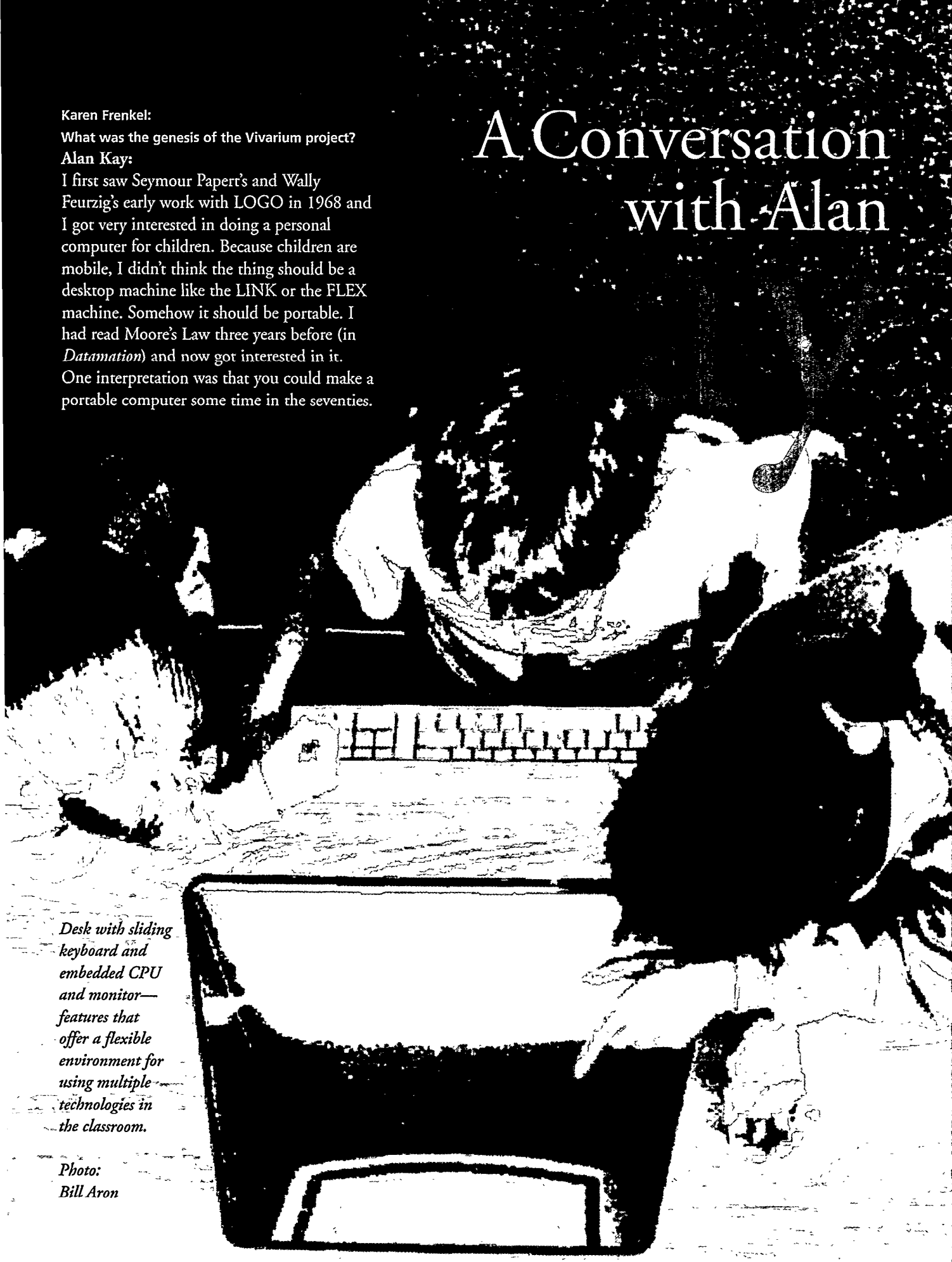
A Conversation with Alan

Karen Frenkel:
What was the genesis of the Vivarium project?
Alan Kay:
I first saw Seymour Papert's and Wally Feurzig's early work with LOGO in 1968 and I got very interested in doing a personal computer for children. Because children are mobile, I didn't think the thing should be a desktop machine like the LINK or the FLEX machine. Somehow it should be portable. I had read Moore's Law three years before (in *Datamation*) and now got interested in it. One interpretation was that you could make a portable computer some time in the seventies.



