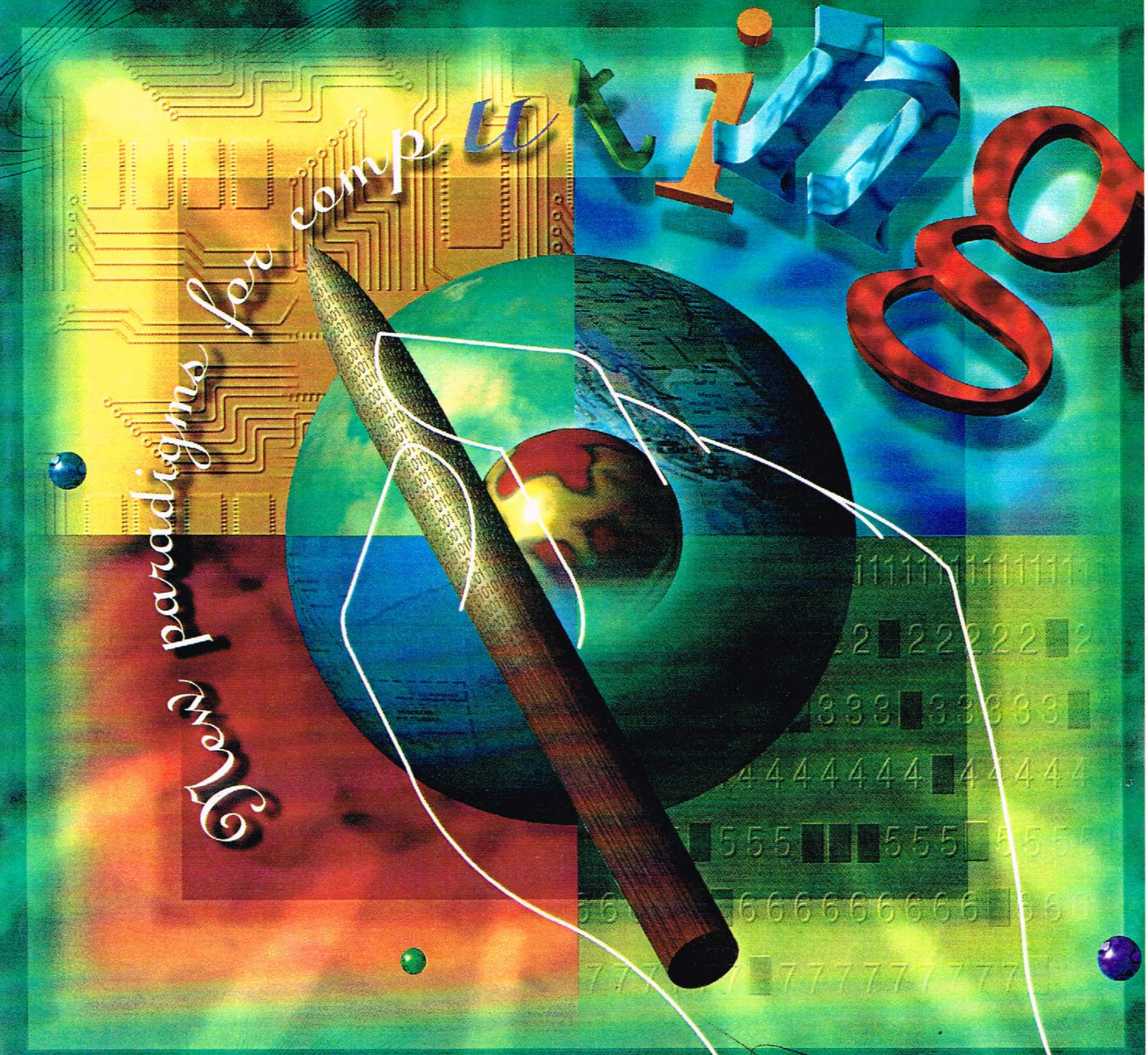
August 1996–Volume 39, Number 8

COMMUNICATIONS of the ACM



CDA Defeat: A Victory for Netizens

The Anti-Mac Interface

Learning via Computational Media

Ted Selker, Guest Editor

New Paradigms *for* Computing

MATSU

As researchers and product designers, we are colleagues as well as competitors, creating systems that will support people in their work. But striking new paradigms, styles of working that could shape the whole field, are usually "in the air" long before they mature and become accepted. We are often working on parts of what will turn out to be a shared vision. A wonderful new tool is envisioned: some try to create it, some find it not useful, and some improve it. The cycle repeats.

How do we as an informal community of innovators choose our projects? Which paradigms for using computers succeed? How do we nurture emerging paradigms before they are ready for widespread acceptance?

Over the last three years I have had the honor of hosting a workshop at IBM's Almaden Research Center entitled "New Paradigms For Using Computers" that addresses these issues. For these meetings, we have invited industry pioneers whose efforts have created many of the new ways we use and think about computers. These professionals, and many others, have been the trailblazers who have created an industry and its ability to change the way people do things. In this small set of articles, we really can't do justice to this topic, but instead hope to offer glimpses into the creation of new ways of using computers.

We begin this section with a short commentary from Xerox PARC's vice president John Seely Brown, a pioneer in AI and education who nurtures and is guru to Xerox PARC—the everpresent foundry of new paradigms in computing interfaces. Brown's piece should not leave us with the feeling that technology advancement is strictly evolutionary. While he cautions us that it is hard to visualize too far from what exists, I am happy for the non-incremental approach and legacy from the actual work that Brown's PARC represents.

To Dream the Invisible Dream

WHEN it is written, the history of computers will, I believe, be quite simple. In the beginning was the computer. Then it disappeared. Of course, it didn't go away completely. It just dissolved. Either it became part of the physical background, forming part of ordinary objects such as tables, chairs, walls, and desks. Or it became part of the social background, providing just another part of the context of work.

