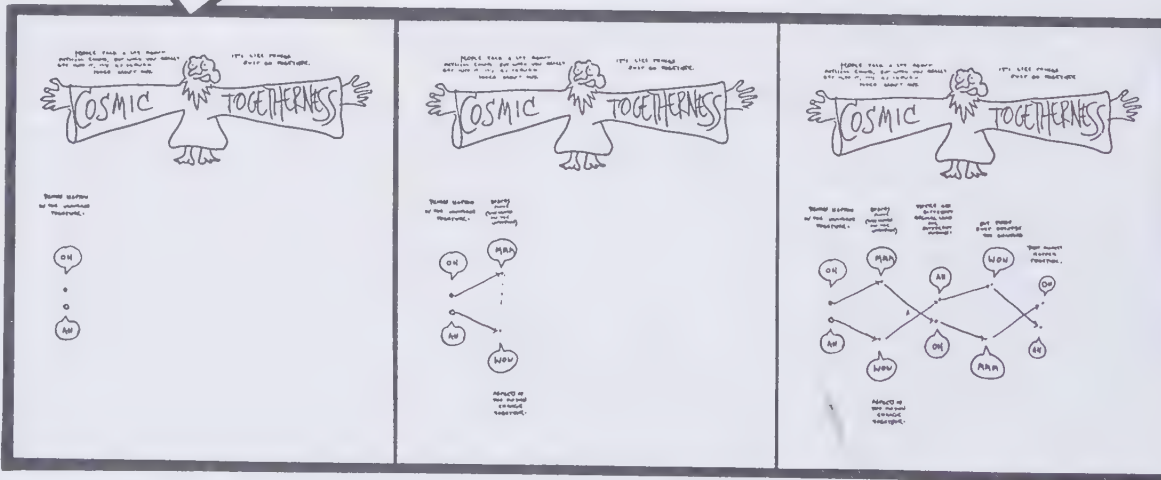
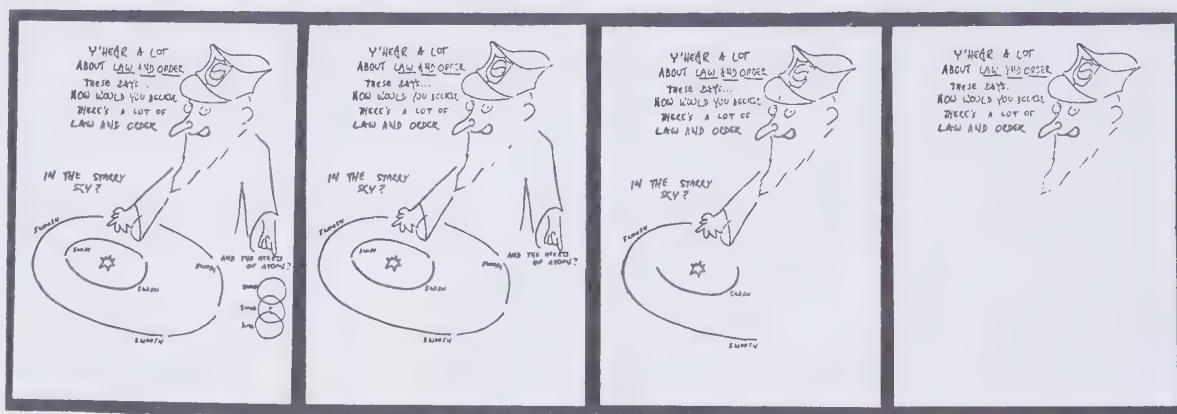
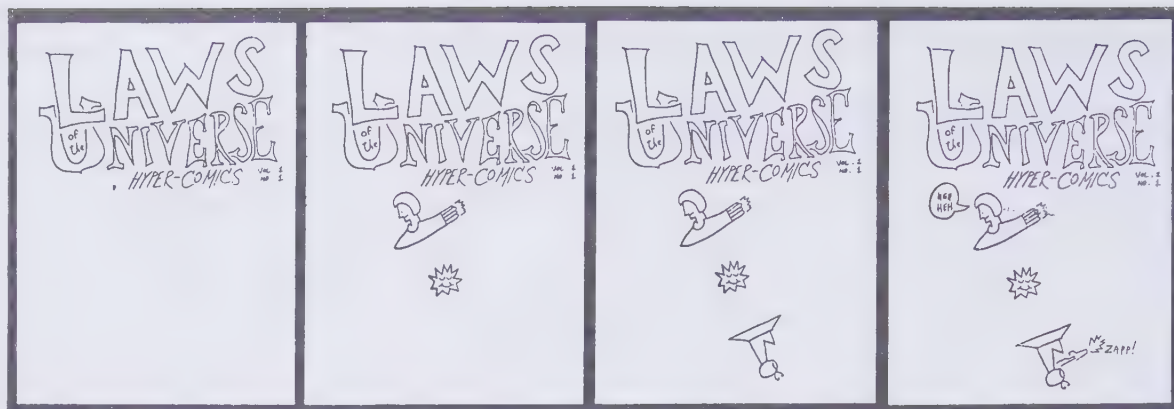
WHAT IT CONTAINS: introductory remarks; statement that physical law (as of motions) simply summarizes constant covariances. Sorry if readability is poor (Xerox™ of a Xerox™).

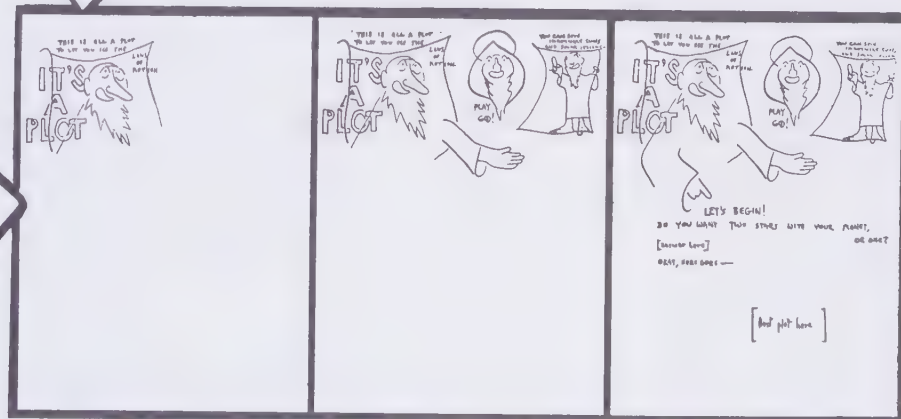
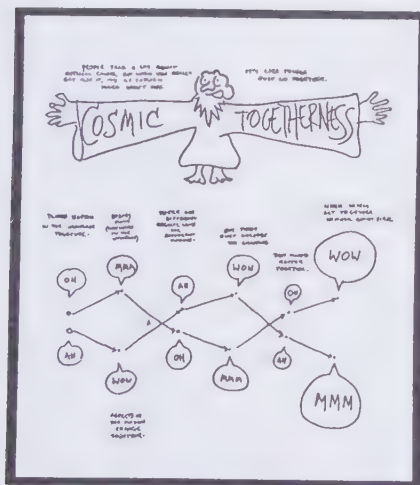
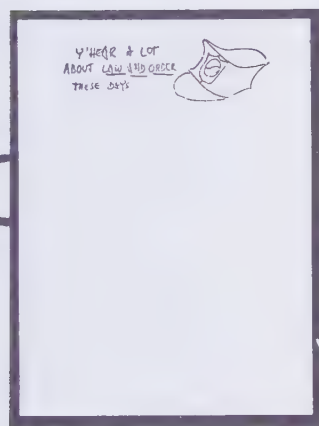
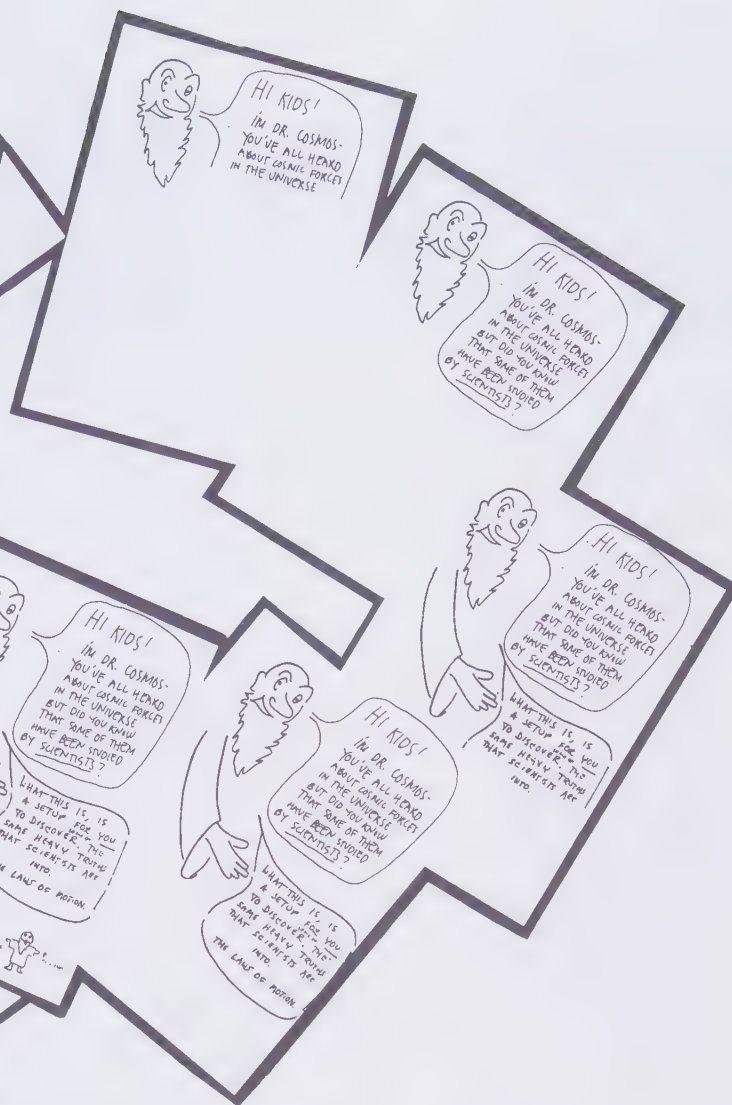
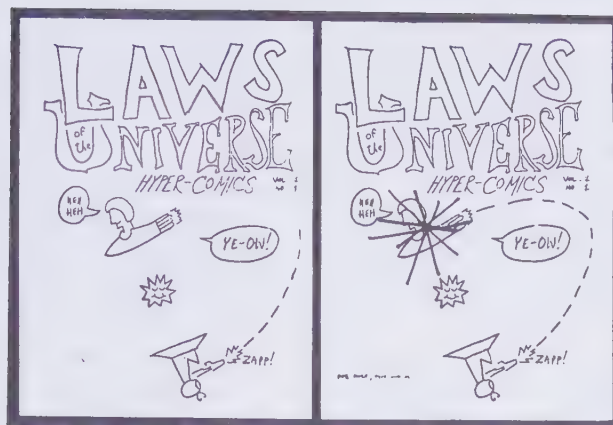
You will note the artistic problem of composing cumulative animation for a display screen.