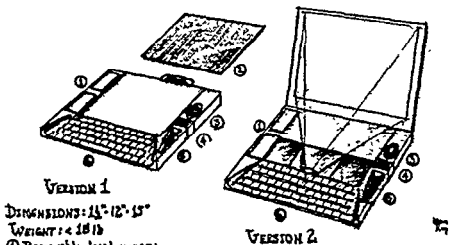
Desk with sliding keyboard and embedded CPU and monitor—features that offer a flexible environment for using multiple technologies in the classroom.

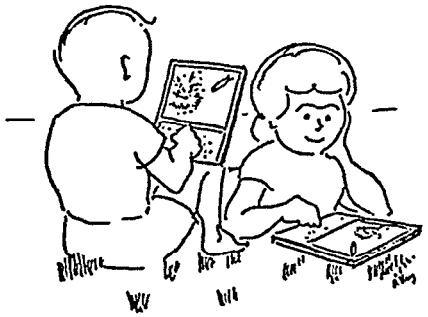
Photo: Bill Aron

The other influence I had was the history of books. There was an "institutional" phase. In the year 1400 in the Vatican Library there were only 392 books. They were done by hand and it took ten years to copy one. In today's terms, 10 years at $15 dollars/hour is $300,000. Special books might cost the equivalent of several millions of dollars.

In the mid 1400s, when the printing press came along, Bibles were large, they were almost two feet high and the type was about a half an inch high. They looked just like manuscripts. The reason for this is that they didn't know what books should look like. A Gutenberg Bible of 1455 cost about 3 years of a clerk's wages (about $60,000 today). So a well off person could own one. I thought of this as the "personal" book. I realized that desktop computers like the FLEX machine looked just like time-sharing terminals. Time-sharing is the institutional phase of computing. Desktop computers are the personal phase of computing. (The FLEX machine was called a personal computer in my 1969 thesis.) Then I remembered that in another fifty years a Venetian printer by the name of Aldus made books the size they are today. And that was because he went out and measured saddle bags in Venice and he realized for the first time that books could be carried with people and so they had to be made in portable size. So this idea of a Dynabook came from thinking about this third phase of computing—what I would call intimate computing—where you can own these things like you own books and you can take them with you.

Early renderings of Dynabook.





So I started thinking about making a powerful idea toy for children that I called the Dynabook. In 1968 it was a cardboard model about 1/2" thick by 8-1/2" by 11". When I came to Xerox Parc in 1970, I set up a group and started working on that, which is what led to the "interim Dynabook" workstation (later known as the ALTO) the overlapping window user interface, modern OOP, and all the rest.

To what extent has your vision of the Dynabook been met?

Marshall McLuhan makes the distinction between a technology and a medium. The technology is the machinery and the medium is the way you use it. Consider for instance printing press machinery (the technology), and then the essay form (the medium), that was invented about one hundred years later. Television is technology, American television is the medium.

The technology of a Dynabook was behind schedule because America gave up the work on the flat screen display. We waited until the Japanese used our patents to do it. It would have been here in the early 80s. But basically the technology looks pretty darn close to the cardboard models I made 25 years ago.

What I think of as a medium is not here at all. Just as McLuhan predicted, computers today are almost universally used—except by scientists—for just imitating paper. We're still doing the same thing the Summarians were doing in cuneiform in 3000 B.C. The exceptions are those computer scientists and some physical scientists who use a computer for what its really good at, which is making and understanding complex models of things —including those that live, and not least, itself.

After the year 1600, science was something everybody needed to learn. But less than 10% of Americans can hold a decent conversation about scientific topics. I believe that there are under a 100,000 Ph.D.s in the physical sciences in the entire United States. So what has been the most powerful force in our civilization in the last 400 years—science—is something about which most people are totally illiterate. It's not too surprising then that most people are not using computers for what they're good for either.

