



Jef Raskin Information

DOCUMENT TITLE

"THE MAC AND ME: 15 YEARS OF LIFE WITH THE MACINTOSH"

SOURCE

JEF RASKIN JUN 1994

DOCUMENT NO. 19

EX LIBRIS
DAVID T. CRAIG
736 EDGEWATER, WICHITA KS 67230 USA
E-MAIL: 71533.606@COMPUSERVE.COM





Apple Macintosh Computer
Historical Information

Document Version Catalog

Jef Raskin's

The Mac and Me: 15 Years of Life with the Macintosh

1994 draft version

August 1995 version (I of II)
The Analytical Engine 2.4

May 1996 version (II of II)
The Analytical Engine 3.3

(Ex libris David T. Craig)

Jef Raskin's

The Mac and Me: 15 Years of Life with the Macintosh

1994 draft version

Note
Missing punctuation characters due to the wonderfully compatible internet and email systems

THE MAC AND ME:
15 YEARS OF LIFE WITH THE MACINTOSH

Jef Raskin / 8 Gypsy Hill
Pacifica CA 94044 USA
Phone: 415-359-8588 / Fax: 415-359-9767
Internet: JefRaskin@aol.com 37° 3737 N 122° 3110 W

-- DRAFT -- 1994 --

TABLE OF CONTENTS

INTRODUCTION

The success of the Macintosh can be credited to no one person. I gave it its human-oriented, graphics-based, compact-size nature from the very first and made many other decisions that proved to be fundamental to its success. I hired a crew of unknowns who have become, almost every one of them, men and women whose names are known throughout the industry for their continued innovation. It was not just me, but my original Macintosh crew of four, then a dozen or so, and finally hundreds of people who created that first Macintosh. Now thousands at Apple continue to create and expand the Macintosh line of computers and the machines that will follow in its footprint. And even with all this effort, it would have been a dead-end product without the work done by tens of thousands of software developers that give the tens of millions of Macintosh users the tools they need. In this logarithmically spiraling cascade of numbers we come today to something over a hundred million people who useat their desks at home or at work, in their schools and libraries, at the beach, in airplanes, everywheresystems that look and feel much like a Mac. Amplified by all this effort and the sincerest form of flattery, the influence of the Macintosh may well have touched the lives of one percent of the worlds population.

The phenomenon that I have just described represents the expansion of one person's perspiration into a flood, but the stream had to first gather force from numerous tributaries. It was not just my own inspiration, but the flowing together of the genius of Ivan Sutherland and Douglas Englebart, the scientists at Xerox PARC, the development of the microprocessor, the success of the Apple II, the efforts of many other people whose work I studied and learned from (I will never be able to thank them all)and a lot of luckthat led to that one nexus in space-time, in the spring of 1979, when I went to the CEO of Apple and told him that I wanted to design a new product, and that I wanted to call it Macintosh.

This year, 1994, the 10th anniversary of the introduction of the Macintosh, is being celebrated with a rash of articles and parties at Apple and elsewhere. But it is also the 15th anniversary of the creation the Macintosh project. This is the story of how I came to originate a product that has changed the face (and interface) of computing.

THE HUMAN-ORIENTED COMPUTER SCIENCE STUDENT

Its hard to say when the idea for the Macintosh computer began. Parts of it be discerned as early as 1965 when I was a graduate student in Computer Science at Penn State. Already steeped in the technicalities of computer design and programming, I nonetheless thought that using computers was aggravatingly and unnecessarily difficult. Always seeking ways of simplifying their use, I soon earned a reputation as being especially sympathetic and helpful to our least technical users, especially those in the arts and humanities. My sympathy to their plight was aided by my feeling as comfortable in the humanities and the fine arts as I am with science and mathematics. I never forgot their pain (or mine) at having to put up with the nonsensical ways computers were and are operated. By way of contrast, most of my fellow students celebrated their detailed knowledge and seemed to enjoy the power and status it gave them with respect to ordinary users. They preferred to work with hard-nosed programming students with whom they could attack problems in full jargon.

In my 1967 thesis, The Quick Draw Graphics System, I wrote that computer displays should be graphics- rather than character-based and that ease of use should be put ahead of software and hardware speed and efficiency. My point of view was radical: in those days input was generally via punched cards and output came in the form of extravagant quantities of paper from massive printers. That visual displays would be the major output device for computers some day was not then commonly accepted. Computer time was expensive, and the idea that human convenience be given pride of place was equally foreign, and far ahead of its time.

I liked pranks: appalled by the daily waste of paper a few friends and I once wrapped the computer center building with one days discarded output. On a day a dignitary was visiting, I set up the computer so that opening the massive printer cover (done by a remote command) would dump a wastebasket of the punched-out paper chips from a card-punch into an air vent, giving the effect of a short-lived snow storm. My friend and office-mate Steve Zins and I engaged in a rubber-band gun arms race: my best designs were single-shot guns of unprecedented accuracy, needed to shoot the flies generated by the adjacent cow fields. Steve created a gatling gun that could plaster my chest with rubber in a fraction of a second. But I digress...

TO THE WEST COAST

The first truly interactive graphical computer system was Ivan Sutherlands Sketchpad, and though the hardware available at Penn State would not allow me to follow his lead, Sutherlands work (which I learned about at Penn Sate) was a revelation and an inspiration. I began to dream about a computer that would be easy to learn, easy to use, do everyday tasks (such as word processing) and, above all, be affordable. At the time, this was simply an impossible dream. Tired of Pennsylvania winters, my wife Karen and I drove west, stopped only by the Pacific Ocean. We ran out of land in La Jolla, just north of San Diego. We knew nobody in the area, and just happened upon the University of Californias Scripps Institution of Oceanography. We walked out onto the Scripps pier where I saw, for the first time, a pelican abruptly fold its wings and splash into the ocean. I thought it had been shot.

My feminist leanings were confirmed by some of the things that happened to Karen. For example, when we were undergraduates at S. U. N. Y. at Stony Brook, I held a job in the computer center. Karen, independent of spirit as always, did not deign to learn programming from me, but took the regular course. The head of the computer center there had offered jobs to the people who got the three highest scores on the final. She got the top score and we looked forward to working together. The jobs, however, went to the three top men in the class! We were mad. She raised a ruckus, and I quit in protest.

We went to Penn State next. I got a job, as usual, at the computer center and, as usual, they wouldn't hire her in spite of impeccable credentials (better than mine in some wayssuch as grade point average) because her boy friend worked there. After we were married, there was no way she could be hired. Nepotism, you know.

When we reached the west coast, having graduating with a master's degree apiece, we finally got smart about jobs. She took a position first, running the computer at the Institute for Geophysics and Planetary Physics at University of California at San Diego (UCSD). Shortly thereafter I got a job at the University Computer Center. They didn't think to ask a man if his wife worked in a professional capacityon the form it only asked if your husband worked at the University. My brilliant cousin Miriam also had to face many sexist hurdles on her way to becoming a professor at Ann Arbor. This all might seem irrelevant, but they helped lead to the name Macintosh for my computer project.

While working at the UCSD computer center I became familiar with the school. The music department, filled with avant garde composers and performers, was open to my interest in computers and music. It looked like a good fit and I became a graduate student there, working toward a Ph.D. in music. Like some English universities, UCSD was divided into colleges. The first two were named Revelle and Muir colleges. The newest was simply called Third College at the time. By a peculiar series of events that would (for once) take us too far afield, I soon became the Third College computer center director and a professor of Visual Art, positions I held from 1969 through 1974.

The noisy main computer center at UCSDs Revelle College was antiseptic, lit by florescent lights that glared off white vinyl floors, and was punch-card-oriented. It was built around a physically huge, multi-million dollar mainframe computer. The computer center I designed for UCSDs Third College was very different. It used a pair of Data General Nova minicomputers with 16 interactive terminals. The decor was bean-bag chairs and Japanese paper lanterns, giving my center a friendly, funky feel. Located in a war-surplus Quonset hut, it became the natural home for people with what were then seen as odd computer applications, like music and art. Some of the more avant-garde computer aficionados found that they preferred the calm, interactive minicomputers to the florescent, buzzy mainframe environment on the other side of campus. Given todays personal computers which operate in homes, it is hard to remember that in the early 1970s a computer center such as the one I created was countercultural, and perhaps unique.

THE THIRD COLLEGE COMPUTER CENTER

My computer center was funded primarily by the NSF (National Science Foundation) and the University of California. I suspect that if Senators Proxmire or Helms had ever visited it they would have pointed it out as a typical waste of taxpayers money. It looked more like a place to get stoned than to get educated, a hippy haven.

But it worked well as an educational facility. Though only 10% of the undergraduates who learned programming at UCSD went through my courses, 50% of the students who held paying jobs at the main computer center had done so. There was no doubt that the Third College computer center was doing a good job at creating future computer scientists. Better still, it was also reaching students who would not otherwise have gone near a mainframe, just as the Macintosh would someday be used by people who thought theyd never touch a computer. Fortunately, Ken Bowles, the computer center director at UCSD, did not regard a second student-oriented computer center as a challenge to his hegemony (an enlightened attitude for a computer center director at the time).

In retrospect the project was all that a grant administrator could wish for: it met its educational aims at the time, resulted in appropriate publications, and later went on to inspire commercial products that have aided the GNP (gross national product) to the tune of billions of dollars. It is definitely possible to see hints of the Macintosh in the third college computer centers low tables with small rectangular monitors and detached keyboardsthough they had to be tied to a common system, the effect was as if each student had a personal computer. In 1973, when a 4K byte memory unit cost over a thousand dollars, resources had to be shared. As this is written, each 4K bytes in my Mac PowerBook 180 computer cost me about 14 cents.

During the summers, I used one of the Novas as my personal computer. My mostly volunteer staff and student friends (notably Jon Collins, Barbara Zakarian, Bill Atkinson, and Steve Clark) helped me put the computer into the back of my truck on a wheeled dolly, and we used it wherever we went. One memorable time we took it with us into a restaurant, using it to figure the bill and the tip, to the amazement of the waitresses and patrons, who crowded around. A computer outside of a lab was an absolute novelty. These experiences with a portable computer system gave me a foretaste of what it would be like to own a personal computer. Like the crocodile in Peter Pan, I would never forget that taste, and craved it for years.