Some people have accused me of trying to be humorous. Obviously nothing of the sort was intended. Research supported by NSF Grant No. GJ296.

## BIBLIOGRAPHY:

For comic technique, study the works of Crumb; also, comic book stands are currently featuring reprint magazines of *THE SPIRIT*, which is some of the finest stuff ever done. Also study Wally Wood in the early *MADs*.







They didn't fit anywhere else, so they  
might as well go here—

# BODY ELECTRONICS and ROBOTICS

## BODY ELECTRONICS

"I sing the body electric . . ."  
—Walt Whitman

There are various people who want to attach electronics to people's bodies and brains.

There are basically two starting points for this ambition. One is authoritarian, the other is altruistic. I am not sure both schools are not equally dangerous, however.

Let's consider first the authoritarians. Prof. Delgado of Yale has demonstrated that any creature's behavior can be controlled by jolts to the brain. Delgado has dealt especially with the *negative* circuits of the brain, that is, places where an electrical impulse causes pain (or "negative reinforcement"). In Delgado's most stunning demonstration, he stopped a charging bull with just a teeny radio signal. Enthusiastically Delgado tells us how fine this sort of thing would be for controlling Undesirable Human Behavior, too.

Now, let's consider just what we're talking about. In these experiments, needles are implanted in the creature's brain. This can involve removing a section of the skull, or it can be done merely by hammering a long hollow needle straight into the skull and thus the brain.

The researcher, or whatever we want to call him, had better know what he is doing. But due to the remarkable mass action of the brain, the destruction caused by such needles will not have observable effects if done properly.

The hollow needle, once in place, becomes a tube for shielded electrical wires, whose bare metallic tips may then be used to carry little electrical jolts, to whatever brain tissue is reached by the tip of the needle, whenever tiny signals are applied.

Now here are regions of the brain, distributed irregularly through its mysterious contents, which are loosely called

the "pleasure" and "pain" systems. They are called that because of what the organism does when you jolt it in those places. (We do not know whether jolts to these areas *really* cause pleasure or pain, because these things haven't been done to human beings. Yet. The creatures it has been done to can't tell us just how it feels; thus "pleasure" and "pain" are in quotation marks. For now.)

Anyway, what happens is this. If you stimulate a creature in the "pain" system it tends to stop what it is doing—this is called negative reinforcement—and if you stimulate it in the pleasure system, it tends to do more of what it was doing. Positive reinforcement.

Now, to some people this suggests wonderful possibilities.

Delgado, for instance, believes that this technology gives us everything we need for the control of Anti-Social Tendencies. Criminals, psychopaths and Bad Guys in general—all can be effectively "cured" (i.e., put on their best behavior) by these techniques. All we have to do, heh heh, is get into their heads, heh heh, habits of proper behavior. And with these new techniques of reinforcement, we can *really* teach 'em.

Unfortunately Delgado is probably right.

In principle this is just a drastic form of behavior control on the B.F. Skinner model (depicted also in *Nineteen Eighty-Four* and *A Clockwork Orange*). The new system is more stark and startling because of its violation of the individual's body interior, but not in principle different.

Skinner has the same naive, simple-minded solutions for everything. All "we" have to do—using "we" to mean society, the good guys, good guys acting on behalf of society, etc.—is control the behavior of the bad guys, and everything will be better, and "we" can accomplish anything "we" desire.

The reader may see several problems with this.

In the first place (and the last), there is the obvious question of who *we* are, and if *we* are going to control other people, who is going to control *us*.

At a time when our "highest" show themselves preoccupied with low retaliations and lower initiatives, we can wonder indeed if it is not more important to prevent anyone from ever getting this kind of control over humans than to facilitate it.

Even if that weren't a problem, there is the more simpleminded question of who in the existing system would use such techniques. It turns out, of course, that they would be added to what is laughably called the Correctional System, or even more laughably called the Justice System. All the sadists you could possibly want work there. (And no doubt some very nice guys—but experiments have demonstrated horrifically that decent people, turned into "guards" even for a short time, adopt the patterns of brutality we have known from time immemorial.)

So, like truncheons and electric shock therapy and solitary confinement and everything else, these techniques, if they are used, will enter the realm of Available Punishments, not to be used with clinical precision but with gratuitously brutalizing intent, new tools for punitivity and sadism. The "correctional" system would have to be magically corrected itself before such tools could be employed without simply making things *worse*. And the prospect is not good.

Such schemes grow, of course, from a caricature of the malefactor—thinking him to be some sort of miswired circuit, rather than a human being caught up in anger, pain, humiliation and unemployment.

(There are also a lot of canards about Free Will, but these do nothing for either side in this controversy.)

## NEW FACULTIES

Starting from an entirely different outlook, various designers and bio-engineers are trying to add things to the human body and nervous system, for the voluntary benefit of the recipient.

A number of research and development efforts are aimed at helping those with sensory impairments, and electronics obviously is going to be involved.

An example: a firm called Listening, Inc. in Boston, founded by Wayne Batteau (whom John W. Campbell considered one of the Great Men of Our Time), devised a system for helping the totally deaf to hear. Supposedly this could transmit the actual sensation of hearing into the nervous system by some scarcely-understood form of electrical induction. The machine was sold off; whether it ever got a safety rating I don't know.

This is the sort of thing people would like to do for the blind as well.

Now, in principle, it might be possible to transmit an image in some way to the actual visual area of the cerebral cortex. (This might or might not involve opening the skull.) Somebody's working on it.

In a related trend, numerous design groups are attempting to extend the capabilities of the human body, by means of things variously called *possums*, *waldos* and *telefactores*.

"Possums" (from Latin "I can") are devices to aid the handicapped in moving, grasping and controlling. Whatever motions the person *can* make are electronically transposed to whatever realm of control is needed, such as typewriting or guiding a wheelchair. ("Waldo" is Heinlein's term for a possum that can be operated at a distance.)

In the space program, though, they call them *telefactores*. A telefactor is a device which converts or adapts body movements by magnification or remote mimicking. Unlike possums, they are meant to be operated by people with normal faculties, but to provide, for example, superhuman strength: cradled in a larger telefactor body, a man can pick up immense loads, as the movements of his arms are converted to the movements of the greater robot arms.

Telefactores can also work from far, far away. Thus a man sitting in a booth can control, with the movements of his own arms, the artificial arms of a robot vehicle on another planet.

(This whole realm of sensory and motor mechanics and transposition is an important aspect of what I call "Fantics," discussed on p. DM 74.)

Then there are those who, like How Wachspress (see nearby), want to *expand* man's senses beyond the ordinary, into new sensory realms, by hooking him to various electronics.

## THOUGHTS

There are two problems in all of this. The first and worst, of course, is who controls and what will hold them back from the most evil doings. Recent history, both at home and abroad, suggests the answers are discouraging.

The second problem, wispish and theoretical next to that other, is whether in turning toward bizarre new pleasures and involvements, we will not lose track of all that is human. (Of course this is a question that is asked by somebody whenever anything at all changes. But that doesn't mean it is always inappropriate.)

In the face both of potential evil and dehumanization, though, we can wish there were some boundary, some good and conspicuous stopping place at which to say: *no further*, like the three-mile limit in international law of old. I personally think it should be *the human skin*. Perhaps that's old-fashioned, being long breached by the Pacemaker. But what other lines can we draw?

The prospects are horrowshow, me droogies.

## BIBLIOGRAPHY

T.D. Sterling, E.A. Bering, Jr., S.V. Pollock and H. Vaughan, Jr., *Visual Prosthesis: The Interdisciplinary Dialog*. ACM Monograph Series, Academic Press, 1971. \$21.

*In the above article I suggested we not break the skin, that being a good place to draw the line. However, that line has already been breached in many ways, and doesn't make any sense. It is immensely important to be able to amplify the faculties of the handicapped. The outstanding example is of course Stephen Hawking, Newton professor of physics at Cambridge University.*

*The prisoner of a wasting disease, Professor Hawking needs (and fortunately gets) every available aid to facilitate his outpouring of creative work.*

*Hawking, with his exalted special brilliance and academic position, is just the extreme example. Thousands of handicapped people need ways to receive from the world and speak back to it.*

*David Edell, at MIT, has been working on circuit chips that can be implanted and spliced to people's nerves as input sensors and output drivers to run mechanical arms or other output actions. While still a way from real splice-on prosthetics, this*

*indicates a direction with real promise for the handicapped.*

## PSYCHO-ACOUSTIC DILDONICS

I originally hadn't intended to include anything like this in the book, wanting it to be a family-style access catalog and all that, but this particular item seems fairly important.

Remember how we laughed at the Orgasmotron in Woody Allen's "Sleeper?" Well, it turns out not to be a joke.

An individual named How (not Howard) Wachspress, electronicer-in-residence at a San Francisco radio station, has been developing just that, except that he has more elevated purposes in mind. The secret was broken to the world in *Oui* magazine earlier this year; but Hefner, the publisher, evidently held back the more startling photographs of a model in electronically-induced ecstasy.