Since you were involved with several versions of Smalltalk, what do you think Smalltalk 2000 could be?

I did Smalltalk 72 and had a fair amount to do with Smalltalk 76. I got less interested in Smalltalk 80. The original Smalltalk was a lan-

guage designed for children to program in, Smalltalk 80 definitely isn't, its a language for adult systems programmers

The nice thing about a tennis racket is that kids use the same basic kind of racket as the pros. Its nice to have a tool where what changes is the sophistication of the user rather than the sophistication of the tool. The English language is used by children and by adults. Its the same language but its the sophistication of the user that gives you the art of prose.

I think the same thing should be true of programming languages. The most powerful programming languages can be and should be created for children to learn. The groups that have produced the commercial Smalltalks have not heeded this at all.

Do you think that children should be taught programming?

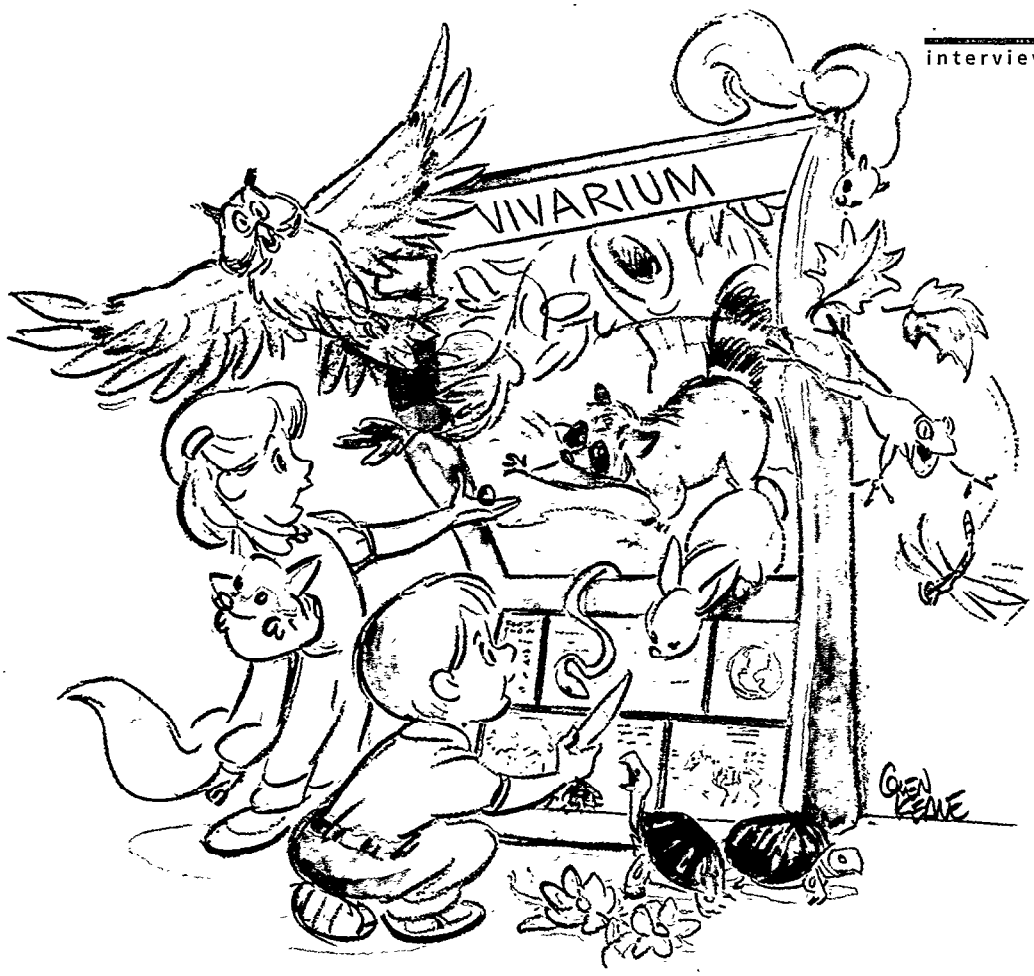
Children should be taught to think. Given that our IQ is the same as it was 10,000 or 30,000 years ago, its probably not genetic changes that have made the difference between what we do today and what we did 100,000 years ago. What has changed is our ability to represent the world in interesting ways and to think using those representations. On top of our wired-in cells we have all these drives for food, sex and so forth, but partly through our linguistic structure we've laid on top of ourselves a kind of Turing machine that helps us think better about things. People who learn to think well in our society are not doing it just by IQ, they do it by being more facile at representing things. That's what I'm interested in. I'm only interested in programming in so far as it is a vehicle for that kind of thing. I think that the kind of programming language you give kids is very critical because they tend to think of what the computer can do and what the computer can not do in terms of what the programming language is good for and what it isn't good for.