It was also at about this time Karen decided to get a second masters, this time in Library Science, a subject she had long wanted to study. She went to UCLA to do so. I tried commuting for a while to our rented house in Venice, but with time together diminished and distance increased we found ourselves drifting apart. Eventually we got a very amicable divorce. This turned out to be a major emotionally trauma even though completely without strife, disagreement, or animosity. In retrospect, though our affection for each other was very real, we probably would never have married but for trying to live together in a small Pennsylvania town where the social pressures for marriage were strong. We each later remarried and started families and, as luck would have it, now live a half hour drive from each other.

SAIL AND SILICON VALLEY

In 1972 I visited the Stanford University Artificial Intelligence Laboratory (SAIL), which had an established reputation as a center for advanced research in computer science. I was also introduced to another magical place that was just a short bicycle ride away. A few years old at the time, the Xerox Palo Alto Research Center (PARC) was to become even better known than SAIL. With a strong interest in early music as well as computers in common, I soon found a close friend in the person of Doug Wyatt, a tall, thin man who is as quiet as he is technically brilliant and musically talented. Doug took leave of absence from PARC and came down to San Diego for a while to write new software for my computer center.

A programming language I designed, called FLOW, was implemented (and improved) by Doug. The human interface and language used in this system were somewhat ahead of their time, so much so that it came to the attention of the cognitive psychologist Don Norman, who was becoming a leader in the fields of cognitive psychology and man-machine interfaces and a writer of popular books on the subject. Norman did some of his first computer-interface-related work investigating why students learned faster and better with the FLOW computer language.

The next summer I was invited to become a Visiting Scholar at SAIL. It was a great place to be. I remember fondly the memorable daily volley ball game, after which most of us would retire to the lounge and watch Star Trek.

A number of experimental robots roamed the halls and parking lots and one, being tested in an autonomous mode, escaped and was found a few hundred feet down Arastradero road. The escape was discovered when someone noticed an automobile going by in the picture the robot was transmitting back to the labs state-of-the-art PDP-10 computers. The robot, in case you were picturing C3PO walking along the center line of the road, looked like a wheeled tables full of surplus electronics and not at all humanoid. On another occasion we were sitting watching Star Trek when a robot wandered in, stopped, swivelled its TV eye at the set and sat there throughout the show. At the end, it turned and left as we did. Later I learned that Hans Moravec was working at a terminal that did not have TV feed (most of them at the AI lab did another development way ahead of its time) and had sent in the robot to beam the picture and sound back to his monitor. It occurred to me that we were the probably only people on earth watching Star Trek in the company of a robot.

While at the AI lab I used the early defense department progenitor of the now-popular Internet. ARPAnet allowed us to communicate with and use remote computers. It felt like magic to be sitting in California and running a computer at MIT. I became an early e-mail junkie, a habit of 20 years that I have yet to kick.

PARC

The populations at SAIL and PARC intermingled freely and I found myself gravitating more and more toward PARC. Their work was based on the same goals as my own, to make the power of computers accessible to non-specialists. I suspect that I fit in easily and well because they didnt have to first convert me to their point of view; I was already there. I felt at home since for the first time I was among computer scientists who were on the same wavelength as I. They had accomplished (independently) what my thesis had called for a few years earlier: computers that were graphic-based, without impediments such as character generators. What was more exciting was that the people I spoke with were concentrating on interface design, which I saw as the area of computer science most in need of development.

By 1974 I was fed up with the politics in the UCSD art department and left the University, making my point in artistic fashion by ascending in a huge hot air balloon, playing the sopranino recorder, and announcing my resignation from 100 feet up in the air. I was also tired of the directions the computer industry was taking. It was all more and bigger and faster (this history is becoming repeated with personal computers today). Nobody seemed interested in what I was preaching about usability. I sold my house near San Diego and moved to Brisbane, a town just south of San Francisco. I tried the life of a street musician and music teacher, started a company that made radio-controlled model airplane kits (a business that continues today), and became the conductor of the San Francisco Chamber Opera. I also worked briefly as a packaging designer, but left when I couldnt convince the owner that we could design boxes faster and better with a computer. In the 1990s the owners son, who better understood what I had proposed, wrote a set of computer programs to do box layout and now has a successful business making boxes and an even more successful one selling the software.

I also worked as an advertising and portfolio photographer (having been taught a bit about the art by my former student and forever mentor David Wing, now a professor of art who teaches photography at Grossmont College). Lastly, during this period of wandering, I started a company called Bannister & Crun to write software and manuals. The company was named after two characters (Minnie Bannister and Henry Crun) on the BBCs beloved Goon Show.

Our first job at Bannister & Crun was to computerize the South San Francisco sewer billing system, a job that required me to visit the sewage treatment plant from time to time and work on one of the most dreadfully designed computers I had ever seen, an early Qantel model. We also worked on other software projects and wrote manuals for companies that included National Semiconductor and Heathkit.

ENTER THE MICROCOMPUTER

In late 1974 the microprocessor was invented and I remember discussing its incredible potential with Doug and another musical friend, another talented and extraordinarily pleasant man named Brian Howard. Of broad learning, with a degree in Electrical Engineering from Stanford, he was working for the preventive medicine department at the university. He was doing a wide range of things (characteristically) including building test equipment. Brian was to become a central intelligence in the development of the Macintosh and, later, one of the designers of Apples first laser printers, another product that changed the face of computing. He continues as a respected engineer at Apple. When the first microcomputer kit, the MITS Altair, was announced in 1975, Brian, Doug, and I just had to have one.

With soldering iron, oscilloscope, and logic probe in hand, Doug and I built the Altair and (somewhat to our surprise) got it working. This was a non-trivial endeavor but I knew how to do this kind of stuff because electronics had been a hobby of mine as a child. I won a science fair prize for a computer I made while in high school. Building a computer is nothing to crow about now, but in 1960 individuals just didn't have computers and kids didnt use or program them. I was still a hardware jock as an undergraduate, designing and building a computer from scratch for the Biology department at the State University of New York, then at Oyster Bay (now Stony Brook). This background proved useful when creating the Mac, since I had a realistic idea of what could and could not be done with electronics. Most important, I could communicate with electronic wizards, and not be snowed when they spoke of impedance or logic levels.

We got the Altair running a program that did stock market analysis, and we sold it for about $5,000 to Jim Hurst, a stock market analyst and guru who had been paying $10,000 per month in time-shared computer charges. The computer had cost us a few hundred dollars and had served us well as an introduction to microcomputing. With part of our profits we bought a slightly more sophisticated IMSAI, and I built the first modem kit that became available. I made back a bit of the money I spent on that kit by authoring a review of it for Dr. Dobbs Journal. I liked reviewing kits and I was soon writing the Consumer Notes column for that magazine. I became a reporter on the early personal computer scene; pieces I wrote appeared in Personal Computing, Interface Age, the Silicon Gulch Gazette, Kilobaud, Datamation, and Byte magazine.

Jim Warren, who ran the wonderfully named Dr. Dobbs Journal of Computer Calisthenics and Orthodontia (Running Light without Overbyte), is a delightful, jovial, and unconventional man who sparked much in the industry, including the West Coast Computer Faires. He often managed these affairs by cruising the huge halls in which they are held on a pair of roller skates. One of the assignments he gave me in 1976 was to interview two fellow-members of the now-legendary Home Brew Computer Club. The club, many of whose members went on to become prominent in the computer industry, was moderated by the very funny and genial Lee Felsenstein. This was also the man who designed the modem I reviewed (its a small valley). Doug Wyatt and I got an ovation one night when I announced that we had run our IMSAI for over a month without once taking the top off to fix something; it was a real milestone (and a tribute to clean soldering).

The two members I was sent to interview were building a new computer in their garage. By coincidence both were named Steve and their project was the Apple I. I was very impressed by Steve Woz Wozniaks brilliant and efficient design and pre-decoded bus concept, and his exposition of the advantages of the 6800 and 6502 architecture over the competing 8008 and 8080-based machines. (Incredibly, this competition between architectures continues to this day). I remember Woz explaining that the predecoded bus made peripherals simpler, that you could send information to peripherals the same way you wrote to memory, and that memory wasnt pagedunimportant details in this essay perhaps, but indicative of the kinds of considerations that Woz paid careful attention to. And I loved the name Apple instead of the techie names everybody else was using; it fit my kind of iconoclastic spirit. Now we have become so accustomed to it that it is hard to remember how jolting that name was at first.

The other Steve, Steve Jobs, was a delight to talk to about less technical aspects of computers. His enthusiasm and business orientation were exciting. They were just starting on the design of the Apple II, and I tried to convince them that they should use all bit-mapped graphics and not have a character generator, but Woz thought that software couldnt handle the character generation task fast enough and Steve Jobs didnt understand why I thought it so important. I had a different vision of what a microcomputer should be like, and PARC had shown me that software could do the job. I tried to convince Woz by working out the code to put bit-mapped characters on the screen and calculating timings by counting cycles, but the Steves were not open to the idea. The concepts I espoused were far from the mainstream of computer design and for all their mold-breaking thinking, Steve and Steve were very strongly conditioned by the minicomputers they had seen. To do them justice, if you did not understand the vision I was promoting then Woz was absolutely correct in stating that a character generator

was much faster and its software less memory intensive. He won this round. The Apple II almost didnt have its Hi-Res graphics mode but Jobs asked Woz how many chips it would take to add the feature and Woz said that it would take only two, so Jobs insisted that they could afford it.

I tried to convince Jobs and Woz to visit PARC, which was a very academic and open place (Xerox may have later felt that PARC was too open), but did not succeed. Jobs repeatedly told me (and anybody else he could get hold of) that a large corporation like Xerox couldnt do anything interesting. The rejection of Wozs proposal for a personal computer when he worked at Hewlett Packard was a prime example of such corporate blindness, and ever remained part of their psychological motivation. If I could have told them then that Hewlett Packard would someday be making millions of dollars by selling mere peripherals to Apple computer products (as has happened), the Steves would have been ecstatic, and rightfully so.