Wachspress' devices *transpose sound* (as audio signals) into *feelings*; you touch your body with an open-ended tube or other soft fixture attached to his device—which in turn is attached to a hi-fi.

The sensations, it is claimed, are profound and moving. You may take them anywhere on your body; the effect is deeply relaxing and emotionally engrossing. Wachspress thinks he has reached an entire neurological system that wasn't know before, much like Olds' discovery of the "pleasure center" in the brain; he sees it as a new modality of experience and a *generalization* of music and touch. That is the main point. "Hyper-reality" is where he says it gets you: a point curiously congruent with the author's own notions of hypertext and hypermedia as extension of the mental life.

This said, we can consider the prurient aspects of Wachspress' Auditac and Teletac devices (which he intends to market in a couple of years as hi-fi accessories, b'gosh). When played with the right audio, in the right places, and a good operator at the controls, they provide a sexual experience said to be of a high order.

Wachspress' work ties in interestingly with today's "awareness" movement, of which Esalen is the spiritual center, which holds that we have gotten out of touch with our bodies, our feelings, our native perceptions. As such, the Wachspress machines may be an unfolding-mechanism for the unfeeling tightness of Modern Man—as well as a less

profound treatment for "marital difficulties" and Why-Can't-Johnny-Come-Lately.

Inscrutable San Francisco! Wachspress gave a number of demonstrations of his devices in Bay Area churches, until he became disturbed at immodest uses of the probe by female communicants who had stood in line to try the machine.

(Auditac, Ltd., 1940 Washington St., San Francisco, CA 94109.)

Harry Mendell, a good friend of mine, rigged an interesting experiment while he was still in high school.

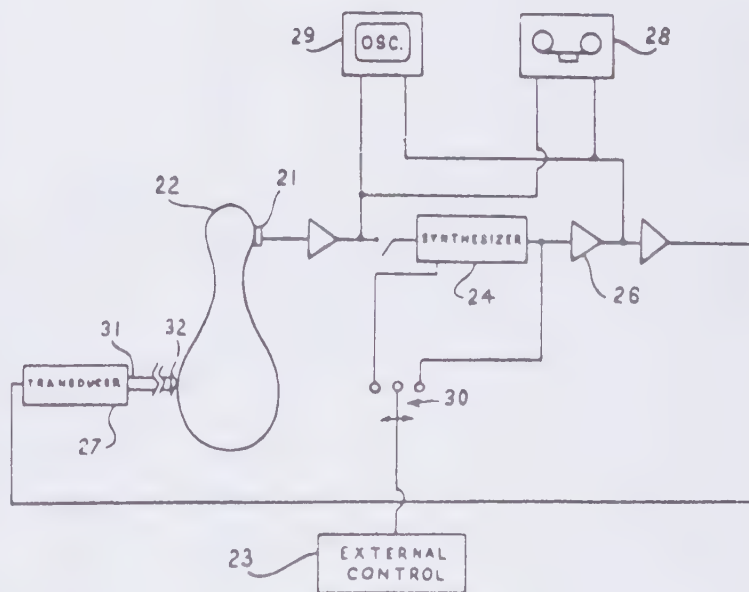
He used a little Hewlett-Packard mini-computer, which the manufacturer had generously loaned to his Knights of Columbus Computer Club of Haddonfield, N.J.

Harry hooked the Hewlett-Packard up to a CRT display (see pp. DM 10-11). At the top of the CRT, following his program, the computer continuously displayed the letters of the alphabet. A little marker (called a *cursor*) would skip along underneath the letters, acting as a marker for each of them in turn. ■

## AUDIO TRANSDUCHER

Patent #3,875,932 has now been issued for How Wachspress' electronic sex machine or whatever it is. In the illustration we see it tickling a shmoo.

After you send Wachspress his fifty-buck royalty, you can either buy the kit or a pre-built model. Concave or convex, as the poet says. (Etchings are antediluvian and waterbeds are commonplace; as an invitation, what more incisive comeuppance could be proffered?)



Speaking of Wachspress, it seems that the unusual I/O equipment offered by the Federal Screw Works (Troy, Mich.) is only a voice output device.

As one might have suspected, the Wox Box was a loudspeaker hermetically sealed in a box and leading to a garden hose.

Those I know who tried it found it a tickling throbbing, but nothing to make the earth move. The earth moves more often in California.



## THE THREE LAWS OF ROBOTICS

- 1—A robot may not injure a human being, or, through inaction, allow a human being to come to harm.
  - 2—A robot must obey the orders given it by human beings except where such orders would conflict with the First Law.
  - 3—A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.
- Handbook of Robotics, 56th Edition, 2058 A.D.,  
quoted in I. Asimov, I, Robot.

*The idea of the robot, the manufactured human, comes down to us a long way. The Yiddish tale of the Golem from the Middle Ages, and Mary Wollstonecraft Shelley's novel, Frankenstein, are key precursors. The term robot came from the play "R.U.R." (Rossum's Universal Robots) by Czech author Karel Kapek. The Slavic root robot means work (Russian robotnik means "worker.")*

*The Kapek robot was a fleshen android. But metallic models of the familiar kind do go back further, at least to the Oz series (the wind-up character Tik-Tok, and arguably the Tin Woodman). But the full chromium robot perhaps emerged in Fritz Lang's epic silent film "Metropolis."*

*While interactive systems can excellently mimic plausible human behavior and pass low-level Turing tests, we are nowhere near programming (or otherwise fabricating) anything approaching a mind that can act reasonably in any social or tactical situation as well as a six-year-old human, let alone perceive it.*

*It is clear now that nothing like these two-legged chromium smarties is going to be built in the foreseeable future. The interesting thing, then, is that this image is so important to us.*

*"Your plastic pal who's fun to be with!"*

—Robot ad by  
Sirius Cybernetics  
Corporation in  
Hitchhiker's  
Guide to the  
Galaxy

Harry rigged one more external device: a set of electrodes. These would be strapped, harmlessly, to the head of a subject. Harry's computer program used these electrodes to measure alpha rhythm, one of the mysterious pulses in the brain that come and go.

Every time the subject flashed alpha, Harry's program would copy the letter above the cursor to the bottom of the screen.

Sitting in this rig, subjects were able to learn, rather quickly, TO TYPE WORDS AND SENTENCES. Just by flashing alpha rhythm when the cursor was under the right letters.

Jubilant, Harry showed this setup to an eminent neuro-physiologist from a great university nearby, a man specializing in electrode hookups. Harry was a highschool student and did not understand about Professionalism.

"What's so great about that?" sniffed the eminent professional. "I can type faster."

So Harry dropped that and went on to other stuff.

*"Robotics" may best be described as anything that involves moving parts on output from a computer program, and distributed sensors on which their motions are based.*

## ROBOTS AND ROBOIDS

*A place that I frequented as a kid in the Forties had a Robot left over, I think, from the World's Fair. It gave a recorded spiel on the glories of a certain wristwatch while repeatedly dipping the wristwatch in a glass of water. Like other viewers, I could not be certain whether it could do more than that, whether it could perceive you or understand what you said, or know what you were thinking. (In hindsight, obviously, it could not.)*

## THE IMAGE

*This public uncertainty about robots is still being exploited to the hilt. Seeing today's news items about "industrial robots," many people apparently imagine that these industrial robots are walking, talking humanoids, two-legged and two-eyed, possibly wearing hardhats and chewing tobacco.*

*No. In fact the "industrial robot" is usually some sort of a grabber that positions things, of use principally on assembly lines because its operations have to be set up so carefully that there's no sense in setting it up if the motions won't be repeated over and over.*

*But these things are useful in auto assembly, in spray painting, welding, riveting and so on; because their motions are repeatable to the millimeter or better, and their strength can be prodigious. (The first accidental human death by an industrial robot occurred, I believe, in 1986.)*

*These things are controlled, of course, by computers, which memorize the motions as a series of steps and rotations, check on various sensors and respond. (Sometimes the sensors get complex, as with image recognizers, but that, too, is mostly another program.)*

*AGVs (Automatic and Guided Vehicles). In some big companies, there are automatic delivery carts that follow a cable on the floor (one model even follows an invisible line on the carpet). These will stop at predetermined locations, and honk or stop if something gets in the way. A convenience, but nothing you would call a technological breakthrough.*

*These are used as mail carts and factory delivery machines.*

*Sentry robots are the same sorts of machines with TV cameras to go peering. An operator at a desk can look around a large factory in absolute safety.*

**Promotional robots.** These are two-legged, or on wheels, and wander around the floor at conventions, typically making lascivious remarks to women.

As usual, some are fooled. But the fun is finding the person who's really controlling it. Wander through the crowd looking for someone who is watching the robot intensely with his hand in his pocket, her hand in a handbag, or the like, and possibly moving his or her lips slightly. Now that you've spotted the perpetrator the game is between you and him or her, as he or she acts casual and pretends not to be running the thing.

As yet there are no strip-tease automata (disrobots).

**Medical instruction robots**, such as those made by Ixxion. These are currently life-size dolls, or parts of them, that respond as the patient would—computer-based interactive systems with realistic humanoid input and output.

Note that in the future we may expect the continuation of these developments to lead to sexual-surrogate robots.

**The home robot.** This is a cheap version of the promotional robot, fairly useless. You can make it wander around by remote control, or at random; some come with a repertoire of prerecorded sassy remarks. My favorite was an advertised home robot named GARCAN: it was in fact a garbage can that had been put on wheels and remote control.

**Toy robots.** Well, there have been remote-controlled cars and stuff for decades. But of late there are some pretty amusing variants. Fuzzy-animal robots called "Petsters" can be programmed to explore randomly, act angry, or do various other things. "We installed one at the office and didn't see the cat for days," says a software developer I know.

**Story robots.** The latest wrinkle is the robot that tells a story. (Remember when parents told stories? Now you, too, can recapture the nostalgia of that bygone day, metempsychosed into a fuzzy doll with lips that move, for \$79.95. Story cassettes are extra.)

The champ is Teddy Ruxpin™, a storytelling bear. Not only does his mouth move with his recorded tale, but his expressiveness is enhanced by the fact that his jaw and snout are separately controlled, making unpredictable double motions that don't look half bad. (And his eyes wink.) He also has a sidekick who talks back on the tapes, and may be bought separately

and plugged into Teddy for an animated duologue.