Indeed, this second phase of the history of computing is already under way. The modern car is really a four-wheel computational platform. Yet I'm rarely made aware of this when I drive it. Furthermore, when I go to the automotive showroom, I don't have to ask what operating system or presentation manager the car uses. Here, at least, computers have finally gotten out of the way.

The field of human-computer interaction is really configured around this central paradox. Designers struggle to produce simplicity out of complexity, direct connectivity out of mediation. Instead of drawing attention to itself, the best design lets us reach through computers into the world, allowing us to focus on creating value, not manipulating tools. So, for example, in panic stops and radical curves, the computational power in my car doesn't add to my problems by drawing attention to itself. Instead, it invisibly helps connect me to the road and the world outside.

From this perspective, I see the new paradigms for design and use developing hand-in-hand. As they adapt to current practice, new technologies become less visible. Yet, simultaneously, by adopting these new technologies, current practice continuously evolves.

Clearly, this is not a view of radical transformation. We all love to be radical and to pursue radically new ideas. But our experience shows that fundamentally new technologies seldom get adopted in a discontinuous fashion. When put to real use doing real work, new inventions almost always miss their mark, no matter how many tests ran in the lab.

Adaptation and adoption require extensive fine-tuning in the real world. The passage from the Lisa to the Mac is a famous example of this. At the same time, the mistakes Apple made in over-hyping the immediate potential of the Newton show how hard it is for any of us to learn this lesson. Nevertheless, I think we all need to learn it. Instead of focusing all our attention on radical transformation, we should try to understand the dynamics of "radical incrementalism." This is what turns radical invention into innovation.

—John Seely Brown

Nolan Bushnell, the founder of Atari who is often credited as the creator of the computer game industry, explores the "Relationships between Fun and the Computer Business." In his article he reminisces, assesses, and projects the effects from the game industry to the computer industry as a whole. Bushnell's article makes tangible some of the incredible distance we've come since the 1970s—from a world where manufacturers didn't take cathode-ray tube use in the computer industry seriously to a time when most two-year-old children know about computers. Bushnell's way is to mix lessons learned from the game industry with some as-yet unexplored visions of the way we will play.

Henry Lieberman's work has been that of creating programming and idea presentation environments using his favorite tools of creation objects (his work on actors), dynamic languages (his work on garbage collection), and interactive environments (his work on programming by example). Lieberman's article, "Intelligent Graphics," focuses on the inventive serendipity that has and can come out of the relationship between AI and graphics—an idea promoted by yet another pioneer, Muriel Cooper. He also briefly discusses programming by example, a style of interaction that he has been working to demonstrate and promote over the last couple of decades.

Ken Kahn also cares deeply about programming languages, having devoted his research career to creating Prolog interpreters, parallel logic systems, and logic-oriented systems. In his article, we see him engaged in his first love, that of creating visual approaches for thinking about programs. Watching programs execute, seeing their structure, seeing their function, organizing them—all with the underlying goal of making programming as easy as child's play.

Finally, as you look to my article, you see me not working or talking about any one idea as I could, but giving a review with some context that describes many of the ideas shaping the way people are using computers today.

I hope this eclectic collection of articles can serve as a celebration of our temptations to explore and design the ways computers will help us communicate, create, and record our experiences. ◘

Ted Selker

New Paradigms *for* Using Computers

Creating new computer-use scenarios.

NEW paradigms are new ways of doing things or are entirely new things that people can do or understand. For example, the use of a keyboard rather than switches to control a computer was a new interaction paradigm. Such new paradigms rarely come up in more established fields like philosophy; several are created every year in computer-related fields. The computer industry may have given users more new ways of doing things than any other. These new paradigms start as visions that must be tested.

The creation of new paradigms for using computers is an essential piece of an invention-design-evaluation triangle for creating human-computer interaction (HCI). HCI relates to the creation and evaluation of technology that affects the interactional qualities of computers. Textual, graphical, gestural and speech user-interface design and evaluation are HCI efforts, but do not define the field. HCI is not defined by the *methods* but rather the *effects* that these methods can produce. The practitioner seeks to create scenarios (computer/user actions and responses) that achieve communication and relationship goals between a person and the task the computer is being used to facilitate.

It is hard to know when technology will support an idea. Will people react to the paradigm in the predicted way? Possible paradigms gain credibility as they are demonstrated and used. The biggest changes in the computer industry have typically taken at least a decade to refine, but seem to have happened overnight: the Graphical User Interface (GUI) with pointing devices, the use of links to explore data on the Internet, and the recording of a user's actions to allow the computer to react 'intelligently' are examples.

The physical world forces computer users to follow technological possibilities. For example, the accessibility of graphical displays preceded the development of the 3D algorithms and interaction styles.

In the examples that follow, it is the artifact and working technology that often help people have ideas about what they can do next. We will describe some of the new ways that visual interfaces, physical form, I/O devices, and models of communication are changing the ways and kinds of things people do with computers.

Driving New Paradigms: Proposals, Experiments and Successes

COMPUTER uses follow technological possibilities. To prepare for future technologies, researchers try to stretch time forward with money to test paradigms before they are affordable. Researchers who were trying to get a head start built inaccessibly expensive bitmap displays with associated pointing devices in the late 1960s and early 1970s to explore new types of interactions. The GUI was pioneered this way at MIT Lincoln labs, SRI International, and Xerox PARC.

Still, the best scenarios tend to get worked out after technology becomes affordable and accessible. For example, the value of being able to read and respond to email any place and any time, whether connected to a network or offline, became apparent when people got mobile desktops. Until the advent of notebook computers people did not think of or need programs that would cache, hold, and send email offline. The value of looking through email in a remote setting has made systems like cc:MAIL and Eudora that implement this cached email important. We are all becoming aware of the effects of the 30-year-old Arpanet becoming available to the public in the past few years. We are also becoming aware of the value of a pervasive hypertext standard and its availability in Web browsers.