Most teachers don't know anything about math and science. They've never done it. What's worse is that even from the liberal arts side, most teachers in America don't write.

What sorts of challenges do you think that educational software designers are confronted with today?

I think that the biggest challenge is to get any kind of content at all onto the computer screen. Almost all serious discourse in our culture for the past hundred years or so has been textual. The computer screen, the CRT, is antagonistic to reading text. Right away you're putting something in front of a child that is antagonistic to the most serious discourse of our culture. What the CRT likes is low-resolution, moving images. You just can't represent many important things in science by showing low-resolution images without attached symbols. So the biggest problem that software designers have to face today is that the general computing systems they're programming for are medieval. Its like electronic stained glass windows—its colorful and vivid and attractive but you can't tell a very modern story with it.

Here's another argument we can make: Children in the US are almost certainly going to grow up to drive cars. So why don't we, at the age of two, put them in little motorized vehicles so that from the age of two on they can learn how to be much better drivers than if they just start at age 16? Sounds almost reasonable doesn't it? From my standpoint this is horrible because the kids' muscles would atrophy. On the other hand, we can give kids bikes, which wouldn't bother me at all because a bike is something that allows kids to exercise at the highest possible peak. Then the bike amplifies that. So the questions we have to ask for any kind of software are: When is it a bike and when is it a car? And when should it be a bike and when should it be a car? Giving kids software that is like the car software that adults want to use, which is basically prepackaged solutions to things, is disastrous. What kids



Early (1986) animator's image of an interactive computing environment where children can program their own ecosystems. Animation: Glen Kean

need are challenges. "User friendly for kids" doesn't mean easy. What they need is hard stuff.

How would you like to see the classroom and technology coevolve to create an optimum learning environment?

If you think about it, most of what people learn well is not learned in schools. Where do people learn to cook? They don't learn that in public schools. If you're interested in it, you learn through other people, you might go to a special cooking class. The same thing holds for tennis or music or whatever. When people are really interested in something, they want to go to a place where other people are doing it. You want to learn from as many different people as possible. For instance, having all musicians in a class be the same age is ridiculous because you really want to group roughly by ability. You also want to have people with different abilities intermixed because that's how they learn. All this is extremely far from any theory of American classrooms.

Most teachers don't know anything about math and science. They've never done it. What's worse is that even from the liberal arts side, most teachers in America don't write. My experience in working with teachers over the last 25 years is that most of them don't read. I'm not saying that they're illiterate, but reading is not a big part of their daily life and writing is a much smaller part. Its very rare to find teachers who can write essays about what they do. So what you have here is the most artificial of all situations; a bunch of people trying to move curriculum—from some book that they didn't write, about subjects that they don't know anything about—into some poor child's mind. That could not be more ridiculous. Its basically an institutional factory type model. And it just doesn't work. The most important thing in any classroom is not even whether you have a computer in there. Its whether you have a teacher who actually understands something.

You don't need a penny to do science. You can do science with film cans and milk cartons. You can do a lot of things that are interesting with a computer on an 8-bit Apple 2. What I'm saying is that a good teacher who understands this stuff, is actually a practitioner of this stuff, can be a hundred times more effective than any technology.

Above and right: Computer generated simulations of kelp forests created on an Evans and Sutherland CT-6 computer.

Were any of the criticisms you just made about "classic" classrooms around the country able to be overcome in the Vivarium project?