Fast on the heels of the successful Ruxpin comes *The Talking Mother Goose*™, a goose doll with similarly double-articulated beak, and a slightly English accent. There is also a story robot that looks like a robot, and has a changeable LCD face.

**Pizza robots.** Nolan Bushnell, inventor of "Pong" and founder of Atari, went on to invent the Pizza Robot. At Chuck E. Cheese's Pizza Parlors, various characters would wave their arms and do lip-sync to speeches, songs and jokes—all controlled, incidentally, by a multitrack tape recorder.

The characters were awful and the showmanship was poor—each supposedly sing-along song was terminated by the time you could find the key—but the concept lasted long enough to generate imitators.

The best part I heard, though, was what happened when a singing-robot pizzeria went out of business in Seattle. "A bunch of artists bought the robots," a friend tells me, "and did some very strange things with them." He did not elaborate.

## BIBLIOGRAPHY

**ARMS AND EYES FOR YOUR COMPUTER!** (But just try programming them.)

**Robotics Age** magazine, and its ads, are largely concerned with moving output devices (arms, wheels, micro-manipulators and so on); languages to control them, interfaces, visual analyzers, etc.

"Is that  
the river  
that  
runs  
down to  
the  
sea?"

—James Stewart in  
"The FBI Story."

"The  
best way  
to  
predict  
the  
future is  
to invent  
it."

—Alan Kay

# XANADU™

One of the great unfinished dreams of the computer field, along with the Dynabook and the Architecture machine:

XANADU\*

(pronounced "Zanna-Doo")

Literary System, Storage Engine, Hypertext and Hypermedia Server, Virtual Document Coordinator, Write-Once Network Storage Manager, Electronic Publishing Method, Open Hypermedium, Non-Hierarchical Filing System, Linked All-Media Repository Archive, Paperless Publishing Medium, and Readdressing Software. The Magic Place of Literary Memory™.

Xanadu, friend, is my dream.

The name comes from the poem (next page); Coleridge's little story of the artistic trance (and the Person from Porlock) makes it an appropriate name for the Pleasure Dome of the creative writer. The Citizen Kane connotations, and any other connotations you may find in the poem, are side benefits.

I have been working on Xanadu, under this and other names, for fourteen years now.

Make that twenty-seven years.

Originally it was going to be a super system for handling text by computer (see pp. DM 16 and DM 29). But it grew: as I realized, level by level, how deep the problem was. ☹

\* "Xanadu," the Eternal-Flaming-X and FEBE are trade and service marks for computer, hypertext and software products and services offered by Project Xanadu, 8480 Fredericksburg #138, San Antonio, TX 78229, and licensed to XOC, Inc., a California Corporation.

For further information, see our videotape "A Technical Overview of the Xanadu Storage System" (VHS Extended Play, \$50) or Ted Nelson's book *Literary Machines* (\$20); add \$5 for purchase order, \$5 for overseas airmail.

Donations are accepted from those who would like to be on our mailing list. We also offer a universal network name or "Xandle" for \$100 (subject to such complex and arbitrary rules and possible cancellation that the payment should be considered a donation).

By special arrangement with Microsoft Press, *Computer Lib* is accessible on the prototype Xanadu System.



It is the summer of the year 1797, the Author, then in ill health, had retired to a lonely farm house between Porlock and Linton, on the Exmoor confines of Somerset and Devonshire. In consequence of a slight indisposition, an anodyne had been prescribed, from the effect of which he fell asleep in his chair at the moment that he was reading the following sentence, or words of the same substance, in "Purchase's Pilgrimage": "Here the Khan Kubla commanded a palace to be built, and a stately garden thereunto; and thus ten miles of fertile ground were inclosed with a wall. The Author continued for about three hours in a profound sleep, at least of the external senses, during which time he has the most vivid confidence, that he could not have composed less than from two to three hundred lines; if that indeed can be called composition in which all the images rose up before him as things, with a parallel production of the correspondent expressions, without any sensation or consciousness of effort. On awaking he appeared to himself have a distinct recollection of the whole, and taking his pen, ink, and paper, instantly and eagerly wrote down the lines that are here preserved. At this moment he was unfortunately called out by a person on business from Porlock, and detained by him above an hour, and on his return to the room, found, to his no small surprise and mortification, that though he still retained some vague and dim recollection of the general purport of the vision, yet, with the exception of some eight or ten scattered lines and images, all the rest had passed away like the images on the surface of a stream into which a stone had been cast, but alas! without the after restoration of the latter:

Then all the charm  
Is broken—all that phantom-world so fair,  
Vanishes and a thousand circles spread,  
And each mis-shape the other, Stay awhile,  
Poor youth! who scarcely dar'st lift up thine eyes—  
The stream will soon renew its smoothness, soon  
The visions will return! And lo! he stays,  
And soon the fragments dim of lovely forms  
Come trembling back, unite, and now once more  
The pool becomes a mirror.

As a contrast to this vision, I have annexed a fragment of a very different character, describing with equal fidelity the dream of pain and disease.—1816.

## KUBLA KHAN.

In Xanadu did Kubla Khan  
A stately pleasure-dome decree:  
Where Alph, the sacred river, ran  
Through caverns measureless to man  
Down to a sunless sea.  
So twice five miles of fertile ground  
With walls and towers were girdled round:  
And there were gardens bright with sinuous rills  
Where blossomed many an incense-bearing tree:  
And here were forests ancient as the hills,  
Enfolding sunny spots of greenery.  
But oh! that deep romantic chasm which slanted  
Down the green hill athwart a cedarn cover!  
A savage place! as holy and enchanted  
As e'er beneath a waning moon was haunted  
By woman wailing for her demon-lover!  
And from this chasm, with ceaseless turmoil seething,  
As if this earth in fast thick plants were breathing,  
A mighty fountain momently was forced:  
Amid whose swift half-intermitted burst  
Huge fragments vaulted like rebounding hail,  
Or chaffy grain beneath the thresher's flail:  
And 'mid these dancing rocks at once and ever  
It flung up momently the sacred river.  
Five miles meandering with a mazy motion  
Through wood and dale the sacred river ran,  
Then reached the caverns measureless to man,  
And sank in tumult to a lifeless ocean:  
And 'mid this tumult Kubla heard from far  
Ancestral voices prophesying war!  
The shadow of the dome of pleasure  
Floated mid way on the waves;  
Where was heard the mingled measure  
From the fountain and the caves.  
It was a miracle of rare device,  
A sunny pleasure-dome with caves of ice!  
A damsel with a dulcimer  
In a vision once I saw:  
It was an Abyssinian maid,  
And on her dulcimer she played,  
Singing of Mount Abora.  
Could I revive within me  
Her symphony and song,  
To such a deep delight 'twould win me  
That with music loud and long,  
I would build that dome in air,  
That sunny dome! those caves of ice!  
And all who heard should see them there,  
And all should cry, Beware! Beware!  
His flashing eyes, his floating hair!  
Weave a circle round him thrice,  
And close your eyes with holy dread,  
For he on honey-dew hath fed,  
And drunk the milk of Paradise.

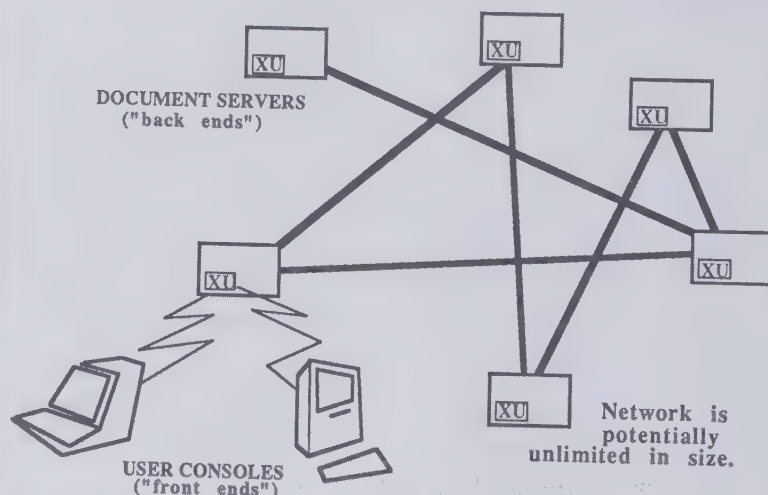
**THE XANADU™ PROJECT** is very well known; many have worked on it over the years. This visibility is both an advantage and a hindrance.

Many people know our goals and ideals, but only a few understand our real technical direction.

## A new form of storage.

In ordinary computer storage, you have to keep copying files in order to reorganize them—a tedious and dangerous process, which (when done wrong) easily wipes out your work. For every hour spent usefully, perhaps half an hour must be spent copying, rearranging and backing up files. Most do not have the patience or resignation to do this, and many desert their computers as the work deteriorates into a rubble of confused and mislaid diskfiles.

Used on your personal computer, the Xanadu system is simply a storage manager. It will enable you to organize and reorganize what is already there without complication. The same materials may be organized in many different ways, and put to many different uses without significantly expanding the storage needed. It's

A SINGLE PROGRAM,  
RUNNING THROUGHOUT A NETWORK

From Literary Machines 87.1.

People think that if the Xanadu system is going to store and supply so much stuff, it has to be an enormous program.

Not so. The same smallish program running in every computer of a network, makes the overall system. So we are much closer than generally realized.

all a unified structure, and no form of organization disturbs any previous one.

Offices need this just as much as individuals. With the advent of office computers, the blizzard of paperwork has melted into a slush of datafiles. (When an employee leaves, others dare not touch his or her data files for months, until some brave soul decides that things lost in them are no longer needed.)

The Xanadu system not only saves space—by not copying—but it shows the interconnections and commonalities among all documents. We believe this may lighten office work by as much as 75%.

We foresee new forms of education and new forms of writing, which show and make clear the interconnectedness of everything.