A tension always exists between the survivability of technology developed for a possible future in which today's extravagance will be commonplace and working within the constraints of what conventional technology is currently available. The inherent danger in using exotic platforms to project forward concerns anticipating how technology will actually evolve and how soon.

Portable Computing

Physical form of objects is the first thing that people react to. From the early 1960s, people envisioned portable briefcase computers. No early work is as famous for describing such a paradigm as Alan Kay's thesis on the Dynabook [8]. The Dynabook was to be a portable educational computer system emphasizing a user interface for education.

People used to want terminal access to a computer with all their programs, files, and email. The new paradigm of Dynabook-like portable computers turns this completely around, giving people their data and programs wherever they are in a trustworthy way. Does this free people from their servers? Not at all, especially right now as the World-Wide Web is evolving. Notebook computers allow people to work even when not connected to a computer network. They help people make presentations, collaborate, keep continuity in their personal work, and allow otherwise wasted time to become productive. New platforms that create new ways of doing things don't necessarily invalidate other existing and emerging paradigms.

Many technologies have had to come together to make portable computers practical. Many of these technologies were themselves envisioned for other possible paradigm goals. CMOS chip technology was developed to make high-impedance preamplifiers but this became the way low-power, low-heat CPUs could be made. LCDs were going to make TVs that could hang on walls like paintings—soon TVs will. Good batteries were going to displace gasoline as an energy source—still difficult to imagine. The TrackPoint pointing device was invented to eliminate the need for users to take their hands off of a desktop keyboard—its value for saving space and working in the keyboard control surface of a notebook or subnotebook has become its entree to success.

The Evolving User Interface

THE first dominant user-interface paradigm included patch panels and Hollerith cards. In the 1970s, people were finally getting access to ASCII terminals with command lines. A few lucky users had access to screen editors, which allowed text to be changed directly rather than describing changes on a command line. Starting in the 1960s, vector graphics was used for exciting visualization and design paradigms. Because it was difficult to manipulate a light-pen and a keyboard simultaneously, and because a person's arm got tired holding a light-pen up to a screen, cursor keys and spin-wheels were used to select spatial position.

The 1980s brought commercial and popular success to the GUI. This shift away from the command line operating system interface has popularized the

Figure 1. IBM ThinkPad 760CD with TrackPoint III shown in overlay

editing using other pointing devices.

New paradigms typically require trial and error to make them work. As recently as 1990, the joystick was described as an obviously second-class pointing device [2]. The approach to testing had been to create a joystick that "felt good," then to test it. By integrating the design and analysis phases, we were able to create a joystick that outperforms the mouse in many ways.

The design of this device was a decade-long journey of experiments punctuated with discoveries about the limits of eye-finger control. Our quantitative studies teased out how physiological limits of finger control, cursor dynamics, and eye-tracking were troubling problems with joysticks. Experimental results often contradicted our initial expectations. In these cases, the data pointed us toward areas in which to make improvements. The cursor's speed is controlled by a special force-to-velocity algorithm that was optimized to provide accuracy, speed to the target, and to meet user expectation. Success in handling these constraints may be responsible for its market acceptance. Figure 1 depicts the IBM ThinkPad 760CD, with the TrackPoint III highlighted in a separate overlay. The figure and inset show the ThinkPad with integrated keyboard and the ambidextrous TrackPoint pointing device.

value and power of HCI. The mouse became an important added feature on computer users' desks. The unlikely mouse design is a surprisingly good pointing device, it is said, proving that "man can draw with rock." The mouse allows fast cursor motion with arm movement. Accurate cursor control is accomplished by taking a hand off of the keyboard, supporting the hand on the desk and moving the mouse with the fingers.

Wouldn't it be useful to integrate graphical and textual input control to match the integration graphics and text have on the GUI output screen? The TrackPoint in-keyboard pointing device demonstrates this smaller paradigm shift to an integrated, interactive control surface. The TrackPoint III is a small, rubber-tipped, joystick the size of a pencil eraser, located in a computer keyboard between the <G>, <H> and <B> keys for ambidextrous use. The cursor moves in the direction of the force a user's finger applies. The TrackPoint's position makes it available but not in the way of the touch-typist's fingers. Tests show that making a single selection with the TrackPoint while typing took 0.9 seconds less than with other pointing devices. It can increase text-editing efficiency by 20% over text

New paradigms often become valuable for serendipitous reasons. Many people have found that TrackPoint II was even more valuable for things such as reducing the size of a computer than for the productivity enhancement for which it was conceived. The first serious interest, in fact, came from a bank that wanted to eliminate the space that bank tellers had to devote to a computer mouse—tellers occupy some of the most expensive real-estate locations.

People have gotten used to controlling their media from a distance by using an infrared remote controller. The TrackPoint III can replace the scores of buttons in a remote control with on-the-screen selection (where attention is focused in the first place), and is also helpful for other portable controls such as surgical instruments, where aids to dexterity are valuable.

Form Factors

THE size and shape of a tool affects its use. For hundreds of years people have worn machines (watches) as fashion statements and to aid them in keeping track of time. The simple paradigm of synchronized time has, of course, been central to the development of modern society. Still, when a person looks at their watch, it makes an uncomfortable social statement: someone wants to go, is late, or is bored. User interface design must consider the social milieu. In our portable computing efforts, we work toward creating interfaces that don't absorb a user's attention in a social situation. The pocketable and wearable computers of today have much more ambitious goals than just being timepieces. For more than a decade watches have been available for calculating, storing phone numbers and schedules, playing games, or with environmental sensors. Scores of pocket-sized digital Personal Information Managers (PIMs) are on the market. Straightforward accessible functions have been the selling point of these and higher-capability machines. The HP-200, for example, evolved from a calculator to a high-priced PIM. The "luster" of programmable pen-controlled handheld computing devices (Personal Digital Assistants (PDAs) such as Sony's Dataman of 1991 or Apple's Newton of 1993) has been more noteworthy than their success. Adding communication to PDAs, as in IBM's Simon and Sony's Magic Link, has not yet made them the gigantic sellers that pagers have become. Even as some forms of pocket-sized computers outsell all other computers, we might continue to split hairs deciding which to call PIMs, PDAs, watches, computers, or communication devices. Even as they become indispensable, we continue searching for forms and working scenarios with ergonomic validity—see Figure 2 for several PDA prototypes. This collection of IBM products and product visualizations are representative of today's possibilities for running full-scale operating systems in scenarios that are as integrated into our lives as pulling a wallet out of our pocket.