APPLE MANUALS

I worked with Jobs and Woz and, under the aegis of Bannister & Crun, wrote the user portion of the manual for the Apple I. There was a tiny misunderstanding about the price: I was talking about $50 per finished page and they thought I had said that it would cost $50 for me to write the whole manual. We resolved our differences amicably and work proceeded.

It was about this time that Jobs made a key decision in the history of personal computers and insisted that Apple take a suggestion made by retailer Paul Terrells. Terrell ran the Byte Shop, one of the first retail computer stores in the world. When Jobs and Woz asked his advice he said that the Apple II must have a non-square, consumer-oriented, plastic case andunlike the Apple I and many of its competitorsmust not be sold as a kit for which you had to scrounge parts at the local surplus or electronics store. It should come complete with keyboard, power supply, and be factory built. Terrell was not quite alone in recognizing this desideratum, the SOL from Processor Tech also had the typewriter look. Another company, Sphere, had a complete little machine with a programmers hexadecimal keyboard (base 16 numbers only) and a video screen even before the Apple I was released. The Commodore PET and TRS-80, both well consumerized, followed along soon after the Apple II. it was not alone. But Apple had advantages. One was that Woz wrote a remarkable BASIC interpreter with color graphics commands embedded in it. Another was that it was led by a raving firebrandwhich was just what the industry needed at that moment.

Bannister & Crun was hired to write the manual for Apple IIs BASIC. This gave me the opportunity to put some of what I had learned as a teacher into a vehicle that I believed would reach tens or hundreds of thousands of people in a few years. It had been hard leave teaching, which I love, but I felt that I could do more good for education by working at Apple than in any other way open to me.

The BASIC manual, first published in 1978, turned out to be a trend-setter. Instead of starting off with the then-customary explanation of the internal architecture of the computer, it got right to what people had to do to get the product working. It first explained in a step-by-step manner how to hook up the computer and use the keyboard. It then quickly moved the learner into doing color graphics (in 1978!). Another first: the manual used color illustrations and photos. It was actually difficult to convince the nascent company that it needed an Apple II manual at all. Mike Scott (Scotty), a large man whose sometimes high-handed manner and gruff verbalizations could be intimidating, had come from National to be Apples president. He said, half seriously, that at National they had done very well with one-page data sheets, and I could save the company a lot of money by doing likewise. I soon learned that in spite of his manner, he was open to cogent arguments, and later he was to protect and nurture my Macintosh project when it was at a delicate stage.

Creating user documentation was a perfect prelude to creating the Macintosh at Apple. Writing manuals forced me to look at each product, in excruciating detail, from the customers point of view. It is an experience I wish all computer and interface designers could share. Any design flaw that interferes with learning or using the product becomes painfully apparent as you struggle to explain the oddity to the user. Brian Howard and I would often struggle with these problems, our frustration sometimes coming out as subtle but snide remarks about design errors which then appeared in our manuals. These comments, which sometimes annoyed marketing, actually worked to make customers like Apple products better. They told the truth, showed sympathy for the customers plight, and created credibility for the rest of the manual and the company as a whole. Too many manuals, then and now, are fairy tales about how the product was supposed to have worked.

When we wrote the Apple II manuals at Bannister and Crun the product was finishedwe werent trying to write a manual from specifications. Trying to document a product still in development is an often-made mistake which guarantees a second-rate result. Since almost everybody now does this, customers have come to accept such manuals as standard. We werent the only group writing good manuals, and I learned from the HP 35s manual. The HP 35 was the first scientific pocket calculator, another fabulous product that started an industry. Its manual was an inspiration in terms of writing, use of color and layout, and informal conversational style. There was nothing about the calculators remarkable stack architecture or revolutionary custom electronic chips, the manual started you out punching buttons and seeing what happens. Later, when it had some value and experiential basis, you were given a mental model of what was going on inside.

Another reason the Apple II manuals worked is that they were tested with typical users and rewritten as necessary, a unique concept at the time, and one that Apple has largely abandoned, to the detriment of its customers. I demanded that I have a real product in its real packaging before I wrote a manual, and in those early days I got what I asked for. Errors were extraordinarily rare as a result of these procedures. Brian Howard turned out to be a great editor along with his other talents,

and his comments (and those of Doug Wyatt) were my education in how to write clearly and simply. My writing has never achieved the standards they set, but inasmuch as it is better than it was, they deserve a lot of credit.

APPLES MANAGEMENT

Having already approved the use of high-quality, coated paper, color illustrations, two-color printing throughout, and full-length manuals, management was reluctant to support the use of a wire binding so that the manual would lay flat: I had often observed that most users didnt have a third hand to hold the manual open as they typed. All during 1977, when I was merely a vendor to the company, two key people supported my point of view: co-founder Steve Jobs and chairman Mike Markkula, also known by his initials ACM.. Markkula had profited from his experience at Intel in two ways: financially and in management expertise. Not having been in industry, I had never worked with a person of his extraordinary business skills, and I am still striving to live up to some of the examples he set. For one, he always gave me the impression that he had all the time in the world to pay attention to what I had to say. He proved that he was listening by acting on my suggestions or taking the time to explain to me why they were not good ideas. One of my not-very-good ideas was to lower the price of the Apple II. I had been upset when I figured out how large Apples margins were. Markkula patiently explained that while Apples products were more expensive, the resultant financial strength of the company meant that Apple would be there for its customers in the future. It would have the money to develop new products and successfully market them while its competitors, who seemed to be doing a favor to their customers in the short term, would soon be out of business. He was right.

In the early days of Apple Jobs was an adamant protector of my writers prerogatives and understood the need for testing and revision when Scotty didnt see things my way. In the end, I did the manuals as I thought they should be done and Apple got what the press and even other companies praised as the best manuals in the business.

DEPARTMENT BUILDING

By mid 1977 my writing had made me pretty well known in the industry and I had lots of job offers. Chuck Peddle at Commodore (he led the PET computer project) wanted me for a position at Commodore. Steve Jobs, who is as persistent a person as Ive met, kept on asking me to join Apple as head of their publications department. I repeatedly declined, and he eventually asked what I needed to join Apple. To put him off I made an impossible list which included an office with a window and a musical instrument, time to play gigs (I didnt want to let my musician friends down), flexible hours, hiring everybody at Bannister & Crun who wanted a job at Apple, and so on. He simply agreed to all my conditions, which I reduced to writing; as it turned out, I should have done this with more of his promises. Bannister & Crun became Apples publications department with me at its helm. I joined on the 3rd of January, 1978, the very day that Apple Computer became a corporation. I presented no resume and signed no forms. Apple did no checking on my background. That I had led a team that had produced the industrys best manuals was enough.

One day I heard that a new product, called the Apple II Pro, was being put together in the lab. From some technical details, I realized that it could not possibly work as expected. So I snuck into the laboratory and turned on the prototype. Sure enough, it didnt do what it was supposed to. I went to Mike Markkula and told him that the machine wasnt up to snuff and he told me that I couldnt be right, his engineers had assured him that all the problems had been solved. In fact, he was making plans for marketing and shipping the product and was about to start taking orders from dealers.

I took him to the lab and demonstrated my discovery. Upon talking to the people working on the project I had discovered what I later learned to be a classic pattern: the low level people said that there were some problems, the next level reported to their bosses that there were a handful of problems that would be rapidly fixed, and they in turn told Scotty that it was nearly done, and Scotty told Markkula that it was just about ready to roll. In a more mature company I would probably have been fired immediately, but in this case I was able to make a case for a New Product Review department. This would do for systems and software what QA (Quality Assurance) programs did for circuit boards and mechanical assembly. Suddenly, I was managing two departments.

Shortly afterward I argued that we would need to provide something new, application software, if we were to sell computers more widely. I created the application software department, which may have been the first at any microcomputer company. I tried to convince Apple to buy Visicalc, the first spreadsheet, when it was offered to us, but was outgunned by Jobs and Markkula. It was Markkulas theory that to become a major application provider would have put a damper on third-party software developers, in the long run hurting Apple. I thought that having a big winner would not only be profitable but would also be a spur to programmers to go and do likewise. I took a brief leave from Apple, arguing that my helping make Visicalc a winner would sell a lot of Apple IIs. The leave was granted, I worked with Dan Fylstra to write the tutorial portion of the Visicalc manual. Visicalc did sell a lot of Apple IIs, while establishing a new category of product.

In 1979 I found managers for two of my departments and became manager of Applications Software. Meanwhile, I was chafing at the limitations of the Apple II. In fact, the publications department was using Poly 88 computers running a word processor I had designed and which we had implemented at Bannister & Crun. The Polys were a competing microcomputer that we were using only because they could handle both upper and lower-case lettersa necessity in manuals. Due to mediocre design (they had no Woz), poor marketing and management, they were soon out of business. Back in the garage days, in 1976, I had argued that the Apple II must have lower-case letters, but Woz disagreed. I held that the single biggest use of microcomputers would be word processing, he claimed that they would be used for game playing and programming in BASIC. But he had the ultimate weapon in this argument: upper-case-only character generators were lots cheaper.

Later I would often find myself at odds with the founders, who were an odd mix of the radical and the conservative. They wanted to create personal computers, but they also wanted them to work much like the hard-to-use minicomputers from DEC, HP, and Data General. Dragging them into the interface future was swimming upstream. From my point of view they didnt look nearly as visionary as they were made out to be in the press. They were visionary, and working like mad to drag the world into the personal computer future, I was just a few years further out in the future. In spite of these differences I was a way-over-100%-effort and totally Apple-oriented employee. For example Apples Cupertino phone number was 408-996-1010. When I moved to Cupertino, I chose my home phone number, symbolically, to be just one step ahead of the rest of Apple: it was 996-1009.

BITMAPPING

When Ken Rothmuller was hired from HP to start the Lisa project I saw a new opportunity to get my computer interface and architecture ideas accepted. I argued again that the screen architecture of this new product should be bit-mapped. But where I had failed with Woz and Jobs, I managed to convince Ken and his crewprobably to Kens detriment as Jobs found him difficult to work with (i.e. had strong opinions and didnt kowtow) and fired him. Jobs probably found me equally difficult, but I had already proved myself and my publications department was one of Apples many gems.