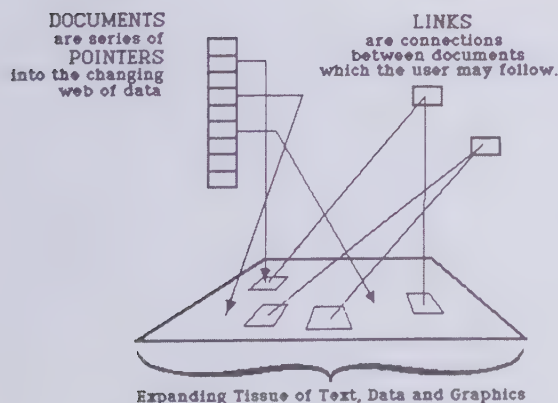
This reflects a fundamental philosophy. For those to whom the world is hierarchical, cut and dried, there is no reason to want to keep reorganizing ideas, writings, indexes, directories. But for anyone who must constantly work with ideas, writings, indexes and directories, the problem is their ever-swirling change.

#### CUMULATIVE ORDER AND COLLAGING:

The ability to build new pieces from old without crippling multiplication of files helps things to become more and more ordered—unlike ordinary personal computer use, which tends to become more and more disordered.

The same facility, extended to a network, permits new forms of pluralistic publishing.

#### IMPROVED ORGANIZATION FOR SINGLE USER OR OFFICE



From Literary Machines 87.1.

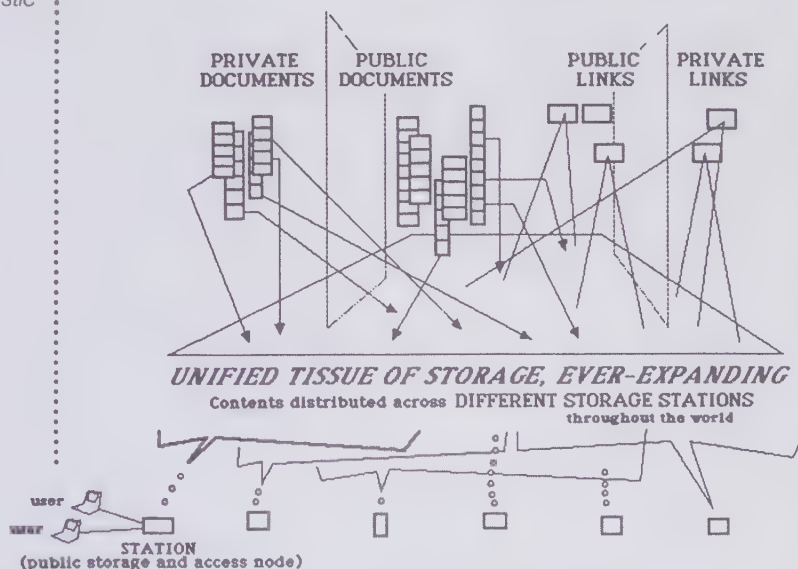
Basically, then, the Xanadu system is just one thing: a new form of interconnection for computer files—CORRESPONDING TO THE TRUE INTERCONNECTION OF IDEAS—which can be refined and elaborated into a shared network. But if you truly understand this form of interconnection, you will understand its revolutionary potential. (K. Eric Drexler of MIT calls the Xanadu program one of the most important developments of our time.)

It is this concept of "true interconnection" that is central; the underlying technicalities, while themselves embodying very new principles of software, are merely the tricks we have found for making this happen.

Thus this supposedly wild dream, what the Whole Earth Review called "a retrieval system as big as the earth," is actually within reach.

It is in reality a single computer program which exists in prototype and needs debugging and reworking.

#### PUBLIC REPOSITORY SYSTEM WITH PLURALISTIC RE-USE, publication by users



From Literary Machines 87.1.



☛ And the concept of what it was to be kept changing, as I saw more and more clearly that it had to be on a minicomputer for the home. (You can have one in your office too, if you want, but that's not what it's about.)

Now the idea is this:

To give you a screen in your home from which you can see into the world's hypertext libraries.

(The fact that the world doesn't *have* any hypertext libraries—yet—is a minor point.)

To give you a screen system that will offer high-performance computer graphics and text services at a price anyone can afford. To allow you to send and receive written messages at the Engelbart level (see p. DM 16). To allow you to explore diagrams (see pp. DM 47 and DM 134). To eliminate the absurd distinction between “teacher” and “pupil.”

To make you a part of a new electronic literature and art, where you can get all your questions answered and nobody will put you down.

Originally Xanadu was programmed around the Parallel Textface (see p. DM 41). But as the requirements of the Parallel Textface were better and better understood, Xanadu became a more general underlying system for all forms of interactive graphic environments. Thus in its final form, now being debugged, it will support not only the Parallel Textface (see p. DM 41), the Walking Net (see p. DM 47), Stretchtext (see p. DM 134), Zoom Maps (see p. DM 134) and so on, but indeed any data structure that needs to combine complex linkages with fast access and rapid changes. Because the data structure is recursively extensible, it will permit hypertext (see p. DM 29) of any depth and complexity, and the collateral

linkage (see p. DM 50) of any objects of contemplation.

### THE REDESIGN, UP AT LAST

*The reason that the Xanadu system has taken so long (and been prematurely announced from time to time) is that it's a hard problem.*

*When Computer Lib was first written, there was already a good design using methods unknown in the literature. It was to be written in assembler on the PDP-11.*

*However, a new invention by William F. Barus promised a whole new beginning: linkage not just between beginnings of chunks, but between spans of bytes. Meaning that links could survive editorial changes very cleanly.*

*Brought in by Computer Lib, a new team (now under Roger Gregory) put an experimental version of the system on the end of a phone-line for public access in January of 1987.*

*We welcome collaborating developers. Those interested in experimenting with the present system with an eye toward application and front-end projects should contact Project Xanadu.*

### THE XANADU NETWORK

First of all, bear in mind that Xanadu is a unified system for complex data management and display. This basically means that the *same* system (without the displays) can serve as a feeder machine for the data network itself.

So far, so good. That means that we can have a *minicomputer* network handling the entire structure: sending out library materials to users on call, and storing any materials they want saved. This saves all kinds of hassles with big computer and big-computer-style programming.

But who will pay for it? To build the kind of capacity we're talking about—all those disks, all those minicomputers in a network—won't it take immense amounts of capital? Now, people ask me, will any American company ever back such a Utopian scheme?

Aha.

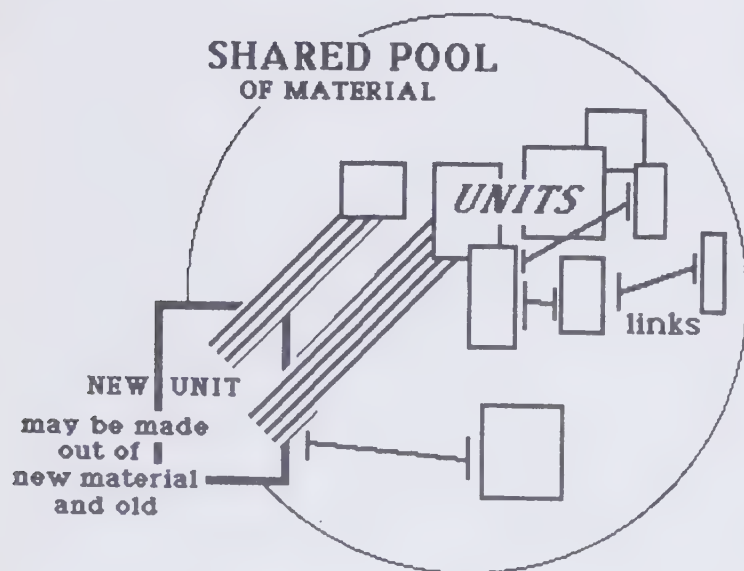
One method of financing has proven itself in the postwar suburban era, this time of drive-ins and hamburger stands. Franchising.

What I propose, then, is the Mom-and-Pop Xanadu Shop. Or, more properly, the Xanadu *stand*. “Mom and Pop” are the owners of the individual stand. But the customers can be families, too.

From far away the children see the tall golden X's. “Oh, Daddy, can't we stop? I want to play Spacewar,” says little Johnny. Big Sis adds, “You know, I have to check something for my paper on Roman politics.” And Mom says, “Say, that would be a good place for lunch.”

So they turn in past the sign that says “OVER 2 BILLION SCREEN HOURS,” and pull into the lot. They park the car, and Dad shows the clerk his Xanadu credit card, and the kids run to

## XANALOGICAL STORAGE



From Literary Machines 87.1.

The Xanadu system is built around a new concept of storage: that new units—we call them “documents”—may be freely built out of material in old units, and linked together arbitrarily.

## 2020 Vision And The Near Sight

In the year 2020, we imagine a network with at least a hundred million simultaneous users, adding a hundred million documents an hour to the system.

The immediate use, however, is as a clarification box: to make the interconnections of files, documents, correspondence, lists and figures clear at last in computer storage—and keep them that way.

screens. Dad and Mom wait for a big horizontal CRT, though, because there are some memories they'd like to share together...

Sis's paper, of course, goes to her teacher through *his* Xanadu console.

## THE PLAN. IS IT AS CRAZY AS IT SEEMS?

Deep inside, the public wants it, but people who think of computers in clichés can't comprehend it. This means "the public" must somehow create it.

One way to go is to start a new corporation, register it with the SEC and try to raise a lot of money by selling stock publicly. Unfortunately there are all kinds of obstacles for that.

Through the miracle of franchising, now, a lot of the difficulties of conventional backing can be bypassed. The franchisee has to put up the money for the computers, the scopes, the adorable purple enamel building, the Johns and so on; as a Xanadu franchisee he gets the whole turnkey system and certain responsibilities in the OVERALL XANADU NETWORK—of which he is a member. He is assigned permanent storage of certain classes of materials, on call from elsewhere in the net. (Naturally, everything is stored in more than one place.)

The Xanadu subscriber, of course, gets what he\* requests at the screen as quickly as possible—or on priority if he wants to pay for it—and may store his own files, including linkages among other materials and marginal notations to other things that can be called. (See collateral structures, p. DM 50; these can automatically bring forth anything they're linked to. (See "Nelson's Canons," p. DM 149)) A user's historical record will be stored to whatever degree he desires, but not (if he chooses) in ways that can be identified with him.