Wearable communication and computing devices are already benefiting inventory specialists, package delivery, and car rental companies. Many assert that wearable computers should be general-purpose com-

Figure 2. PDA prototypes

puters. For such a paradigm to work, people must be able to interact with the device in a facile and immersive manner as with a PC. To this end, several people have been wearing early head-mounted displays and carrying PC computers in a sash or backpack (see the Viewpoint column in this issue). Unfortunately, the psychological absorption requirements of installation, navigation, and maintenance imposed by a general-purpose computer seem to compromise the wearer in social situations. These experiments are preparing people for a time soon when eyeglasses will be able to augment one's view of the world by providing and recording information.

The use of speech or handwriting recognition to eliminate the physical space and use of hands required by a keyboard have often shown even poorer results. In *unrestricted* forms, these input scenarios have had a reputation of being brittle, taking more concentration and further distancing users from their social milieu. *Crafted* selection and gestural use of speech and stylus input modalities have been robust and useful in a large range of applications. A mass-marketed voice-input interface is available in a toy parrot—for $20.00 it will repeat words twice with a shrill parrot-like enunciation and ruffle its feathers. This scenario for use is non-absorbant, obvious, and trustworthy, three important goals for user interfaces [11].

For more than a decade, computers have been part of automobiles, communicating with drivers and controlling everything from the temperature to the way the car stops. Computers are now taking on navigation and even driving in traffic-control activities. The Global Positioning System (GPS) is now qualitatively changing the driving-computer user paradigm. These computers annotate a driver's experience with a cursor on a map or give audible directions such as "exit coming up", attempting to orient the driver. Dramatic demonstrations have been made by Mercedes-Benz and others deploying computer-driven cars to negotiate freeways at high speeds without human intervention. The safety required for vehicles centers their computer scenario development on reliable non-absorbing transparent interfaces.

Marvin's House

Marvin Minsky has been at the center of many computer science movements that have made a difference. I've always wondered how a mind like Marvin's works. How does it take ideas from different, unusual sources to solve a problem? Marvin is interested in how we think. He wants to build computers, machines that will assume responsibilities and work with the community. This is a story about a visit to Marvin's house.

As you walk into the house, you are greeted by the barking of an artificial dog. The doorknob is covered with a carefully adhered rope that protects a person's hand from contacting the brass on a cold day. Inside this three-story brick house is Marvin and Gloria, his wife. Gloria, a pediatrician, greets us with a presence that says, "Hello, I am ready to play." Marvin, too, works easily at being a child. His living room is filled with several pianos, an organ or two, and a swing hanging from a beam—the swing moves out of the way, allowing for more "play" space. On the floor are some ziplock plastic bags that serve as balloons when Marvin's grandaughter comes to visit. Marvin and I stand on the balloons enjoying the ways they can be repeatedly popped open and refilled.

In the corner is a beautiful mural on the wall: an assemblage of diagrams from patents painted by Marvin's sister. We walk by a French guided missile that has two kilometers of guidance cable in it. A painting on the fireplace mantel has slipped partway out of its frame, making it appear quite surreal. Marvin spends a few minutes commenting on this picture and how, in fact, he likes it better this way.

Among the strewn mess on the mantel are pieces of interesting things, museum art. We find ourselves standing in front of the first confocal microscope—an idea so unusual that I didn't understand it the first or second time it was explained to me. A sensor and a light source both use the same optical paths, and the user's job is to change the distance of the light source from the optical path so that the distance is determined to the reflecting specimen using the same concentric focal point. He used the zenon arc lamp to get a good point source and an electromagnet to position the device. Marvin invented this decades ago; now it stands looking simple and awkward. An expensive "copy" of his confocal microscope is now the centerpiece of many modern high-tech biological laboratories.

The ingenuity of the object next to it boggles my mind. I know many good mechanical designers, but this is one object I am sure none of them would have designed. Not because it might not be the best design, but mostly for how deeply the designer had to think about the relationships between mathematics and constrained joints to design it. Complicated mechanical devices are so difficult to make because they are so difficult to visualize; typically no one would use eigenvectors

The Visual Language of Interface

Since our eyes are the major input to our brains, it is no wonder that new paradigms for using computers often involve making better computer images and animated interfaces. As good displays and enough memory to drive them have become available, people have found themselves immersed in graphically-rendered interfaces. In the early 1970s, people at Xerox PARC started experimenting with mice and bitmapped displays, while the Spatial Data Management System was being developed at MIT. The GUI has been maturing to become the dominant computer-interface paradigm of today. In 1979 Atari's Pong game brought computer environments to the public. Computer interface designers have been learning to use dramatization interactivity and stylized 3D realism ever since. Attempts to put 3D direct manipulation into mainstream desktops have been spurred on by General Magic's Magic Cap, and software manufacturers making interfaces like Microsoft's Bob or Packard Bell's "hallway" introduction interface. The value of 3D interfaces for visualization and design is accepted. Productivity gains for general user interface actions are coming as we begin to articulate the value of the striking 3D spatial landmarks and other qualities that characterize efficient navigation in physical space.