And Apple had a diadem of the brightest and best cut jewels of Silicon Valley, some well known and some newly discovered. Whether it was in financial management, marketing, manufacturing, engineering or whatever, I was amazed at the competence of the people. And all seemed willing to share their knowledge and points of view with me. Competence clustered at Apple, partially due to the many contacts men like Markkula and Scott and our investors had in the industry, and partly due to Steve Jobs incredible persistence. When Jobs was convinced he wanted someone, that person would be hounded to death, complimented, provided blandishments suited to his or her nature, and offered the world. And soon enough, Apple could deliver the promises.

In spite of the loss of Ken Rothmuller, the bit-mapped screen survived. This was a key win for me and (though they didnt know it at the time) for Apple because it would force the kind of software I was dreaming of to be implemented. No longer would computers be confined to whatever font was in the character generator and have to treat characters and graphics as fundamentally different kinds of things. Another major battle that I fought was to have black characters on a white background instead of the then-conventional white (or green!) lettering on a black background. The Lisa hardware designers, like Jobs and Woz, were dead set against this idea, noting that it took too much power, would require a higher refresh rate to avoid flicker, was not the way computers usually worked, and so on. I argued that people often printed computer output on white paper, and that was black-on-white, and that if you wanted it to look the same on screen and print (the WYSIWYG, or What You See Is What You Get principle) you had to do it black-on-white. But my industrial-strength argument had to do with something the Lisa crew (like the whole microcomputer industry) was just not thinking about at first: graphic images. If you worked in white-on-black and had a picture on the screen which got reversed on printing, then either the screen image or the paper image would have to be a negative, and nobody wants to be forced to look at negatives.

Again, after many memos, meetings, informal and formal discussions, I managed to sell the idea. It was another key to the future.

ANTI-MICRO ATTITUDES

The computer industry in the middle 70s tended to ignore or minimize the microcomputers (they were not usually called personal computers at the time) that I saw as the future of computing. Nonetheless, I was invited to chair the National Computer Conference session on documentation in 1979but this was mostly on the basis of my presence in the large-computer world. At first, microcomputers were turned down when they asked to exhibit. By 1978 they were given a room in the basement. A few of my friends at the large computer companies asked me why I was throwing away my career by working for a microcomputer company.

At one of the National Computer Conferences I was on a panel where I was expected to uphold the proposition that microcomputers were useful. Many mainframers thought and said that they were and would remain toys. We each gave our little talk, but I didnt score until the discussion session. One of the speakers, to show the superiority of large computers challenged me to some benchmarks. The exchange went something like this:

> Anything your little Apple can do, my mainframe can do, and do it better, he boasted, For one thing, microcomputers dont have the speed of a mainframe!
> OK, I replied, name your speed benchmark.
> Invert a 100 by 100 matrix! It will take me about 40 seconds.
> You win, I conceded, it would take my machine hours to do it. The audience gave a bit of applause for the main frame.
> My turn.
> My interlocutor gave me a disdainful look, Whats your speed benchmark?
> We both have to run across the hall. The person getting to the other side first, carrying his computer, wins.
> There was laughter as people pictured him trying to pick up and run with his mainframe, much larger and heavier than a refrigerator, and then there was a solid round of applause as I raised my Apple II with one hand.
> For my next benchmark, lets discuss power. Have each of our machines create an index to a thousand page book.
> I had to concede. My computer couldnt even hold that much text. This admission got a few guffaws from the audience.
> Then I proposed my second benchmark:
> You take $100 out of your salary every month and III take $100 out of mine. The person who can pay for his computer first wins.
> There was a lot of laughter and applause.
> But, argued my opponent, thats not computer power!
> A computer, I answered, has no power at all if you cant afford it.
> From the audience reaction, it was clear I had won the debate.

THE MACINTOSH PROJECT PRELIMINARIES

Early in 1979, probably in March, I talked about my idea for a new computer with Markkula. He had had an idea for a game machine, which he called Annie. He agreed that my project had more potential for Apple, and I announced that I would call it not Annie since I felt that the trend in the company to give new products feminine names was sexistand if you had spoken to the namers youd have had no doubts about their motives. I suggested Macintosh, naming it after my favorite kind of apple, the McIntosh. I changed the spelling in what turned out to be a vain attempt to avoid conflict with the name of an electronics manufacturer. Markkulas Annie project would, besides games, have allowed the user to program in BASIC but it was not intended for business applications. I counterproposed that any new product should be able to handle the much wider range of applications that I thought personal computers would be used for. I also said that using a TV set or a third-party monitor was playing Russian Roulette with one of the most important selling points of a system--how the screen looked. Markkula sent me off to do design and cost studies. Working with my friends, notably Brian Howard, I came back with an absolute minimum selling price of $1,000, far from Markkulas goal. It was based on the 6809 chip and had a 256 by 256 bit-mapped screen. I designed a proportionally-spaced character set that would display 25 lines with an average of over 80 characters per line on the little display. To put this into the perspective of the time, the Apple II displayed only 40 upper-case characters per line. The idea of proportional fonts on a display was unknown at Apple, though a commonplace at PARC. The 6809 was chosen due to the tight price constraint imposed initially by Markkula. The better 68000, when it first became available a little later, was $400 in quantity. That would have made the product have an introductory price of about $3000. My original concept was biased toward the inexpensive and memory efficient, again because of cost. I noted that a 256 by 256 display could be addressed in exactly two bytes, making fast software easier to writespeed is of the essence in a good interface.

To convey one of the Macintosh design features to others in the company, I built an Apple II with a monitor incorporated into the lid. I used it at lectures and demos and it had great appeal wherever I demonstrated it. To this day I do not know why Markkula, to whom I pitched the idea the strongest, Jobs, and all the other people in management resisted the idea. The Apple II had a pop-off lid, and we could have sold a replacement lid with an angled CRT built in. My very happy experience with this prototype settled the idea that the first Mac would have a built-in monitor.

While the company was thinking about manufacturing tens of thousands of computers a year (another unheard-of idea) I wrote a paper, later published, called Computers by the Millions. In it I looked at questions of design, manufacturing, marketing, and a few of the social and economic impacts of computers in those numbers. Management found the paper valuable, and would not allow me to publish it for a number of years to avoid letting the competition know what we were thinking.

Jobs, unaccountably, did not approve of my presenting my views of the future. By doing so and by proposing new products independently of him, I began to get on Jobss wrong side. By this time Jobs had begun to have (or I had begun to notice this trait in him) people who were in and those who were out. If you were in, everything you did was golden, and if you were out, everything you did was rotten. Thinking I was still in, I kept on trying to get Jobs to go to see what PARC was doing but since I was actually out, he resisted the idea more strongly than ever.

PASCAL

Early in 1979 a major part of my efforts were devoted to convincing the company that we should move away from using BASIC and assembler as our main languages for applications and system software. I argued that we should base our work on Pascal. A clever and inventive ex-student of mine, Bill Atkinson, who I hired for this project, implemented a Pascal developed under Ken Bowles at UCSD. They had it running on the 6502 processor, the same processor used in the Apple II and Atkinson suggested porting it to our product. In the process he had to write graphics routines, a bit of experience that proved extraordinarily valuable for Apple. Many in the company had rejected PASCAL as impossible, in spite of a number of technical memos I had written showing how it could be done. As Atkinson said, We had a bunch of self-trained amateurs who didnt really understand modern software development. The system software team actively resented a new language. Once we had it up enough to demonstrate the word processor and Markkula saw that, it was clear sailing. I supported Bills implementation then wrote a PASCAL manual with Brian Howard. Pascal, as I had predicted, allowed us to hire more professional programmers, and later became the main development language for Lisa and the Mac. As a side light, I personally paid a license fee to UCSD so that Apple could use their Pascal system. Apple never reimbursed me since Jobs turned down my request on the basis that Apple didnt need and would never use Pascal. Almost all Mac and Lisa software was written in Pascal, I was amused by the thought that in some vague sense, it was all owned by me.

THE MAC BECOMES OFFICIAL

By September, 1979 Mike Markkula had, over Steve Jobss objections, approved the project. By going around Jobs I had unknowingly finished setting up a dynamic that made the project far more difficult politically than I could have anticipated. From the first, Jobs opposed it, calling the Macintosh the dumbest idea hed ever heard of. He would often recite a list of imagined advantages that the Lisa project had over the Mac and put obstacles in the way of my obtaining staff or supplies for it. His interference eventually became so overt that Mike Scott had me move the entire Mac project to some buildings behind a Texaco gas station across De Anza Blvd. so that we would be able to develop the Mac in peace. Later, when Jobs took over the project he put up a pirate flag and claimed that he moved the project out of Apple headquarters so that it would remain pure and uninfluenced by the stogy company engineers. To me the pirate flag really indicated a pirate within: Jobs had taken over the project by fiat and lies andas I was to learn latertried to steal the credit for creating it.

From the beginning, to keep the project on track and so that we would not lose good ideas and the reasons for abandoning others in the press of development, I created a document numbering system and put the collected documents in the Book of Macintosh which grew to some 400 pages. I wrote most of the book, since I liked to write and was the fastest typist in the group, but the ideas were generated by everybody, and they were credited in the text. There was a standardized heading format, heres one example.

MACINTOSH PROJECT
DOCUMENT 18 VERSION 0
DATE: 20 OCTOBER 1979
TITLE: DELIMITING STRINGS
AUTHOR: JEF RASKIN

I asked all the participants to explain the reasons for their conclusions, their right and wrong turnings, as we went along. I ended up writing most of the documents late at night at home; for the most part everybody else was too busy to do so.

Jobs opposition was partly due to his not understanding what I was trying to accomplish, though at this time I still thought of him as the supportive friend he had been for so long. His opposition was often based on keeping the status quo. For example, when I insisted on bit-mapping and square dots, he would retort that Woz had put a character generator in the Apple II and it didnt have square dots and its sales were paying my salary.

INTERFACE INNOVATION

It has been often said in the computer press that the Mac was simply a straightforward ripoff of the work done at PARC. This is very far from the facts and does a disservice to the hundreds of people at Apple who developed the hardware, software, and interface concepts. This theory has also been rejected by the courts. Strangely, I was briefly hired not by Apple but by Xerox as an expert witness in this matter, but they soon learned that the main thing I could testify to was the originality of the work done on the Macintosh. The Lisa group did do some shameless copying of the Xerox Star, but that is a different story.