Home users need only dial a local phone number—their nearest Xanadu stand—to connect with the entire Xanadu network. (The cost of using something stored on the network has nothing to do with where it is stored.)

(Special high-capacity lines need not be installed between storage stations, as

appropriate digital transmission services are becoming available commercially.)

Various security techniques prevent others from reading a subscriber's files, even if they sign on falsely; the Dartmouth technique of scrambling on non-stored keywords is a good one.

The Xanadu stand also has private rooms with multiple screens, which can be rented for parties, business meetings, design sessions, briefings, legal consultations, lectures, seances, musicales, and so on.

The choice locations for the Xanadu stands are somewhat different from hamburger spots. But that's probably not anything to go into here.

Within the Xanadu network, then, people may read, write, send messages, study and play.

*Note: THIS IS NOT IN ANY WAY AN OFFER TO SELL FRANCHISES. Such an offer will only be made at such time as the experimental Xanadu program appears to be functioning and networking*

## THE FIRST COMPUTER TALKIN' SINGING COMMERCIAL, continued from p. 15

So continuing under our guidance inertial,  
Let's have the XANADU SINGING  
COMMERCIAL.

[strings] It's got everything to give.

It'll get you where you live.

[chimes] Realms of mind that you may  
roam:

Grasp them all within your home.

[brass flourish] The greatest things you've  
ever seen

Dance your wishes on the screen.

[bass bautant] All the things that man has  
known

Comin' on the telephone—

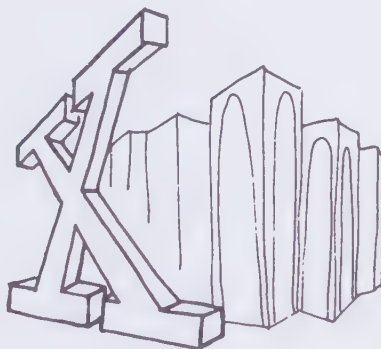
[TUBULAR BELLS!] Poems, books and pic-  
tures too

COMIN' ON THE XANADU—

[kettledrums] XAN-A-DU, OO—

THE—WORLD—OF—YOUUUU!

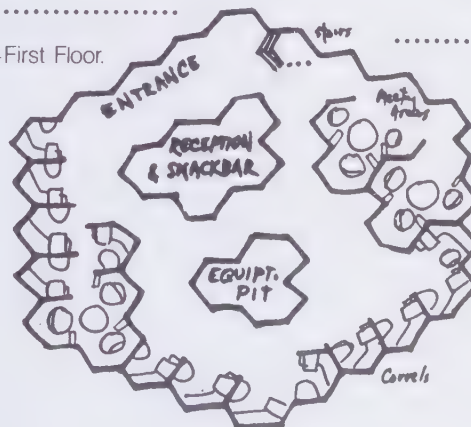
The Golden X's welcome the mind-  
hungry traveller.



View from the snack balcony of a  
large Xanadu installation, overlooking  
the internal greenery. Hexagonal archi-  
tecture permits physical expan-  
sion without interruption of services.  
(The mollusks have been telling us  
something about expansion.)



Local Xanadu Stand—First Floor.



\*Please read "he or she" and "him or her." This book was written back when the masculine included the feminine, or so we were taught in earlier days.



correctly, and only to sophisticated individuals aware and fully warned of its unusual business risks and unproven technology, who can afford to lose all or part of their investments, and who are prepared personally to work long and difficult hours on the maintenance and supervision of a complex and unusual facility.

#### XANADU: BRASS TAX

WHAT IT IS: the heart of the Xanadu system, now being debugged, consists of

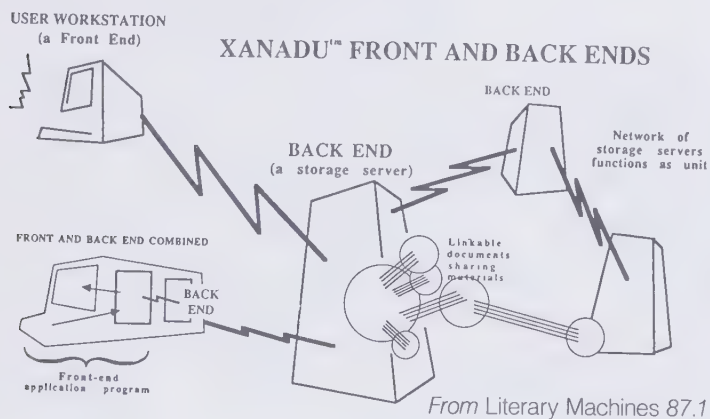
a highly integrated program for use on minicomputers ("software"—see p. 52). It is an operating system with two programs: a highly generalized data management system for handling extremely complex data in huge files, and a generalized display system, married to the other, for handling branching animation and retrieval and canned display programs. These ordain retrievals by the data system. The Parallel Textface (see p. DM 41) and the Walking Net (see p. DM 47) are two such canned programs.

(See below, "Front And Back End.")

## "Front and Back End" "FRONT AND BACK END"

*The Xanadu system has become basically a storage management program for networks. This is the back end. Xanadu storage remains standard across all applications. Indeed, we would like it to become an archival standard for all data structures, permitting anybody's programs to use the shared data freely.*

*Anything you want to do with material stored in the Xanadu system is done by a program in your computer or workstation. This is your front end.*



*Many different front-end programs are possible for different possible uses of Xanadu storage. We invite software developers of every persuasion to write programs for any legitimate purpose to make use of these new storage forms and search mechanisms.*

## A TECHNICAL DESCRIPTION OF THE XANADU SYSTEM

© 1987 Project Xanadu

*The Xanadu system—more properly, the Xanadu network storage engine—is a file-server program for linked compound documents. It is designed to run on a network and manage the update and delivery of document fragments on demand (not just full documents) from anywhere on the network. It operates in write-once mode and thus is especially suited to optical disk storage.*

*All documents are virtual, not stored in finished form (since there can be many "finished forms"), and assembled on demand as their pieces are requested. Material may be shared between documents with complete information available on its origin and other documents in which the same material is used.*

*Users requesting parts of these virtual documents need not be aware (but may find out) what documents their fragments originally came from. A request for a long piece of text from the user's workstation may bring back a shower of fragments from the separate documents of origin on different storage machines, which are assembled by the user's front-end program.*

*Links (separate from text and other data) are used for all demarcation, permitting data to remain uncluttered by special codes. Thus one use of material does not corrupt it for other uses. (For instance, paragraph marks and text attributes are stored as links, so the same text may be marked up many different ways.) Link types are open-ended. Searches of the link data are a specific class of user interaction. Any link to material shared between documents is available to all instances of that shared material.*

*With respect to any document or fragment, a user may request all other documents linked from it, all other documents linking to it, and all other documents presently containing portions of it.*

*The back-end program presently runs on a Sun Workstation under UNIX. It is written in C and presently (May '87) compiles to about 137K of 68000 native code on the Sun. This does not include buffer space, of which the more the merrier (1 megabyte and up recommended).*

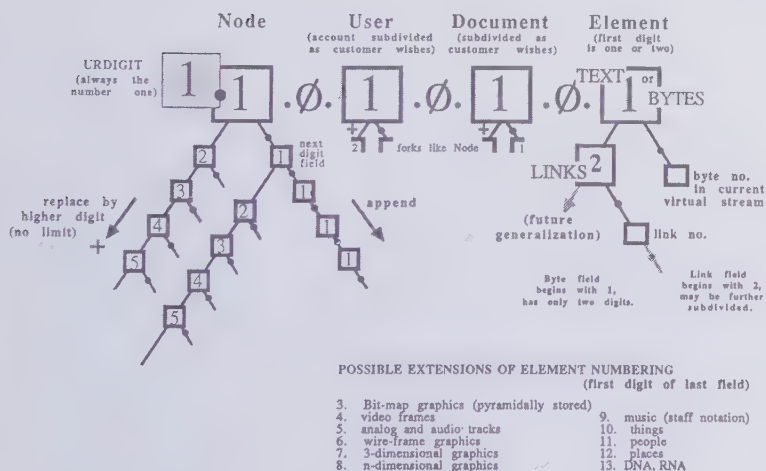
*Front-end programs should include our protocol manager, a module which handles sending and receiving in*

the FEBE™ Front End-Back End protocol. It presently compiles to about 30K on the Sun.

The user operates from a workstation program set up to take advantage of the back end's special facilities. All operations which are not part of the file service are delegated to these workstation programs; these are "front-end functions." Such front-end functions include making book-marks, keeping track of personal actions at the workstation, etc. Certain audit trailing of each document's modification is maintained automatically. Users may be easily informed about update of a document by various versioning features.

This design has been built with on-line network publishing as a goal. It will be possible to publish documents, links, or associated material on-line with royalties to their owners. These will be fragmentary royalties, automatically paid by the requesting user. Whenever a fragment is sent, a small financial increment will be credited for each byte shipped to the originating owner of that byte.

## TUMBLER ADDRESS MASTER DIAGRAM

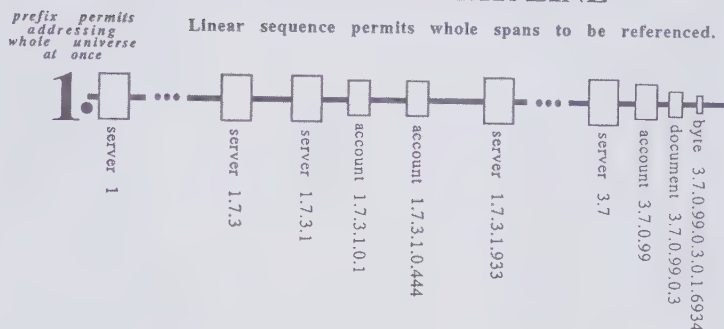


From Literary Machines 87.1.

The Xanadu network address is called a tumbler, which specifies the network node, user, document and element within that document. (A "user" may be a company or a department.) These fields may be subdivided in various ways. All tumblers may be mapped to a sequence called the tumbler line.