The paradigm of using graphical visualization as an analysis tool can make abstract concepts manipulable and concrete. Possibly the simplest graphical computer interface with the greatest impact of all time was the spreadsheet introduced by Visicalc in 1979. This simple table visualization makes columns and rows of numbers add up on a computer screen as might be done on paper, changing the way most businesses do calculations. Ron Baeker created visualizations to teach students to see how sorting algorithms move data [5]. We created a computer memory hierarchy visualization that enabled us to see how the matrix multiply algorithm on a commercial compiler could be increased 30 times. The visualization emphasized unnecessary memory transfers and reminded us to leave space for the answer to percolate back up through the memory stages system [1]. By augmenting the standard analytic approaches with visualization analysis, we were able to see how bottlenecks occur in moving data through a computer memory hierarchy: from disk, memory, cache, registers to the Arithmetic Logic Unit in the processor, and back out again to disk. A logarithmic scale for the size of the stages in the memory hierarchy make a system that can show memory stages that differ in size by

to do so. In this case six cables are wrapped around six joints. Six pulleys on each of six joints make the entire contraption of thirty-six pulleys on six joints and six cables work as one unit. As you move any of the three joints on one arm, the three joints on the other symmetrical arm moves in exactly the same manner. What is strange is that none of the cables actually end at any of the joints; instead they wind around in a continuous fashion. The "hand" works by the constraints imposed by the six cables as they wind around the different joints in different patterns. The eigenvector for the entire system allows for the constraints of any joint's position to be determined by three joints on the associated joint on the other arm. I wonder if it only has to move three joints, why did he mistakenly use six cables to solve three equations? Marvin doesn't say. He does note that he added the other three for redundancy so that if any of the cables broke the system would still work. Whether the three redundant were in fact created because he thought of it or because he accidentally designed it remains unclear. But the playful serendipity of design is always a part of Marvin's life, and as such, either answer is reasonable.

In Marvin's house, his inventions and the inventions of others are displayed side-by-side. He talks about them with equal amazement. One of the remarkable things about a great man like Marvin is his generosity. Instead of jealously guarding his invention, Marvin enjoys creating a community of inventors. As Marvin walks through his house, he thinks of how to promote other people, how to make their work shine. I like to say that science is the act of making ideas that make other people have ideas. To the extent that we promote ideas, we are creating science. I believe Marvin also thinks this way. I picture him working in his refreshing, unusual way, on his books and other projects as being the quintessential, most valuable kind of human. A person who creates, who values the creations of others and, more importantly, values and creates the spirit in people around him.

A mind that can create a self-reflecting mechanical arm, the intricacy of a confocal microscope, or nurture the field of artificial intelligence, is a mind that focuses and works on hard problems. And the joy of getting that person to show his playful side, to consider possibilities that are only starting to emerge, is the way that we want to grow as humans who create the communities that allow for the kind of communication and multiinvolvement available today. Maybe it isn't the robots in our minds that will make us more aware of the possibilities of what an intelligent computer can do. Maybe it's working with people who have outside opinions, like we do on the Web, or in universities, or with Web crawlers and search systems that will create the agents of the future that Marvin envisions. —Ted Selker

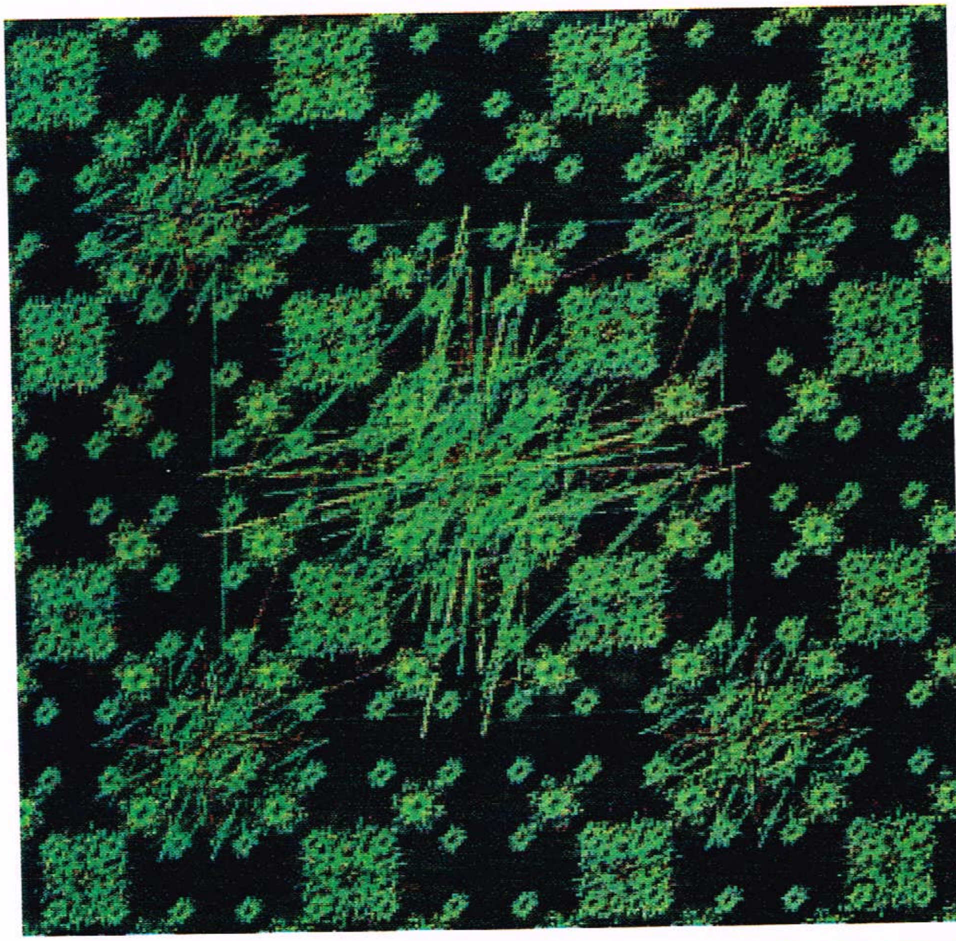
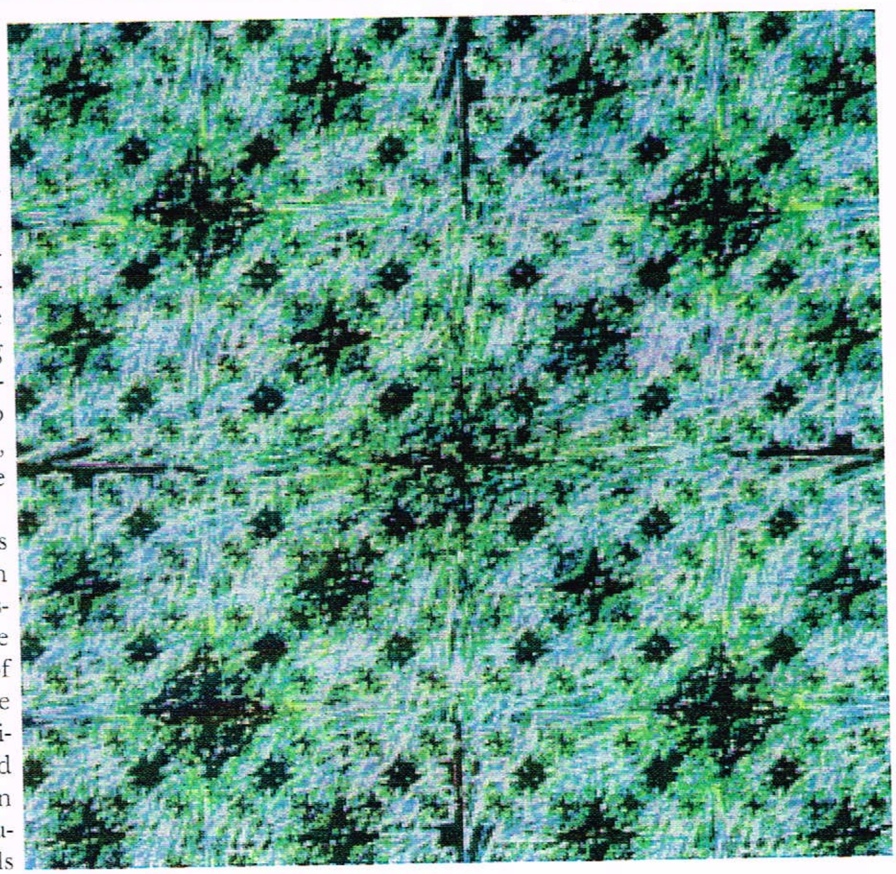