The Vivarium project was possible in part because the school was already extremely well set up. Now this was a public school, but one that was organized in a special way by the parents, teachers and the principal. To entice parents to use the busing system in Los Angeles, the Los Angeles school district said all the busing schools could choose their own process—the parents and teachers and principal would decide on the curriculum. So each of the magnet schools in Los Angeles is a different experiment. This school was run like a pretty good liberal arts graduate school.

How old were the kids?

Grades one through six.

What did you learn about learning?

Part of what I learned I already mentioned—learning is done best in the context of doing. Another thing is that the most important learning happens at home. Schools are very exciting and there's a lot of social interaction—and school is a great place to learn about things you haven't heard about before. But basically there is not enough quiet time at schools, so even in good schools, what the kid does at home has a lot to do with how the stuff sinks in, with how the kid learns to think.

What impact do you think networks in the schools will have on learning?

Kristina Hooper Woolsey, a distinguished scientist at Apple, started a very nice project using modern electronic mail systems to connect up professional scientists with kids in a fifth grade classroom. The kids can send email to the scientists with their questions. Now a good scientist won't answer any questions directly but instead will challenge the kids back in a way that teachers won't generally do if they view their job as dispensing information instead of asking questions. Another thing that the professional scientists can often see is when a kid is going in an unorthodox direction that might be

on the track of something interesting. Most teachers react in a negative way when kids go outside of the curriculum syllabus. In science you learn how to argue about your theories and that's something scientists can teach the kids.

Here's an example. Many five and six year olds have a theory that wind is made by the trees moving their branches. You should never tell a child that is wrong. The real question is in what setting is that theory really working. Now that theory does not work well out in the physical world. It works just fine as a poetic image, and it works really well in a stage play that has some mystical elements in it, like *Snow White.* Science doesn't believe in truth and falsity. What interests scientists is the setting in which the theories"work".

What do you think about computers that learn about the user? What impact might that have on schools?

The only system I've ever seen that might have a chance of doing that is Doug Lenat's CYC. I think its great to have a computer learn more about a human. There are a lot of things you can do by coaching with meta ideas—asking leading questions that help focus a learner's attention more strongly on what is actually going on, instead of telling them what they need to know. I used to do a lot of theater, which is one of the things that led to my interest in user interfaces. In theater your job is not to tell people the truth, but to evoke it from them. There is a huge difference between the two.

Can we talk for a moment about your time at Atari?

Well I actually went to Atari as a Trojan Horse.

What do you mean?

Most people can't imagine 20 years later. So this year 350,000 Americans will die because they didn't change their diet and exercise habits. For some the answer is that they would literally rather die than change.

I wasn't a big fan of computer games, but I went there because Atari game machines were computers. I was wondering what else you could do. In fact, the Vivarium project came out of a kind of a game that we made there called the Aquarium. It was a game where you constructed various kinds of ecological environments.

Can you lend some perspective on what you gained from that experience? Is there any kind of a convergence going on between games and educational software?

Well, the first video game that I ever played was SpaceWar!, which goes back to the PDP-1, probably around 1962, Steve Russell did it at MIT. The first adventure game was in the late 60s. Certainly there was stuff on the Arpanet in the 1970s. The first game that Nolan Bushnell did was a game called Computer Space which is essentially SpaceWar!. That game was too sophisticated for most users in bars. The big breakthrough was Pong, which is a much simpler game. The games have stayed similar, but the media have changed. They were done on very coarse bitmapped screens and were basically shoot-'em-up or adventure games. The screen animation has gotten better but generally the games are designed to fit the stereotypes of preadolescent and early adolescent males.

Its very hard to come up with a game that's actually satisfying to people and is about something adult thinkers actually do. Everybody knows what Michael Jordan does and everybody knows what Michael Jackson does. But nobody knows who Michael Faraday was or did because he was a scientist and his world is internal. The problem of getting intellectual stuff into games is extremely difficult because most games don't involve much abstract thought.

What about things like interactive TV?