A WRITE-ONCE UNIVERSAL LINEAR ADDRESS SPACE

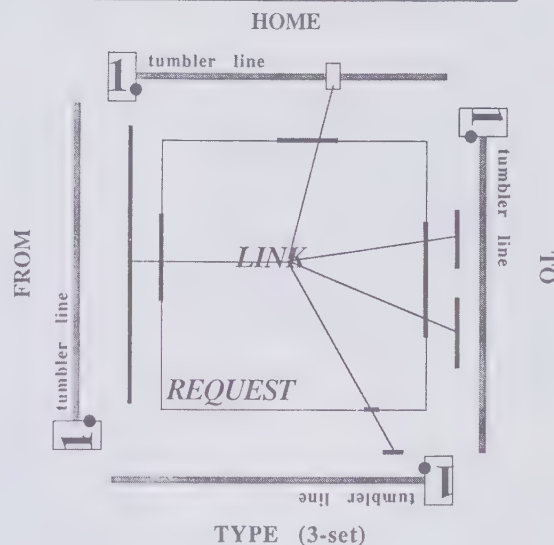
## THE TUMBLER LINE



From Literary Machines 87.1

Four tumbler lines may be arrayed into a tumbler square, useful for visualizing both links and search requests. A link has a home in which it resides (the original copy of the link is in its native home); it has a from-set, the spans of bytes from which it links; a to-set, the spans of bytes to which it links; and a type (called for symmetry a three-set), which is some span of the tumbler line which has been assigned an arbitrary "type" meaning. A link request is that part of a FEBE command specifying the spans covering the home-sets, from-sets, to-sets and three-sets to be searched. (This might be a request for a list of links, or for the material actually connected to them.) This too may be mapped to the tumbler square.

LINK AND REQUEST-SET BOTH MAP TO THE TUMBLER SQUARE



From Literary Machines 87.1

We might think of the system as a database of links connecting addresses in virtual documents. But what makes the problem (and this view) difficult is that all documents remain modifiable even while the links remain correctly attached to their shifting parts.

We expect to extend these linking facilities to all types of data structure; 2D and 3D graphical structures are of particular importance.

These matters are explicated more comfortably and fully in Literary Machines.



Xanadu is self-networking, two on the phone make a network, and more can join.

Important features of the data system are huge addressability (in the trillions of elements) and Virtual Blocklessness.

COMPATIBILITY: because of its highly compacted and unconventional structure, it is *not* compatible with other operating systems (including time-sharing). Anyway, to put it on a larger machine is like having your Mazda driven around in a truck.

LANGUAGES: Xanadu programs will not be made available in any higher languages, mainly because of their proprietary character. The purpose of Xanadu is to furnish the user with un-computerish good-guy systems for specific purposes, not a chance to do his own programming.

STANDARDIZATION. Taking a lesson from the integrated work of various people whose work has been described in this book, we see that if you want a thing done *right*, you have to do it yourself. (Great Ideas of Western Man: one of a series.)

Several levels of standardization are important with Xanadu. One, all Xanadu

systems must be able to work with all Xanadu files (except for possible variations in screen performance and size of local memory). Now, there are those who would *not* be concerned for this sort of universality, and who might even try to make sure systems were *incompatible*, so that you had to buy accessories and conversion kits up and down the line. That is one of the things that must be avoided: "partial" compatibility, subject to expensive options and conditions, a well-known technique in the field.

Second, all Xanadu systems should be able to work with outside systems either through or off the net, *if* they conform to the unusual data rules required by the unusual design of the system. This assures that Xanadu systems will be compatible with any other popular networks. It also assures others who want to offer Xanadu-class services to system owners (through, e.g., conventional time-sharing) that if they adhere to the rules (see "Canons," p. DM 149) they can play the game on a certified basis.

By stabilizing the "Xanadu" trademark, I hope to prevent such shenanigans. Thus every accredited Xanadu system will offer full compatibility with

the data structure, and either full performance or substitutes as necessitated by the hardware. The "Xanadu" trademark can in principle be made available to manufacturers abiding by all design features of the system.

AHEM. There is a lot to talk about, but a lot of time can be wasted talking. It is suggested that thoughtful computer firms, interested in some form of participation, study this book carefully—at least enough so no one's time need be wasted.

*We're no longer setting dates, but it's going to happen.*

*Whether the Xanadu system will be available first as a service, or as a complete package for single-user personal computers, depends on whether we get backing in the near term.*

So, fans, that about wraps it up. I'll be interested in hearing from people who *want* this system; many hardheaded business people have told me nobody will. Prove 'em wrong, America!

Of course, if hyper-media aren't the greatest thing since the printing press, this whole project falls flat on its face. But it is hard for me to conceive that they will not be.

I would like to thank (in chronological order) Elliot Klugman, Nat ("Kubla") Kuhn, Glenn Babecki, Cal Daniels and John V.E. Ridgway for the considerable time and involvement they gave to the Xanadu program design sessions; thanks also to various others who sat in from time to time.

*The official credit line for the present system is: Theodor H. Nelson, William F. Barus, Mark S. Miller, Roger Gregory, Stuart Greene, Eric Hill, Roland King, John V.E. Ridgway and Calvin Daniels.*

## BIBLIOGRAPHY

Theodor Holm Nelson, *Literary Machines*, edition 87.1, published by the author.

K. Eric Drexler, *Engines of Creation*, Anchor/Doubleday, 1986.

Howard Rheingold, *Tools for Thought*, Simon & Schuster, 1985.

"Nelson's the Name, and What He Proposes Could Outdo Engelbart." *Electronics*, 24 Nov '69, 97.

Is Xanadu worth waiting for?  
That depends, doesn't it,  
on the value of the hand-  
bush differential bird utility  
ratio.

Clearing up the problem of computer files, dis-  
solving file boundaries and allowing their free  
connection and interpenetration, will lead the way  
to both the office and library of the future.  
And keeping track of things in open-ended and  
evolving categories—just the opposite of the  
"computerish" stereotype—is the most important  
thing that personal software needs.

Tell the  
world  
we're not  
just  
ducklings  
that  
followed  
an ice-  
cream  
truck.

—Michael Butler,  
Project Xanadu

**Q. How many Xanadu people  
does it take to change a light  
bulb?**

**A. None, that's a front-end  
function.**

# FLIP OUT

I have had a most rare vision. I have had a dream, past the wit of man to say what dream it was: man is but an ass. if he go about to expound this dream. Methought I was—there is no man can tell what. Methought I was,—and methought I had,—but man is but a patched fool, if he will offer to say what methought I had. The eye of man hath not heard, the ear of man hath not seen, man's hand is not able to taste. his tongue to conceive. nor his heart to report, what my dream was.

—Bottom the Weaver in "Midsummer Night's Dream"

*So there you are, friends. The real issues do not change.*

*By presenting this book as a magazine-collage, rather than something formal or official-looking or pompous, I got your attention, and may have even gotten the ideas across more effectively than in linear style.*

*Which is what tomorrow's hypermedia will do, which is much of the point of the book.*

Now you see why I brought you here. This Gem-maniacal book has, obviously, been created as a crossroad of several cross purposes: to furnish a needed, grabby layman's introduction to two vast but rather inaccessible realms; to present a coherent, if contentious, point of view, and unroll a particular sort of apocalyptic vision after preparing the vocabulary for it; to make bright friends and informed supporters for my outlook and projects; to get home to some of my friends the fact that what I am doing is at bottom *not* technical; and finally, if nothing else, to set forth some principles about the way

things *should* be, which others will have to answer if they propose to do less. Thus, overall, this book is a message in a Klein bottle, waiting to see who's thirsty.

I suppose it all started in college. Swarthmore left me with an exaggerated notion of the extent to which ideas are valued in the academic world; it took two graduate schools to clear this up. After that, as far as I was concerned, Ph.D. stood for Poophead. But I still cared about ideas, and the deep necessity of

finding their true structure and organization. From writing I knew the grueling difficulty of trying to make ideas get in order. I believed in the pure, white light of inspiration and the power of the naive but clever mind to figure out anything, if not obstructed but dumb dogmas and obtuse mental schemata fostered by the educational system.

When I finally got the idea of what computers were about, sometime in 1960, I took endless walks at night trying

**Q.** Aren't you afraid that writing a flippant book will keep people from taking you seriously?

**A.** I do not want to be taken seriously in some quarters until it's too late.  
*That part worked a bit too well.*

## Nelson's Canons

### ■ A Bill Of Information Rights

It is essential to state these firmly and publicly, because you are going to see a lot of systems in the near future that purport to be the last-word cat's-pajama systems to bring you "all the information you need, anytime, anywhere." Unless you have thought about it you may be snowed by systems which are inherently and deeply limiting. Here are some of the things which I think we will all want. (The salesman for the other system will say they are impossible, or "We don't know how to do that yet," the standard putdown. But these things *are* possible, if we design them in from the bottom up; and there are many different valid approaches which could bring these things into being.)

These are rules, derived from common sense and uncommon concern, about what people can and should have in general screen systems, systems to read from.

#### 1. EASY AND ARBITRARY FRONT ENDS.

The "front end" of a system—that is, the program that creates the presentations for the user and interacts with

him—must be clear and simple for people to use and understand.

**THE TEN-MINUTE RULE.** Any system which cannot be well taught to a layman in *ten minutes*, by a tutor in the presence of a responding setup, is too complicated. This may sound far too stringent; I think not. Rich and powerful systems may be given front ends which are nonetheless ridiculously clear; this is a design problem of the foremost importance.

**TEXT MUST MOVE**, that is, slide on the screen when the user steps forward or backward within the text he is reading. The alternative, to clear the screen and lay out a new presentation, is baffling to the eye and thoroughly disorienting, even with practice.

Many computer people do not yet understand the necessity of this. The problem is that if the screen is cleared, and something new then appears on it, there is no visual way to tell where the new thing came from: sequence and structure become baffling. Having it slide on the screen allows you to understand where you've been and where you're going; a feeling you also get from turning pages of a book. (Some close substitutes may be possible on some types of screen.)