Figure 3. Sample KALIDE output

10 or more orders of magnitude simultaneously.



VISUALIZATION can be convincing; this is a motivation for immersive graphical computer environments that allow a person to visually explore. These immersive virtual realities are becoming prevalent solutions in applications for teaching people how to navigate a house in a wheelchair, fly an airplane, or locate software on a computer.

Room With a View (RWAV) is a room-mounted rather than head-mounted virtual reality system. Workers' offices used to be their information reserves, full of personally valuable reference materials and paraphernalia. Visitors could browse the books and pictures on the walls to learn about the inhabitant of a particular office. Today these materials are often kept online. Computer users rarely have large enough monitors to show an orienting amount of text on their screens. Still, they typically leave a backdrop of icons visible as an orienting context. We started working on RWAV in the mid-1980s to allow office workers to return to using their environment and peripheral vision to orient them. Orienting imagery of shelves with books and folders are projected on a user's physical office wall. High-resolution text or graphics that the user is manipulating are displayed on clipboard-like devices. High-resolution "viewboards" can be placed on a lap, in a desk stand, or on a table as surrogate books, folders, and papers. Picking a viewboard off of a desk is like picking a book or folder off the shelves on the wall. The user directs one of these at a 'book' or other icon on the wall to open it for manipulation on that viewboard. Gestures or a keyboard are used to manipulate information on it. The board is pointed at the wall to replace the book or article in its filing position in the virtual room. A user's peripheral vision can be used to orient, and foveal vision to concentrate, as the eyes seem to be designed to work best.

New approaches often produce ideas that can be useful in established approaches. A simplified version of RWAV can be arranged by adding an orienting monitor next to the primary monitor on a standard workstation.

The paradigm of using computers to augment relationships between people has been a recurring interest in the computer community. The Data Disk at the Stanford AI lab in the 1970s allowed a user in any office to instantly look at and work on a screen from a friend in any other office. People formed tight collaborations working on the same text or code with little conversation. Today such shared data and Computer-Supported Cooperative work (CSCW) is skyrocketing with the World-Wide Web. Xerox's "Viewboard" for present and tele-present collaborative meetings and IBM's group decision support system that creates structured collaborative meetings are examples of the many products available in this area today.

A WONDERFUL cooperative work paradigm would allow me to tell you and show you everything I want to share: all the people I know, all the places I have been, all the things I can think of—and I want them to be in front of you as we are having a conversation. Such technology, now available on the Web, might change our perspectives as we make new computer