Intellectual thinking is highly romantic but a lot of it is done in cold rather than hot blood. Democracy is a "cool-thoughts" process. Everything that is built into our constitution is to hold down any kind of direct action. Imagine having direct voting via television—something that could be done today—then something happens and somebody gets on TV and says "Let's go bomb the hell out of the Libyans." Television is all about emotions and in the heatedness of the moment, 60% of the people might say, "Yeah, lets go out and do it." You just can't run a country that way. The whole idea is to have delays built in so that when you start thinking about it you are in a much cooler frame of mind and are capable of thinking about alternatives. Removing those barriers with electronics is a very bad idea.

One of the things that printed books were good for (besides having a lot of different opinions) is that they are a very cool technology. Books are the most remote, the most alienated thing from the world that you can imagine. But in fact, it is that remoteness that is enormously responsible for the invention of science and democracy in our country and a whole bunch of other things, none of which were thought up by people in hot blood.

I understand that Apple's strategy is to become more consumer-oriented and people speak about ubiquitous computing. I wonder if we are moving toward a kind of computer literacy whereby people would interact with a computer in a fluid and enjoyable way?

Getting stuff into people's homes is only part of the deal—everybody has televisions, but

not everybody thinks. The whole consumer business is pretty much looking for the next kind of narcotic, the next thing that people can be convinced they need. Apple has always been very interested in content in education. As this franchise extends into millions of homes the challenge is to show people an alternative and a stronger set of values for using these things.

This is like the wide variety of food you can choose from so that you don't wind up with heart disease 20 years later. Most people can't imagine 20 years later. So this year 350,000 Americans will die because they didn't change their diet and exercise habits. For some the answer is that they would literally rather die than change. This is a very tough fight. Media needs a value system to go along with it and selling people on a value system today is pretty darn hard.

What are you interested in now?

My topic is still "what is a computer when it is trying to be good." I'm trying to figure out what computers are good for and how to make that appealing, interesting, and understandable to people. That's all I've done for the last 25 years.

Is Vivarium still in the works?

Vivarium has been phased out as a research project. The school we worked with is now in a consortium of a number of schools in the Los Angeles area that are sharing the things they've learned.

What projects are you working on now to help you understand what computers are good for?

We have a very large-scale experiment that we're creating with the entire school district of San Jose. It starts off by designing a whole new school, called the 21st Century school. And we're also working with San Jose State University, which graduates about a thousand teachers a year from their college of education, to set up a new curriculum for teachers. This is on a much larger scale than the Vivarium.

Is this an Apple-funded project?

Basically its an alliance with three partners—The San Jose School District, The College of Education at San Jose State University, and Apple. We've been approached for advice and our experience on designing learning systems for children. Its not just a school, but a way of doing schooling and a way of training teachers to fit into that kind of schooling, and a way of designing and fitting technology into that. So its pretty exciting.

When did it start?

The project began last November and the major design started after the new year. We also started our first teacher training at San Jose State last fall.

Do you see the 90s as an era of invention, or are we massaging current technology? Are we explorers, or pioneers?

It depends on who you look at. The commercial world has never been into invention. They're just trying to get by and make money. One of the things that has been detrimental in the commercialization of computer technology—starting in the early 80s—is the lack of pure research. Its hard to keep good people in the universities when you can make so much money writing applications software or working for a computer company. So the number of people doing really top notch research is less than it was in the mid-1970's.

What is getting the attention of these researchers?

Artificial life, the Danny Hillis stuff is really hot, some of the agent things and some work at MIT and Stanford. The biggest problem is that its hard to invent architectures and with the machines doubling in speed and capacity so rapidly the only things that help you are ways of reducing complexity. Right now I don't think we're doing so well. I think that computers are growing in capacity much faster than our ability to either invent or adopt architectures that help control that capacity.

What's going to happen with all that computer power? What's going to happen with user interfaces as computer zorch continues to increase?

Well, I think a number of things. Parc has developed a beautiful flat screen with great resolution which is the first one I've seen that you can really read from. The user interface is always a problem because once you get hooked up to a network, you're hooked up to maybe half a trillion potentially useful objects. That's where this idea of agents—which goes back nearly 30 years—comes in. You can't directly browse so you have to have an information utility, you have to have surrogates that are able to take on some of the role structure and ferret around in the information space. I think we're fairly far behind on this.

You've been right a lot of times and several of your visions have come true. What are some of your visions for the future?