On front ends supplied for normal users, there must be *no* explicit computer languages requiring input control strings, no visible esoteric symbols. Graphical control structures



to hash these things out and see where they led. The text systems came clear to me, at least in their beginnings, in a few weeks; the realization that 3D halftone was possible came to me as a shock the following spring, I believe as I was walking across Radcliffe Common. Since then trying to build these systems for creation and the true ordering of intricate thought has been my driving dream.

My own life among these dream machines has been thoroughly unpleasant, and if people are right in telling me that nobody wants systems like the ones I am designing, I'll get the heck out of this.

I first got into this as a writer; all I wanted was a decent writing system that would run on a computer. Little did I realize the immensity of what that entailed, or that for some reason my work and approach would engender indignation and anger wherever I went. There is a fiction

that everybody in these fields is doing something fundamentally scientific and technical, and this fiction is usually upheld in carefully enacted mutual playlets. Trying to cut through that and say, "Let's build a home for mankind that will at last be shaped to fit man's mind," does not seem to generate immediate warmth and welcome.

As far as I know, there is *still* not a Decent Writing System anywhere in the world, although several things now come close. It seems a shame that grown men and women have to rustle around in piles of paper, like squirrels looking for acorns, in search of the phrases and ideas they themselves have generated. The decent writing system, as I see it, will actually be much more: it will help us create better things in a fraction of the time, but also keep track of everything in better and more subtle ways than we ever could be-

fore. But nobody sees this—I suppose it's only writers and editors that know they're trying to "keep track of ideas"—and I have been unable to get this across to *anybody*. (The professional writers, of course, won't talk to me either.)

And, to lighten the burden, I've finally given up on trying to reach professionals, who evidently need a thick gravy of technicalism to make the obvious palatable; with this bookity I am taking my case to The People. It is there, anyway, out in Consumerdom, that the real action is going to occur. So the important thing is for everybody to know what's really possible, and what they *could* have. That is why I have shot off my big canons (and this epistol).

To me, you see, this is really a holy crusade, whereas I know guys to whom it's just a living. It's no less than a question of freedom in our time. The cases of Sol-

zhenitsyn and Ellsberg remind us that freedom is still not what it should be, anywhere. Computer display and storage can bring us a whole new literature, the uniting and the apotheosis of the old *and* the new; but there are many who would not necessarily want to see this come about. Deep and widespread computer systems would be tempting to two dangerous parties, "organized crime" and the Executive branch of the Federal government (assuming there is still a difference between the two). If we are to have the freedoms of information we deserve as a free people, the safeguards have to be built in *at the bottom, now*. And the opulence which is possible must be made clear to everyone *before* we settle on an inferior system—as we did with television.

Some people have called my ideas and systems "Orwellian." This is annoying in two ways. In the first place it suggests the

having clarity and safety, or very clear task-oriented keyboards, are among the prime alternatives.

All operations must be fail-safe.

Arbitrary front ends must be attachable: since we are talking about reading from text, or text-and-picture complexes, stored on a large data system, the presentational front end must be separable from the data services provided further down in the system, so the user may attach his own front-end system, having his own style of operation and his own private conveniences for roving, editing and other forms of work or play at the screen.

## 2. SMOOTH AND RAPID DATA ACCESS.

The system must be built to make possible fast and arbitrary access to a potentially huge data base, allowing extremely large files (at least into the billions of characters). However, the system should be contrived to allow you to read forward, back or across links without substantial hesitation. Such access must be implicit, not requiring knowledge of where things are physically stored or what the internal file names may happen to be. File divisions must be invisible to the user in all his roving operations (FREEDOM OF ROVING): boundaries must be invisible in the final presentations, and the user must not need to know about them.

## 3. RICH DATA FACILITIES.

Arbitrary linkages must be possible between portions of text, or text and pictures; annotation of *anything* must be

provided for; collateration (see p. DM 50) should be a standard facility, *between any pair of well-defined objects*; PLACEMARK facilities must be allowed to drop anchor at, or in, anything. These features imply private annotations to publicly-accessible materials as a standard automatic service mode.

## 4. RICH DATA SERVICES BASED ON THESE STRUCTURES.

The user must be allowed multiple rovers (movable placemarks at points of current activity); making possible, especially, multiple windows (to the location of each rover) with displays of collateral links.

The system should also have provision for high-level moot-ing and the automatic keeping of historical trails.

Then, a complex of certain very necessary and very powerful facilities based on these things, viz.:

A. ANTHOLOGICAL FREEDOM: the user must be able to combine easily anything he finds into an "anthology," a rovable collection of these materials having the structure he wants. The linkage information for such anthologies must be separately transportable and passable between users.

B. STEP-OUT WINDOWING: from a place in such an anthology, the user must be able to step out of the anthology and into the previous context of the material. For instance, if he has just read a quotation, he should be able to have the present anthological context dissolve around the quotation (while it stays on the screen), and the original context



nightmare of Orwell's book *Nineteen Eighty-Four*, which obviously I want no part of. (But hey, do you remember what that world of 1984 was actually like? The cryptic wars against unseen enemies that kept shifting? The government spying? The use of language to twist and manipulate? To paraphrase Huey Long: "Of course we'll have 1984 in America. Only we'll call it 1972.")

The second reason the term "Orwellian" is offensive is that it somehow reduces the life of Orwell, the man, to the world of "1984." This is a shallow and shabby thing to do to a man who spent his life unmasking oppressiveness in human institutions everywhere.

In the larger sense, then—in homage to that simple, honest, angry man, who cared about nothing more than human freedom—I would be proud indeed if my systems could be called Orwellian.

That reminds me. Nowhere in the book have I defined the phrase "computer lib." By Computer Lib I mean simply: making people freer through computers. That's all.

Fantically—or fanatically—  
Yours for a better world,  
Before we have to settle for Any—



reappear around it. The need of this in scholarship should be obvious.

**C. DISANTHOLOGICAL FREEDOM:** the user must be able to step out of an anthology in such a way and *not* return if he chooses. (This has important implications for what must really be happening in the file structure.)

Earlier versions of public documents must be retained, as users will have linked to them.

However, where possible, linkages must also be able to survive revisions of one or both objects.

## 5. FREEDOM FROM SPYING AND SABOTAGE.

The assumption must be made at the outset of a wicked and malevolent governmental authority. If such a situation does not develop, well and good; if it does, the system will have a few minimal safeguards built in.

**FREEDOM FROM BEING MONITORED.** The use of pseudonyms and dummy accounts by individuals, as well as the omission of certain record-keeping by the system program, are necessary here. File retention under dummy accounts is also required.

Because of the danger of file sabotage, and the private at-home retention by individuals of files that also exist on public systems, it is necessary to have **FIDUCIAL SYSTEMS FOR TELLING WHICH VERSION IS AUTHENTIC**. The doctoring of on-line documents, the rewriting of history—cf. both Winston

## Rebeginning: As To Spades And Emperors

A spade is a spade. The emperor has no clothes.

The computer world is an atrocious tangle of excellent incompatible pieces, well-intentioned incompatible junk, and inexcusable incompatible junk.

We have to end this chaos.

We have to re-unite the things that should never have been separate.

We have to make it work for everybody. It is time indeed for real computer liberation.

Smith's continuous revision of the encyclopedia in *Nineteen Eighty-Four* and E. Howard Hunt's forging of historical telegrams for "The White House"—is a constant danger. Thus our systems must have a number of complex provisions for verification of falsification, especially the creation of multilevel fiducials (parity systems), and their storage in a variety of places. These fiducials must be localizable and separate to small parts of files.

(Note: these are now called "authentication systems;" very sophisticated ones exist, and the government is trying to suppress them. See p. 163.)

## 6. COPYRIGHT.

Copyright must of course be retained, but a universal flexible rule has to be worked out, permitting material to be transmitted and copied under specific circumstances for the payment of a royalty fee, surcharged on top of your other expenses in using the system.

## BIBLIOGRAPHY

Theodor H. Nelson, "Computopia and Cybercrud," in Roger Levien (ed.), *Computers in Instruction* (Rand Corporation, 1971).

Theodor H. Nelson, "A Conceptual Framework for Man-Machine Everything." Proc. NCC '73.











## A COMPUTER CULT CLASSIC RETURNS!

When COMPUTER LIB was published in 1974, it quickly became the first cult book of the computer generation. Embraced by hackers and quoted by the press, it inspired thousands. Written more than 10 years ago, COMPUTER LIB predicted the major issues of *today*—design of easy-to-use computer systems, image synthesis, artificial intelligence, computer-assisted instruction, even information-storage methods now seen in CD ROM.

## REVISED AND UPDATED!

In addition to Ted Nelson's original thoughts and analyses, you'll find plenty of new material in this edition, including commentaries, insights, updates, and reconsiderations. All written in Nelson's characteristically opinionated, startling, uplifting, and informative style. Now, more than ever, this is a book whose time has come.

"In 1975, I stole a copy of Ted Nelson's COMPUTER LIB from my boss's desk. Luckily for me, the book was so damn good that my boss—Ed Roberts, the father of personal computing—could understand why I did it. He threatened to kill me, but he didn't fire me. It was a great, great book then and the world is truly blessed that Ted Nelson has updated it."

David Bunnell,  
Marketing and Advertising Manager  
of MITS in 1975, currently Chairman  
and Editor-in-chief of *PC World*,  
*Macworld*, and *Publish!*

"Well, up on some magnificent Himalayan mountaintop, next to the cerebral left or right lobe, where a few men and women get to look down over human affairs and can predict what's going to happen—we put Galileo, Tom Jefferson, Tom Paine, and Buckminster Fuller. There for 13 years, and maybe a thousand, has been Ted Nelson."

Timothy Leary, Ph.D.

"COMPUTER LIB explained, for the first time, not only what made computers tick, but how computers should and must be used; a landmark of computer literature."

Cary Lu,  
Author of *The Apple Macintosh Book*

"Ted succeeded with COMPUTER LIB to rally a rabble of latent crackpots into an anarchistic army which breached the sanctum of Official Computerdom and brought computers to everyone. And the fun's just begun!"

Lee Felsenstein,  
Designer of the Osborne 1

\$18.95

ISBN 0-914845-49-7



DREAM MACHINES

This is the flip side of *Computer Lib*.