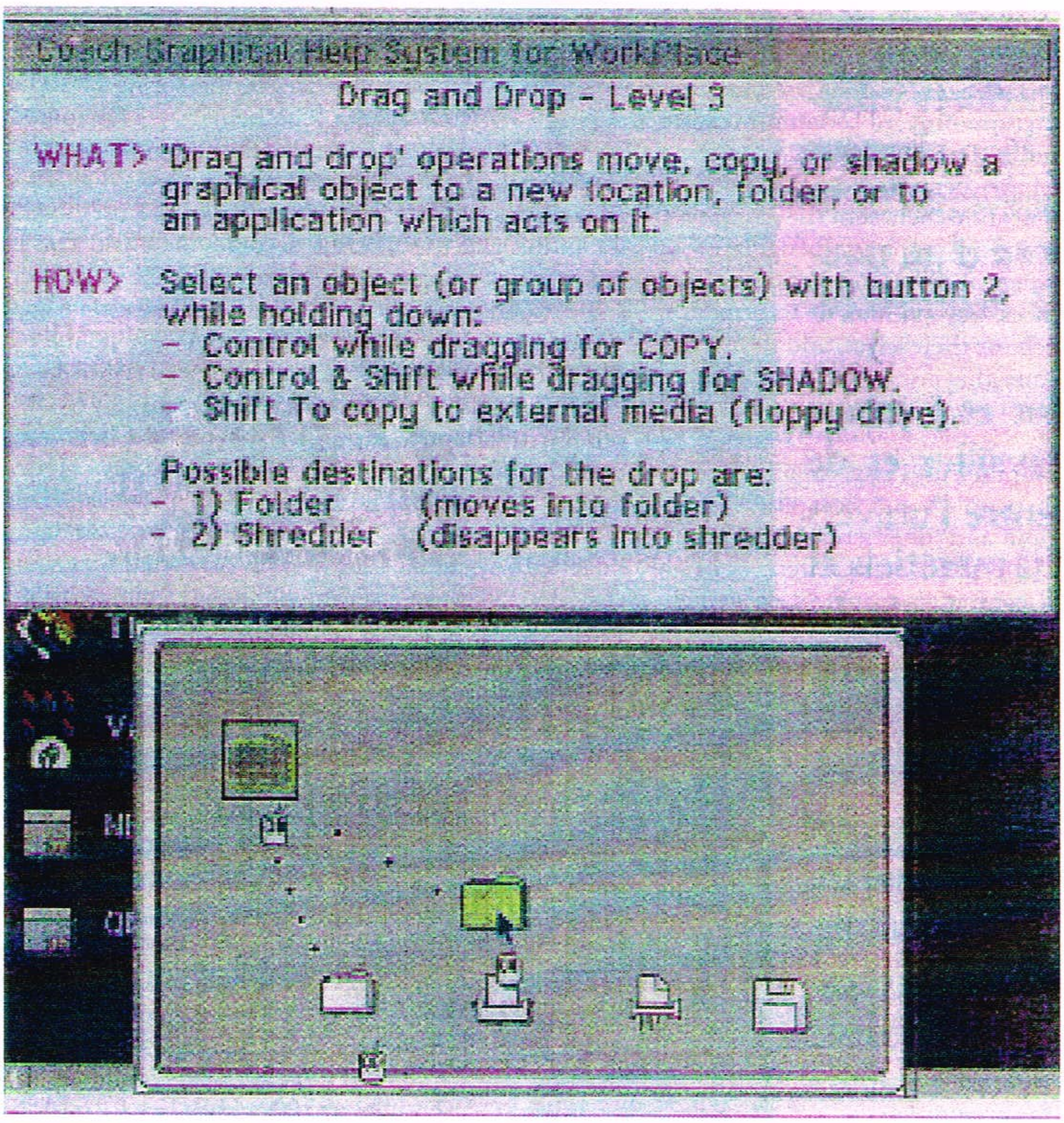


Figure 4. COACH screen

and programming environments. Such technology allows us to focus on accessing, browsing, searching, and working toward having scenarios where the task is more showing and testing than searching and constructing.

Thinking Interfaces

People have always had a tendency to imagine machines that could think. An AI Turing test success goes to any machine agent that cannot be distinguished from a person in some domain. In the 1950s Newell, Simon, Minsky, Selfridge and others were writing about the things that have turned into what we now think of as agents (see the sidebar "Marvin's House"). A variety of agent computer programs are described in the July 1994 special issue of *Communications of the ACM* on agent technology.

Social agents create user-interface paradigms in which a computer is adding other people's similar experiences to one's sensibilities. Social agents compare statistics of one's interest or values with other people's; this comparison is used to predict if a person might like particular music using the Ringo system, or want particular mail using the Max system [11]. Smart user interface agents can be used in other ways to reduce the amount of communication between a person and a computer, as well.

Since 1979, we have been building systems to enrich the communication between the computer and the person by having an underlying layer of understanding between them. This Adaptive User Model (AUM) reduces the amount of information that users must search for themselves, and so increases the amount of time that users could be concentrating on their tasks.

NEW computer uses must simultaneously match people's and computers' computation and interaction constraints. Early efforts to make interactions in which computers would learn to help users were often stymied by involving large experience-driven domains. This would turn a project like creating a "smart chess program" into a project requiring a large representation for the system to understand. The focus would then center on capturing a domain-knowledge expert in a computer rather than learning to understand the user. Other agent systems that attempted to perform deep analysis of difficult problems ran into computational-complexity problems. These systems would doom themselves by basing success on trying to do something like proving a program's correctness to understand a user [12]. Creating new ways for using computers requires focusing on the interaction scenario; the nature of the human computer conversation, the pacing requirements of the task and robustness requirements.

With such thoughts we set out to create an agent that would only 'perform' if it could learn from user criticism. We wanted to make an AI learning user interface that would give a satisfying experience only if its learning matched user expectations. The approach was to choose an area in which the human had no more special knowledge than the computer. A radially-symmetric kaleidoscope with repeating patterns was used as the domain. At each of 5 levels of pattern complexity the system recorded an AUM describing the users' experience. As users liked the patterns, they pressed forward on a joystick, creating a model of what shapes and patterns a user liked (and didn't like) as the AUM. The system drove the patterns the user would see entirely off of the adaptive user model. The computer would then not be at a representation disadvantage from the beginning. It should interact continuously and not lag in the conversation. Finally, it had to have enough feedback that the user could easily feel that the system was responsive.

THE KALIDE agent system, in which user criticism is used to drive system creation, was given to users in the spring of 1980. The system immediately fascinated enough people that the computing administrators asked that it be removed from public use. This proactive assistant-style agent can change a user's skill set; in this case the art critic becomes the artist. This is a new way of using a computer in which the computer is the expert but the user is the author. The two images shown in Figure 3 are taken from the animated KALIDE agent, which draws in reaction to criticism.

Although they may not have had persistent AUMs, other early adaptive expressive systems have been successful as well, for example, Myron Krueger's Media Space [10], and Harold Cohen's ART projects [6].