One of the major ideas that led to a very different overall appearance to the interface and the way it felt to use the Macintosh was the one-button mouse. The one-button paradigm has become so pervasive that many applications for IBM-compatibles completely ignore the second button that clutters most IBM-compatibles mice (mouses?). The third button, part of the Englebart and PARC mice, has disappeared almost completely from popular interfaces. It was my own difficulties with the three-button mouse and watching other people having trouble learning it that led me to rethink the design. With one button, I reasoned, you could not get confused about which to use. It took a while for me to figure out just how youd do all the necessary operations with just one button, but I was able to find methods that in every case required the same or fewer operations than those required by the PARC system; it was faster, easier to learn and use, and it was far less modal. Of the methods I invented, the most fundamental was the idea of pressing and holding a button while dragging and using the release of the button to indicate that the operation was complete. This differed from the method used at PARC (which dated back to Sutherland) of click, drag, and click again.

When Larry Tesler joined Apple from PARC he was naturally resistant to the one-button mouse, being familiar with the three-button version and having long touted its real advantages over non-mouse systems. It took considerable effort to convince him, point by point, that my solution was not only workable (which he and others doubted at first) but in fact superior. In any case, the interface we were developing was a distinct and new creation, though sharing many elements with and owing a very real debt to what had been done at PARC. A major part of that debt, of course, is that I was able to use PARCs work as a living demonstration of the effectiveness of the flavor of interface I was bringing to Apple.

The one-button mouse was not the only major difference between the Mac and the systems at PARC. Another interface improvement that made the Mac feel so much easier to use were the ways selections were made and menus were engaged. At PARC, menus were relatively static lists of limited length that you could summon and dismiss. Bill Atkinson proposed instead that just the title of a menu be shown, and that when you point to it, click and hold down the mouse button, the menu appear. Then you would release the mouse button when the cursor was pointing at the desired item. This made menus appear when you needed them and disappear without significant extra effort. Furthermore, as we both pointed out, having the menus at an edge of the screen and having the cursor position confined at the edge meant that you had to point accurately in only one dimension, which made the menus easier to use. The design of Microsofts Windows and similar interfaces missed this useful pin to the edge idea.

Atkinson was led to pulldown menus that you can drag across to your desired item by analogy with my point and drag methods. Probably because it worked much as a typewriter SHIFT keys does and as a pencil does (you put it down at the beginning of a line and pull it up at the end) my way of using a mouse has prevailed. I extended this idea to drawing lines and to creating rectangles and other shapes by pointing and sweeping across the diagonal. My hold and sweep concept was then applied to making graphical selections. We created a rectangle that surrounds or touches the items to be selected while the button was held. The methods Bill and I devised are now so universal that even some people who worked on the earlier systems assume they worked as the Mac does now. I suggested that Apple patent the one button mouse and the new way of using such a pointing device, but Jobs nixed the idea in favor of patenting Atkinsons pull-down menus. Apple missed this opportunity simply because Jobs didnt want my name to appear on any Apple patents (though I have about a dozen of my own).

A more subtle difference between the Mac and the work at PARC is that in the Mac you pointed to something to and then you told the system what to do with it. It is called the noun-verb paradigm and is now nearly universally recognized as desirable by interface designers. As Bill Buxton of PARC reminded me, on the Xerox products they used a more complex noun-verb-noun method and there were a bunch of function keys (like the current IBM compatibles). To quote Buxton, Both the concept and the operation were quite different... it is remarkable how few people who teach and talk about GUIs dont even seem to understand the differences to even this degree of subtlety.

Another fundamental part of the Mac from the very beginning was the idea that there was to be unifying software built in. Knowing about real-world time constraints and the inherent laziness of all humans, I suspected that if we built in an interface, programmers writing applications would use it, grudgingly, for their first mock-up as it was much faster and easier than writing the interface themselvesstandard practice in all products prior to the Mac. I knew that writing a rule book would only antagonize the independent spirit of software developers who are inherently entrepreneurial. They had to be tricked into using the Mac interface at first. I could depend on their time constraints and the fact that our interface was likely to be far superior to what they planned to insure that the details used in the software prototype would appear in the final product. This underhanded trick in fact worked. When the Macintosh was released, users found that learning new applications on a Mac required far less of an effort than on any competing system. This gave third-party software developers added incentive to do things in the Macintosh Manner, and Mac users have reaped the benefits. The success of the Mac led other companies to copy its interface, and one can now move without too much difficulty from the Mac to Windows, to Geoworks, to most workstations, and even to some mainframe front ends without retraining and with barely a glance at a manual or help screens.

My unifying software originally was to be a graphics-and-text editor within which applications could run as additional commands (via menus), all input and output being through the interface designed for the editor. Later, the PARC desktop metaphor was adopted from the Lisa group (and that from the Xerox Alto and Star computers). Due to the incredible work of the Mac software team, the necessary code was designed and squeezed into a Toolbox that fit into a relatively small ROM (Read Only Memory) that we could afford to put into the product.

The interface concepts I wanted to implemenet required fundamental hardware changes. One example was the way the electronics of keyboards were designed; I am not talking about keyboard layout which obviously affects the interface, but the way the keyboard works at the chip level. Before the MAC, each key on commercial keyboards (PARC was at that time a non-commercial exception) put out a signal when pressed. By the middle 1970s a special encoder chip took a signal from a key and produced the code for the symbol that key represented. There were usually a few exceptions: the SHIFT key could be pressed and held and while other actions took place, the same was often true of other kinds of shifts such as control. But these few exceptions were built into the encoder chip; what I wanted was a keyboard where the computer knew whether any keys were up or down. By analogy with piano and organs which can use any combination of keys simultaneously (playing what musicians call a chord), this was called a chord keyboard. I had long believed that this was an essential step toward improved interfaces and when I first went to PARC I was delighted to learn that they had come to the same conclusion. Burrell Smith, our hardware designer, participated avidly in the discussions about the interface, and often suggested ways in which hardware changes could help the interface and sometimes how making a change in the software design could simplify hardware requirements. In each case the interface requirements took precedence, but it was probably the first time a commercially successful computer was designed with hardware and software subservient to the issue of usability. The Mac succeeded because the initial impetus for its creation came from a humanitarian impulse, rather than a hardware dream or a marketing study.

SELLING JOBS ON THE IDEAS

A popular description of Jobs that arose around this time was that he was a reality distortion field. This phrase accurately described Jobss ability to convince people that whatever he was saying at the time was inevitable. Id seem him charm otherwise reasonable people into believing absolute nonsense. Some of this is helpful when doing something new in the world, but Jobs lived at the center of this field and actually believed and acted not only on vision, but on the basis of his own falsehoods, with sometimes unpleasant consequences.

It seems to me that Steve Jobs was also mesmerized by the power of part of his key insight that helped make the Apple II a success: make it look attractive became a guiding principle. He continued to confuse appearances and quality ever after; decades later at Next he hired decorators for the new office complex as his first major expenditure. This passion for appearances would have been neutral or positive in someone who also understood the products but Jobs often did not. In 1979 he botched a poster that summarized how the Pascal programming language was written. Programmers found similar sets of diagrams created by the originator of the language, Niklaus Wirth, a handy reference. This example is not significant in the history of Apple per se, but is diagnostic as to how Jobss went from being a major asset to a significant company liability.

In writing the Pascal manual I had discovered a number of errors in Wirths diagrams and also found some simplifications. The manual had a set of diagrams that reflected these corrections and improvements. I thought that it would be both good advertising and of real benefit to programmers to put the entire set into a decorative poster. A key idea was that color could be used to link items of the same syntactic type, making relationships between the language elements clearer. Jobs thought it was a great idea, and promptly hired a prominent graphic artist, Kamifuji, to produce the poster. Jobs asked me for a copy of my diagrams so that the artist could estimate the project, telling me that once we had a quote I would work with the artist to do the poster. Next thing I knew, Jobs proudly came into my office with the finished work. Thousands had been printed. It was very good looking, with bold colors on a jet-black background. The problem was that some of the diagrams were no longer correct and the colors had been chosen purely for esthetic effect, making the chart unnecessarily hard to use. I told Jobs that it was very pretty but was wrong, but he didnt care, and blissfully went on to something else. The posters were sent to stores as advertising posters, but they couldnt be shipped with the Pascal product as planned. It was a waste of time and money, and the incident serves not only as an example of Jobss valuing surface over substance, but also the fatal flaws in Jobss modus operandi of working exclusively with the stars himself and his self-delusion that he always knew best.

By the end of 1979 it was clear to many people that unless Jobs had a better understanding of what was being attempted on both Lisa and the Mac, he would continue to inadvertently sabotage the former and be antagonistic to the latter. My friend Bill Atkinson knew a great deal about what was going on with the Macintosh even though Jobs had officially forbidden him to work with people on the project. This meant that Bill had to keep his involvement with the Mac secret, lest he lose his in status, while he worked on the Lisa project. At the time he was writing his meticulously crafted QuickDraw graphics systemthen called LisaGraphfor Lisa (Apple knowingly used the name of the list-structured graphics system I designed for this central piece of softwarewithout permission or compensation).

With Bills connivance and the help of Tom Whitney, who had given me the title of Manager of Advanced Systems to correspond with my work on the Mac, and by keeping my name out of the picture, we managed to finally convince Jobs to visit to PARC. The visit was apparently delayed while a financial deal with Xerox was put together, but I was not privy to those negotiations. Jobs later said that after he went to PARC, he returned inspired, and launched the Lisa and then the Macintosh. This story, once promulgated by Apples P/R department and often repeated in books, articles, and even by the generally excellent PBS series on the history of the computer, is (to say the least) anachronistic. As Atkinson put it in a phone call to me You were instrumental in getting Jobs to go to PARC, and that was central to getting his support for new interfaces. Jobs pointedly did not invite me on this visit, and excluded me from the conversations about it when he got back; it would have been very hard for him to have admitted that I had been right about the value of the work done at Xerox.