The best way to predict the future is to invent it. Let's not worry about what other people are going to do. One of the ways Gordon Moore made his prediction work is that he became one of the founders of Intel. He was one of the inventors of the technology he knew would work.

You have to predict what the real problems are versus just engineering better versions of what we have today. We looked at user interface because we realized that with the real problem of having lots of computers, the user interface could not be what it was on a time-sharing system.

That wasn't even thought of as a problem in the 60s. If you had a few thousand people that needed to know how do something, you'd just train them. With millions, they will have to learn from the user interface. Why do we think agent-based interfaces are going to happen in the next 5 to 8 years? The answer is that there is nothing that we can conceive of right now to handle our dealings with networks that won't require autonomous processes to carry out our goals.

One predicts the future by trying to understand well ahead of time what the new problems are going to be, not by making better versions of the old things. That's where a paradigm shift comes from. To me, the trick of the future is to figure out the things that are going to be obvious ten years from now but don't come out of any current problem. ■

Karen A. Frenkel, an author and science writer specializing in high technology, has written feature stories for a number of magazines including *Communications of the ACM*. Now, as Producer, ACM Special Projects, she is making a documentary on women and computing. *email:* frenkel@acm.org

Our Latest Seminar Turned into a Real Zoo.

Designing a graphical user interface can be tough. In fact, as our latest group of seminar attendees found out, it can be a zoo. That's because even the most experienced software developers have received little or no training in GUI Design. Well here's your chance to learn from the experts.

Give Your Software a Human Factor

At Acacia Design's intensive 3-day seminar, "User Interface Design: a Practical Approach," we'll give you the skills you need to perform this vital task. The key is adding a human factor to your software. Combining the skills of a psychologist with those of a graphic artist, you will learn everything from user and task analysis, to rapid prototyping, to layout, color and sound.

Learn from a Practical, Hands-on Approach

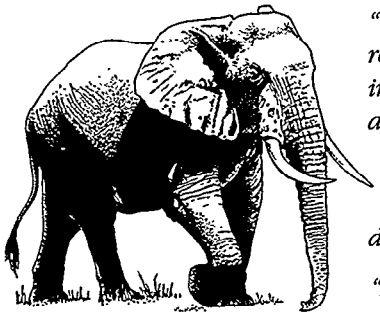
In an intimate, small group setting, you'll learn by doing. You'll be critiqued by us (as well as your peers) as you tackle a design project from start to finish.

- Hear from leading experts
- Understand the principles of design
- Design an application
- Test your interface on real users

We'll make **you and your software** more marketable!

No Holds Barred

This isn't a passive seminar. Crazy things happen. And that's okay; that's how you learn. Even the wildest ideas (a metaphor based on zoo animals came up at our latest session) can be explored and tested. The whole point is that you will leave with skills you can put to use the moment you return to work.



Proven Performance

Attendees consistently rate our seminars "excellent." Here's what just a few have said:

"A great job of imparting knowledge. The source material was both impressive and helpful."

"An excellent instructor. Open minded, articulate and knowledgeable."

"Workbook provided useful references to source information, principles and processes."

"Content is applicable to every aspect of the developmental process."

"Excellent and great fun!"

"...loved the Two Golden Rules...want to bring this seminar to our entire company."

Learn Now!

Acacia Design, consultant to some of America's most prestigious software firms, is now scheduling its spring and summer seminars. Sessions are reasonably priced and information-packed. To learn more about these exciting 3-day sessions, return the coupon below.

To find out more now, or to reserve your space, call:

1-800-964-9325.

FREE Preview/Reservation Package!

Return this coupon today for more information, an enrollment form and a complete schedule of Acacia GUI Design Seminars.

Name _____

Title _____

Address _____

City _____ State _____ Zip _____

Phone _____

I am interested in:

❑ East Coast Seminars ❑ West Coast Seminars ❑ On-Site Seminars

❑ Include information on local accomodations and attractions.

Sign up Now for our May 13-15 Boston Seminar! Class size is limited. Act now!

acacia design, inc.

One Kendall Square Suite 2200
Cambridge, MA 02139
A Microsoft Solutions Channel Member