How far can such an assistant-style agent paradigm be taken? As late as 1985 articles predicted that such real-time adaptive user modeling was not feasible [12]. To create a new paradigm one must create a demonstration that its interaction scenario will solve real problems.

The COgnitive Adaptive Computer Help (COACH) system was created in the 1980s to demonstrate that adaptive agents could be useful in user-interface scenarios. COACH is a system that records user experience to create personalized user help. It is an interface agent that teaches a user rather than acting on that user's behalf. Just as a football coach stands on the sidelines and encourages, cajoles or reprimands, so COACH is a system that does not interfere with the user's actions but comments opportunistically. COACH might choose to use description, example, syntax, timing, topic, style and levels of help according to user-demonstrated experience and proficiency. A description advertises a command or function and is helpful for getting started, but might become ignored if it is presented too often. Example information demonstrating how to perform a procedure is often valuable until the procedure is mastered. Syntax is a generalization of examples that is useful for deeper understanding of procedures.

The COACH system records a user's experience and expertise for learnable things (syntactic and conceptual) as they are being used. It uses this passively recorded user model of how long it has been since a person had an experience with some part of a system to guide its involvement and approach to helping them. Figure 4 shows the kind of graphical contextual help that COACH2 creates to adaptively teach a user to use a GUI. In a user study teaching the Lisp computer language, the COACH system allowed users to complete five times as many exercises as their counterparts without the adaptive agent.

Machine-learning mechanisms can be employed to shift computer education paradigms away from a pre-structured programmed or syllabus-style classroom experience to concentrate instead on users' individual needs. The adaptive teaching scenario moves students toward an apprenticeship, or learn-while-doing approach. The COACH help system taught us that an adaptive user model can improve people's productivity even when it is not an assistant. The current COACH work involves teaching GUI material with sound, animation, and annotation.

Let's look forward to adaptive systems explaining and simplifying the clutter in HCI screens and scenarios. The value in all these adaptive systems will not be that people *like* the adaptive nature; it will be that they don't notice the system.

Conclusion

New paradigms for using computers are created as we change the roles computers play in our lives. This article has described examples of small and large changes to show the problems and approaches to creating the human-computer interfaces of the future.

The goal of HCI is to make non-absorbant interfaces that take the tools out of the task. When people think of new ways of using a computer they should ask questions: Does this paradigm fit with the behavioral motor and perceptual aspects of a human? Does it match the technical realities or possibilities that can be demonstrated with a machine? Can this paradigm create scenarios that improve human experience or reduce the physical or mental energy that is needed for humans to do the things they want to do? How will we know if it is a great way to do things? How can it be tested without too much effort now? What else does this paradigm enable?

As a community of innovators, researchers, and designers, we nurture the paradigms that we believe will succeed. As the successes accumulate, people will build libraries of techniques and technologies that are useful in particular situations. The computer field is in the enviable position of being able to define the tools of the future through new paradigms for using computers. ◻

Acknowledgments

I would like to thank my wife, Ellen Shay, and Pierre Vandeginste for their patience and help with this material.

References

1. Alpern, B., Carter, L., Feig, E., and Selker, T. The uniform memory hierarchy model of computation. *Algorithmica*, 12 (1994), 72–109.
2. Barrett, R.C. and Selker, E.J. Negative intertia: A dynamic pointing function. In *Proceedings of CHI 95*, ACM, April 1995
3. Bennahum, D. *Marketing Computers*. Feb. 1995. http://www.reach.com/matrix/thereturnofthepda.html
4. Bewley, W.L., Roberts, T.L., Schroit, D., and Verplank, W.L. Human factors testing in the design of Xerox's 8010 Star office workstation. In *Proceedings of CHI '83* (1983), 72–77.
5. Brown, M.H. *Algorithm Animation*. MIT Press, Cambridge, Mass., 1988.
6. Cohen, H., Cohen, B., and Nii, P. *The First Artificial Intelligence Coloring Book: Art and Computers*. Kaufmann, Los Altos, CA, 1984.
7. Frankish, C., Hull, R., and Morgan, P. Recognition accuracy and user acceptance of pen interfaces. In *Proceedings of CHI '95* (1995), 503–509.
8. Kay, A. and Goldberg, A. Personal dynamic media. *IEEE Comput. 10*, 3 (1977), 31–41.
9. Koved, L. and Selker, T. *Room With a View*. IBM T. J. Watson Research Center Tech. Rep. RC, Yorktown Heights, NY, 1991.
10. Krueger, M. VIDEOPLACE and the interface of the future. In *The Art of Human-Computer Interface Design*. B. Laurel, Ed. Addison-Wesley, Reading, Mass., 1990.
11. Maes, P. Agents that reduce work and information overload. *Commun. ACM 37*, 7 (July 1994), 31–40.
12. Selker, T. COACH: A teaching agent that learns. *Commun. ACM 37*, 7 (July 1994), 92–99.
13. Sleeman, R. and Brown, J.S. *Intelligent Tutoring Systems*. Academic Press, New York, (1982).
14. Stefik, M., Foster, G., Bobrow, D.G., Kahn, K., Lanning, S., and Suchman, L. Beyond the chalkboard: Computer support for collaboration and problem solving in meetings. *Commun. ACM 30*, (1987), 32–47.
15. Sutherland, I.E. Sketchpad: A man-machine graphics communication system. In *Proceedings of the Spring Joint Computer Conference* (1963), pp. 329–346.

About the Author:

TED SELKER is an IBM Fellow and Manager of User System Ergonomics Research (USER) at IBM Almaden Research Center and a consulting professor at Stanford University. His recent successes include the design of the TrackPoint in-keyboard pointing device with its special behavioral/motor match algorithm available in IBM computers, creating COACH, an adaptive agent that improves user performance, and the IBM 755CV overhead projection display notebook computer. **Author's Present Address:** Almaden Research Center, 550 Harry Road, San Jose, CA 95120. email: selker@almaden.ibm.com