For the most part, I remained oblivious to the politics going on at Apple, and concentrated on the design of the Macintosh. This left almost no room in my life for anything else except practicing the piano and occasionally getting out to fly a model plane. I bought a house a few blocks from Apple so I could bicycle in and back on a moments notice. The Macintosh project was my life.

What was the Mac concept like in the early days? We researched many possibilities. For example, we considered a bit-mapped LCD display which had a resolution of 256 X 26 (yes, twenty six) had a cost to us of about $240. At our usual five to

one parts-cost-to-list-price ratio, that part alone would have been $1200 at retail. A 256 X 256 or larger display with any technology other than the cathode ray tube (CRT) was totally out of the question at that time since a CRT display cost between $35 and $50. A drawing, done by Brian Howard in 1980, shows a one-piece box with built-in CRT screen, 5 1/4 drive, keyboard, and joystick. The joystick is in the position the trackball appeared in the later Mac Portable. We also worked on a strain-gage stick almost identical to the current IBM graphic input device. Embedded pointing devices have a long history at Apple, or example in 1978 Woz had come up with the idea building a pair of orthogonal thumb wheels (one each for vertical and horizontal motion) under the Apple II keyboard in response to my request that we build a pointing device into the box, one that could be operated without removing the hands from the keys. This seemingly obvious good idea (which I preached often) reached fruition years later with the PowerBook series. The idea was probably reinvented independently by the PowerBook group.

Graphic input was an essential element of the Macintosh from the first. I thought that the mouse in particular was a clumsy way of going about itfor one thing, it takes up too much desk space, for another you have to find it anew each time you want to use it. But Jobs was an adamant mouseist (mainly, I think, because thats what PARC had) and until third party vendors supplied trackballs, the mouse was the only graphic input device available for the Mac.

TEAM BUILDING AND THE TOOLKIT

One of my basic concepts was that a software nucleus would be built into ROM, a fixed part of the computer that would form a home port to the user tossed about on the high and varied seas of application software. To write the software, I hired Bud Tribble, who had similar thoughts. He and the two other Bs, Brian Howard and Burrell Smith, were the first Macintosh team. The Mac Toolkit was initially written by Tribble (who was in charge of Mac software), and it was taken over by unstoppably hard-working UC computer science dropout Andy Hertzfeld, Bill Atkinson, Bruce Horn (who had been one of the kids who had tested SmallTalk at PARC, he had been 14 then), and others.

Each member of the original gang of four came to the group through a different route. Bud Tribble was a medical student, a programmer and designer of genius who I had known at UCSD; he and Bill Atkinson had been good friends there. Atkinson pointed out the talents of a man working in repair, Burrell Smith, and after interviewing Smith, I hired him as head of hardware design. And of course Brian Howard had been a friend of mine for years. There were established hardware designers that I had tried to bring over to the Mac (and who wanted to work with me), but Jobs had forbidden them to join the project, hoping perhaps that without them I would be stymied. Smith proved a first-rate designer who was open to thinking from a software and human-interface point of view, and he was a delight to work with.

I brought MIT Anthropology student Joanna Hoffman on as our marketing person. Her major contribution to the Mac was to make sure any design decisions didnt preclude international sales; this concern was also unusual in the then-parochial microcomputer industry. Thus the Mac had, from the first, the accents, special characters, and diacritical marks needed in languages other than English. We had come a long way from the philosphy that upper-case letters were all you needed. She also introduced me to my future wife. Steve Clark (another UCSD student I brought to Apple, and whose Olympic-level kayaking sister Candi was later to marry Woz, as Ive said, its a small valley), and a few others I had formed the nucleus of a team that was easily the equal of the much larger, better funded Lisa group. Some, such as programmers (and musicians again) Gareth Loy and Bill Schottstaedt I hired from SAIL, another, the remarkable poet Bana Witt, had been a music student of mine when I taught at the San Francisco Community Music Center. She later married Bruce Tognazzini, another Apple employee who worked with me and was to write and lecture extensively about interface design. Being a minister, I had the pleasure of conducting their nuptials. Donald Reed, the very image of a bookish intellectual in appearance and manner, worked with me closely on documentation. And, of course, there was the under-the-table help of Atkinson and others on the Lisa team who believed in what I was trying to do and the warm support of the late Tom Whitney, who had been hired to head engineering for all of Apple.

THE END OF THE BEGINNING

John Couch, a good manager and insightful computer scientist who was running the Lisa project, increasingly found Jobs a nuisance, and eventually managed to get him removed from the Lisa project. Jobs, at loose ends, and hearing rave reports about the Macintosh, decided to have a hand in it. Apples top management helped shunt him to the Mac project to get him away from Lisa, which was seen as the companys hope for profitability in the 1980s. Jobs attempts to undermine the Mac project now gravitated to destroying my credibility. One of the more blatant incidents was the brown bag lunch at which I was to describe the Macintosh project to the company at large. It is discussed in a confidential memo that I wrote to Mike Markkula to explain why, though I was seeking someone to manage the Macintosh project so I could concentrate on technical issues, I didnt want Jobs to be in charge. The memo specified in detail Jobss many and egregious failings as a manager.

I had asked that the memo be kept secret, and Markkula agreed, though he said that he didnt think he could do anything to control Jobs. I believed this assurance and felt betrayed when Jobs called me in to his office a few days later to discuss the memo. I dimly recall Markkula saying something about having had to discuss it with Jobs. But it was a very open company, doors were left unlocked, and people wandered freely into one anothers offices. Any of a number of people might have seen the memo and made a copy for Jobs, or he may have noticed it himself. Markkula thinks that something of this nature is what must have happened, and it might well be the case.

The memo reflected the running joke that the way to get Jobs to agree to something was to tell him about it, let him reject it, and then wait a week until he came running to tell you about his new idea at which point youd exclaim, great idea Steve, well do it right away! In the memo I also made what was to prove a perfect prophesy: Jobs was wrong on his Apple III schedule, wrong on the LISA schedule, wrong on the cost and price estimates, and he will be wrong on Macintosh. He is a prime example of a manager who takes the credit for his optimistic schedules and then blames the workers when deadlines are not met.

The memo also related the incredible brown bag incident: Jobs is often irresponsible and inconsiderate. An example is the brown bag seminar I was scheduled to give on 17 February. In January, he first cancelled the seminar, but then he agreed that I was to give it. Two hours before the talk he called me to say that he was cancelling it again. His reason was: I cancelled it because of the reorganization in PCS. However, Jobs did not tell the seminar's organizer about the cancellation, nor did he place any notices announcing the cancellation.

At noon, fortunately, I made a last-minute decision to go over to the seminar site, where I discovered a crowd of over 100 employees waiting to hear the seminar. I announced the cancellation myself--and then I gave a talk on my current work and interests at Apple, instead. I was careful not to mention Macintosh or give specifics since Jobs had forbidden it, but just explained the cognitive aspects of the interface and design principles my group and I had developed; it was as everybody knew a veiled introduction to the Macintosh project. The talk was received very enthusiastically.

The next morning after the brown bag seminar Jobs called me into his office and told me that I had violated his explicit instructions and was fired. Ignoring what he said, since he often spoke without thinking things through, I told him that Id come back in the afternoon, after Id completed something I was working on, and we'd discuss the matter. Later I went back and, as I expected, he had decided not to fire me but I was given an extended paid leave from Apple. The leave turned out to be a very important time in my life. For one thing, I went to a party at marketer Joanna Hoffmans house where I met my future wife, Linda Blum. I found a place to live on ten acres of land high in the foothills of the Santa Cruz mountain, having a magnificent view of Silicon Valley. I rebuilt the dilapidated old house that was there, adding a large music room for my piano and a small flying field for my model planes.

When I came back from my leave I was offered the position as head of Apples research division. I had been offered this before, had accepted, hired a good group, and had them whisked away to put out fires. In those days, Apple didnt know what research meant and looked at the talented people I hired as resources wasted if they werent working on current products. Besides, there was the matter of personal integrity. Steve Jobs had become impossible for me to work with, and had begun to exhibit the irrational traits that would waste hundreds of millions of dollars at Next. Most people worked around him or sucked up to him or were in awe of him. I could not share the awe of him many people expressed, some because of the legends, some because of his wealth and power. In fact, he was no genius but like a planet shone by reflecting the light of

the brilliance of others; yet he thought of himself as the Sun King. He could not abide someone who was unimpressed by Steve Jobs, he had by his actions lost my respect, and I am incapable of being a sycophant.

Steve had chutzpah in the extreme; he said that the Mac would make a dent in the universe without the least idea how big the universe is or how little a dent all our activities really make. But there is an aspect of the man that can best be explained with another Yiddish word, mensch. It is high praise to say of a person that he (or, in these enlightened days, she) is a mensch. The terms superlative is a real mensch. A mensch is cultivated without losing the common touch, upholds high principles while remaining practical, is kind and generous without short-changing himself, and is attentive to his responsibilities to himself, his family, his business, his associates, his community, and the world. If you understand the qualities that make a man a mensch, then you understand a lot about Steve Jobs. Everything a mensch is, he isn't.

At this point the only alternatives left to me were to leave or learn to toady to Jobs. I resigned from Apple and gradually watched my predictions about the Mac come true. Jobs took until 1984 to get the project out (Burrell Smith quipped that it was in constant time to completion mode and I was repeatedly told that Jobs did exactly what he said I would do, make endless mindless changes; I have many faults, but lack of direction was never one of them.) Steve was given to putting absurd requirements on the projects designers. This was especially ironic since one of the arguments he used to convince management that he should be given the Mac project included the claim that I was an academic dreamer and a perfectionist who would keep on changing his mind and never bring the project to fruition on time. My detailed schedule showed release of the product toward the end of 1982. Jobss verbal plan was something like six to eight months shorter. There is some evidence to back up my claim: the next project of comparable scope that I managed (a workstation for Canon) was completed on budget and on schedule, the next project that Jobs tried to manage (the NeXT computer) was a disaster in both these regards.

The resultant Mac not only took more time to come to market, but was a far less coherent andin some waysa less capable product than what I had been working towards. The interface was was less consistent and harder to use and there was no way to get at the hardware bus. Other parts of the Mac design were improved. Whether my version (as it would have matured as it approached production) would have been more commercially successful than the 128K Mac is an unanswerable question. The experiment cannot be done, and we will never know. I feel it would have been a somewhat better product that would have penetrated the market faster. I would guess that Jobs would disagree.

DREAM FULFILLED... ALMOST

Three decades ago I dreamed of a computer with which I could compose music and print it out in full musical notation, write properly formatted text in a panoply of fonts, have the ability to mix text and graphics, and do drawings with precision and ease. Today I do all this and more at the tiny and capable Macintosh PowerBook that sits at my desk. It goes wherever I do. This much of the dream has come true.

My reasons for deciding to abandon teaching for commerce turned out to be correct. It was the Macintoshs profitability (as contrasted to Xeroxs extensive published record) that convinced companies such as Microsoft and IBM that the interface was the controlling element in most sales. Now a vast majority of the computers sold has an interface that looks much like the Macs.

Because of the Macintosh project, computing has been made easier and more pleasant for hundreds of millions of people years ahead of when it otherwise might have happened. Though I made not a penny for my work on the Mac (besides my salary at the time), the fact that I helped change the world in accord with my own personal vision and that I have seen the effect of this in my own lifetime, would be fully satisfying if it werent for the fact that I know we can do much better.

The desktop metaphor, used by everybody from PARC though Apple and Microsoft and extended almost absurdly by Apple spin-off General Magic, was a clever way of making the workings of an operating system palatable and learnable. A far better solution is to eliminate the need for an operating system altogether. The current paradigm of using application programs is inherently wrong from an interface design point of view. This is widely recognized, but the solution offered is to make them interoperable, which solves some of the problems but by no means all. GUIs as presently designed and used are an interface dead end. Though they can be patched endlessly, a large jump in usability can only come from a completely different approach. The Cat computer, which I developed for Canon, demonstrated that my alternate approach is implementable and both more productive and more pleasant than GUIs. Canon, possibly because the moribund Electronic Typewriter Division had been given the task, failed to market the product effectively, and it is now a dead Cat. The parts of computer interface design that I am working on are now not dependent on particular technologies (until we get direct connections between machine and mind): any advances made in the basics of interface design will apply however more powerful computers become, however broad the information networks of the future spread, and however the technology is melded into our everyday or even everymoment lives.

--- END ---

SIDEBAR I: HOLES IN THE HISTORIES

Many friends have suggested that I counter the numerous incorrect accounts of the history of the Macintosh with a true one of my own. It is difficult, for even though nobody was or could have been more closely involved with the initial creation of the Macintosh than I, I cannot eliminate the colors that tinge my memories. It was a very emotional time, full of strong feelings, massive egos in conflict, distinctive personalities, and many rights and wrongs. But I cannot do worse than some of what has been published. There is a strange avoidance of scholarly seeking after truth.

An egregious example of the anti-academic attitude occurs in a book by Robert Cringely, who writes a delightful column that appears weekly in InfoWorld. In his book he has the Mac and Lisa (an Apple computer that didnt make it commercially) projects being created by Steve Jobs after Jobs made a visit to PARC in 1980 and came back all inspired.

I wrote to Cringely and pointed out that his booklike those of a number of other authors was wrong; Jobs had indeed made the visit in 1980 (some say in December of 1979) but the Mac project was proposed in the spring and officially started in September, 1979. In other words, the project was well under way before the event that was supposed to have inspired it took place. Cringely was unabashed. He wrote back: As for all the business of what project started when, whether Lisa started before or after Steve visited PARC, whether the Mac had already begun or not, well I dont think that it really matters very much. My attempt was to EXPLAIN (I say that at the front of the book), not to be a historian.

How one can hope to explain what happened if you dont even know what happened eludes me.

A PBS special on the history of computers made the same mistake of attributing the genesis of the Mac to Jobs visit to PARC. When I wrote to Jon Palfreman, its producer at WGBH, he replied, The part of the program you are referring to comes at the end of a lengthy segment about the highly innovative work done at Xerox PARC. This section was based on extensive interviews with Alan Kay, Bob Taylor and Larry Tesler. The purpose was to show that the key concepts of interface design which today are a feature of most PC's (if you count Windows) were first discussed at Xerox PARC. When those ideas were embodied in an affordable machine--the Macintosh--they began to change the world of personal computing. I was aware of your key role in the Macintosh project, and indeed of the contribution of people who developed Lisa. My aim in this particular program wasn't to detail the history of Apple but to show how the key interface ideas found their way into consumer PCs.

Again the false scenario seems so plausible and story-like that the person in charge does not care to detail the history. But it is in that history, and not only the history of PARC, that the interface ideas found their way into consumer PCs. The people he interviewed were at PARC, their association with Apple began only after the Mac was well under way. Thus they could only tell him about the development of the ideas at PARC and, in the case of Larry Tesler, about the work on Lisa only after some time in 1980 that is after Apple was committed to the basic direction I wanted the company to take. Larry was quite resistant to some of the non-PARC ideas that we had developed independently. He did not understand, at first, many of the improvements over Xeroxs work that I created (for example, he fought against the one-button mouse). He did not work on the early Macintosh project at all but on Lisa, which was modeled closely on the Xerox Star, even to the point of having the same Xerox-created names for the fonts. The Macintosh proceeded for years much more independently and, significantly for the reports that used them as sources, out of the view of the people interviewed!

The years of thinking and experimentation on the early Macintosh project have gone unreported, though they led to many of the breakthroughs that made the Macintosh and the systems that have been built on that work so much of an improvement over what went before. Against this reality we have the powerful mythological image of Jobs going to PARC, having an aha! experience and coming back at full cry to Apple to create a fantastic project. The fabricated Jobs story is familiarit parallels that of Archimedes jumping naked out of his bath crying Eureka! and a dozen other stories. That there was a little-known computer scientist who had been working on the concept for over a decade, who created the project and then maneuvered Jobs to go to PARC so that Jobs would begin to understand (and thus support) what was already going on at Apple, is a very different, more complex, and unlikely-sounding story. Also, there was the appealing (and basically true) legend of the two college drop-outs who created the profitable and excellent Apple II. It is an easy fiction to go on to have one of the dropouts create the even more revolutionary Macintosh. It is less striking a tale that a former college professor and computer center director with a degree in computer science instigates such a thing. Its not as good a story perhaps, just the way it happened.

Cringely and Palfreman were not being underhanded, only a bit careless and in Cringelys case cavalier. In some other cases, an author drew the wrong conclusion from not having accurate information. In Jeffrey Youngs book Steve Jobs he writes of the first time that Steve Jobs (along with Atkinson and others) saw the work at PARC. Atkinson and the others were asking Tesler questions, one after the other, What impressed me was that their questions were better than any I had heard in the seven years I had been at Xerox... Their questions showed that they understood the implications and the subtleties... But Young did not ask why they had such a high level of instantaneous understanding. The reason is that I had been explaining all this stuff to Atkinson and Jobs for years; Atkinson (who had been a student of mine and worked with me for a long time prior to this meeting) had grasped it very well. Tesler didnt know about this background, wasnt told, and so was bowled over. Atkinson couldnt very well say that Raskin had briefed him and some others because (i) I was out of favor with Jobs at the time and anything I proposed of was automatically rejected, Jobs was like that and (ii) it had taken much planning and sleight of hand to get Jobs to go. It helped for it to appear to be opposed by me and desired by Atkinson.

There is also the halo effect. During the years that Steve Jobs was on top at Apple and before Next showed his fundamental weakness, he was usually credited with inventing the Macintosh. Later, when his star was declining as his company, Next beat one strategic retreat after another, and with General Magic, cofounded by Bill Atkinson, Andy Hertzfeld, and Marc Porat about to announce its first product I found, in the Dec 27 1993 / Jan 3 1994 issue of Infoworld a story hailing Bill Atkinson and Andy Hertzfeld as the creators of the original Macintosh. Their contributions were essential to the product and represent some brilliant work, but neither of them has ever claimed that they created the Macintosh. Again we find the heroes of today falsely credited with the achievements of others not currently in the limelight. The Steven Levy book on the history of the Macintosh, Insanely Great, published to ride the wave of 10th anniversary announcements for the Mac, is occasionally equally at odds with historical fact . Aside from retelling the Jobs-at-PARC story, I am credited with having paintings shown at a famous museum; in fact I have never done any paintings. Strangely, Levys adaptation of his book, as published in the February, 1994 issue of Popular Science tells a story that is far more accurate (though I am still a painter). Sculley, in his ghost-written book Odyssey, lists me as a programmer at Apple. I was never a programmer at Apple, and the rest of what he says is nearly as inaccurate. He got his misinformation about the history of the Mac primarily from Jobs, with whom he spent a lot of time. Like Cringely, Levy, and Palfreman he never chose to interview me or even call meor others who were thereto check on his facts. My experience with Jeffrey Young was especially disturbing: I had agreed to the interview on the basis that I would get to see and comment on the galleys before publication; he never sent themhis inaccuracies compounded by a breach of trust.

But Jobs own view of how things came about was necessarily distorted. For example, we very often deliberately misattributed ideas when speaking to Jobs. If the group liked an idea by someone Jobs didnt like at the moment we said that it was due to someone currently on Jobss good list. It was also often necessary to use reverse psychology on Jobs, the way you might with a recalcitrant 4-year-old. We got a lot of features into the Mac by having someone (usually me) spout the opposite, then Jobs would see the problem in that approach and often tell us to turn it around. Another technique was to tell him about something informally. Often he replied that the idea was dreadful. Then when he proposed that same idea a few days or weeks later, after its merits had settled in, wed tell him he was a genius for having thought of it.

Even if Jobs reported his recollections of the history of the Mac accurately, they would be far from what actually went on. Being very independent, I was often on Jobss bad person list, so I had to rely heavily on these techniques. Youd think hed catch on when an idea he thought was his turned up implemented later that afternoon or the next day, but he simply believed that it was his great engineers and his way of driving them that got it done so quickly. The fact that he had little intellectual basis on which to judge the difficulty of a software or hardware task often helped us pull the wool over his eyes. How it all looked to him I cannot say. Eventually his impossible management style became so well known that Sculley and the board of directors of Apple had to remove him from all functional duties in the company. My memo had finally been acted upon.

-- END OF SIDEBAR I --

SIDEBAR II: JOBS AND THE PIPE ORGAN

When I first started working at Apple, Jobs and I would take long walks (probably like the much-reported walks he would later take with John Sculley). I remember giving him mini-lectures on the philosophy of science or the performance practices of early music. On one of these walks I shared with him my life-long ambition to own a pipe organ. I explained that the valves to the pipes in many organs were driven electrically and I hoped to hook up an Apple II to one which would turn it into a modern player organ. I published an article with details in Byte. He asked me why I didnt have one; I told him that it was mainly a matter of space and that there was a secondary consideration of cost.

Jobs had a suggestion: if I could find an organ I could afford, I should buy it and Apple would let me put it up in the lobby of the new, large building on Bandley Drive, in Cupertino, California. I would hook it up to an Apple II, which would play it for visitors. After hours, I could practice on it, or even give company concerts.

Jobs was excited about the idea and told many people about the organ that was going to be installed. With this encouragement, I searched for an organ in earnest. In a few months I got lucky and found an abbey where the organ was being replaced. I told Jobs the good news and he congratulated me on the find. I purchased their old organ, had it crated and moved onto the abbey lawn (no small task in itself) and called Jobs to tell him that the crates with the organ would be there in a day or two.

What organ? he asked. The pipe organ were putting up in the lobby, I replied, thinking that he must be thinking about something else to have forgotten. He first said that he had changed his mind because it looked like space would soon be tight, and when I suggested that it was a bit late to change his mind, since I had already purchased the organ, he said that he had never agreed to have the organ installed at Apple in the first place.

When I got back I reminded him that he had made a commitment and I had gone to some trouble and expense based on his assurances. He told me that he had never assured me that he would give me room for the organ, and refused to speak in my presence to the people who had heard his promises. I asked if, until the issue was resolved, I could store the crates in the still-empty buildings. The crates were out-of-doors, this was an imposition on the abbey, and if it rained, the organ could be seriously damaged. He simply said no.

I was stuck. The organ was too large to fit into my house or even a rental storage unit. I called every organ builder, organ teacher, and church I could find and after much desperate work found a church in Santa Clara that needed an organ. They, in turn, found a benefactor to purchase the organ from me for them and, after long negotiations, I succeeded in selling it for a fraction of its value.

This was my introduction to the new Steve Jobs, or perhaps a phase of the old Jobs I hadnt yet seen. His friend of many years and Apples first employee, Dan Kottke, who had travelled with him in India and worked with him day and night to help Apple get started was treated even more shabbily. In late 1980 Dan found out to his surprise that in spite of their long friendship and the many uncompensated hours he had put in, he was not going to get any stock options. Later Woz gave some stock to Kottke (and some other deserving people from the early days of Apple such as Bill Fernandez, Chris Espinosa, Randy Wigginton, Cliff Huston, and Dick Hustonall of whom Jobs had turned his back on), an admirable act of pure generosity. As for me, I still dont have a pipe organ.

--- END OF SIDEBAR II ---

SIDEBAR III: COMPUTERS BY THE MILLIONS

In late 1979 I started writing about the implications of manufacturing and using millions of computers. At that point there were in the order of 100,000 computers of all types in the world. I wrote this to alert the company to the areas we would have to think about in the future. As far as I know, it has the first use of superhighway just now (1994) becoming fashionable with regard to enhanced computer communications channels. I am generally pleased with the match between my futurism and the actual future. It was circulated at Apple during the first few months of 1980 and published in 1982 ("Computers by the Millions," SIGPC Newsletter, Vol. 5, No. 2.) Years later, in 1987, people at Apple Computer Inc. were kind enough to give me the millionth Macintosh produced; it sits proudly in my shop and is still in use.

COMPUTERS BY THE MILLIONS

by Jef Raskin © 1979

INTRODUCTION

If "personal computers" are to be truly personal, it will have to be as likely as not that a family, picked at random, will own one. To supply even our own nation with enough computers to make this happen (over, say a four-year period) we will require, for example, twenty-five companies each producing over a million computers a year. To keep this essay short, no attempt is made to give a comprehensive treatment, but to just give the reader an idea of the magnitude and character of a few of the problems.

FINANCIAL CONSIDERATIONS

Let's say that you were producing a million computers per year. This is about a hundred thousand (more or less) per month. Let's also say that it cost you $1500 to build each one, of which $1000 is parts and $500 labor (a moderately expensive business system). These amounts are not accurate as to either parts/labor ratio or to absolute costs, but they are not so far off as to substantially prejudice the argument.

To be prudent, you have enough parts on hand for thirty days production. The section below on the availability of parts makes up only part of the argument as to why having some parts on hand is a good idea. (It is also a good idea to have some of the finished product in stock as well.) Given our assumptions we have to lay out one hundred million dollars a month on parts. Having a hundred million dollars of inventory laying around for a month is an expensive business. At 12% simple interest, you could make a million dollars a month just by putting that money in the bank, and not bothering with manufacturing at all.

You also have fifty million dollars a month in labor costs to think of.

On the other hand, if you are selling these computers for about $6000 each, you have more coming in than is going out. It is by no means all profit; for example, most of it probably goes for advertising.

The numbers given here are by no means representative of the industry, and many items (such as distribution, engineering, software and much else) have been ignored completely. But we are certainly in the right orders of magnitude. It should be clear by now that you have to be rich to produce a million computers a year.

The largest personal computer company in the world right now isn't even close to these figures.

Yet.

SOFTWARE PROBLEMS

Software, at first, doesn't seem to be nearly as much of a burden as the hardware. Certainly it does not seem to entail the constant financial drain as does hardware manufacturing. You might think that software might be written once, and then may be supplied identically for each computer. This is, of course, not quite true, as will be seen in the next section. But, for the moment, assume that software can be written once, and then merely duplicated for each customer.

Part of the catch is in the phrase "merely duplicated". Just how mere a duplication is depends on what is being duplicated. If the software is supplied on tapes or disks (or any magnetic medium) we find ourselves in a morass of time-consuming steps.

Magnetic media must be duplicated serially, and written (at best at accelerated speeds) individually. Take a diskette as an example: it might take one minute to load, copy, verify, and unload a diskette. In one month, to produce 100,000 diskettes, you will have to spend 100,000 minutes in duplication.

A working month, given two 8-hour shifts, has but 19,200 minutes, say 20,000 in round numbers. You will need five duplicators running at full speed, with no down time or other problems. To be realistic, you'd better have custom-made mass disk duplicating systems. Now have a catalog of 100 programs, and you will have to have a factory with nearly 1,000 duplicating stations, say 100 machines each of which can duplicate 10 diskettes at a time. Now you need 100 operators, their management, janitors, and so on... It is likely that companies will come into existence to fulfill the function of mass storage media duplication in this kind of scale.

Now, as an exercise in large numbers, the reader should go back and calculate the square footage required to build and warehouse the 100,000 computers per month.

Other problems arise with programs supplied on ROM, where the lead time becomes a headache (to say nothing of having someone produce all those chips, or getting into the semiconductor business yourself.) ROM is also impractical for large programs. On the opposite tack, video disks are much faster to duplicate on a per bit basis than almost any other medium, but are impractical for small and medium size programs since you have to create the whole disk at once (given most current technologies).

One technology for distributing programs that turns out to be very practical for extremely large runs (even intermixed with small runs) is the printing press. Bar codes, magnetic printing and other formats that can be run off on a printing press and read by a computer offer great opportunities in a large-scale operation. Fast readers for such codes are not presently in development.

Since documentation, point of sale information and advertising material must be distributed with the programs, the printing press is at an even greater advantage: it can do the whole job.

Software can also be supplied via a communication channel. This possibility requires a few paragraphs of its own. It is discussed below.

It is clear that software duplication will require much planning, and perhaps some new technologies.

NEW KINDS OF SOFTWARE

The full power of the computer is not available to an individual who owns one until he or she can program it. This opinion is rapidly becoming a heresy. The trend is to more and more packages that do specific tasks. This trend is not to be deplored, as software packages fulfill a useful role. Another trend is toward fill-in-the-form or pick-an-item-on-the-menu customizing of programs. This trend, too, is to be encouraged. Nonetheless, unless extended far beyond what is now being done (say, to the point where the menu consists of all possible program statements) it does not give the user the full power of a computer.

This is not the place to discuss techniques for easing the average user into programming (and it certainly will not be done with BASIC, Pascal or Fortran), but it can and must be done. If not, the computer will become a mere appliance--at best performing a small number of possibly related tasks. What is desired is for the computer to become an appliance, but not a mere appliance. Its presence must be taken for granted by its user, but in the long run, the act of programming itself must be taken for granted as well.

In the short run it will be, if successful, a mere appliance.

SOFTWARE UPDATES

Software is seldom perfect and usually has errors (called "bugs"). The manuals that describe computers and their software often have errors, of which some will be caught. In the past, it was the custom for a computer or software manufacturer to supply "updates" to their customers to correct such errors. When you have a million customers, this becomes impractical (if not positively ridiculous).

Personal computers, unlike their larger brethren, are not sold to fixed sites, but to mobile individuals who are free to sell (and give) these machines to others without informing the factory. Of a million people, a few hundred thousand move each year, and it is very expensive to find a few hundred thousand missing people.

It costs (at present mail rates, and allowing a pittance for the material to be sent, and unrealistically underestimating labor costs) well over $300,000 to merely send a letter to a million people. To keep track of which person has which version of the software (updates imply versions) would require a large computer in itself. It will be very easy for a programmer (or almost anybody else) to make an error that costs the company a million dollars, even without anybody generating a lawsuit. All the error must do is force the company to make an update to a piece of software that went out with each machine for the last few months.

It is clear that the concept of supplying updates to software or manuals will have to go. In its place should come much better tested software and manuals, and the concept that what you buy is what you get. Later versions will be sold as new, separate programs; compatibility (especially of data prepared under previous software offerings) will become a major concern. It can be expected that people will stick with an old, familiar system rather than buy a new one--unless it contains some new features that are in themselves worth the entire purchase price.

I am concerned that what might happen is that the same old quality of software will go out and then it will not be supported or updated due to the magnitude of the problems in reaching so large a customer base. The industry has the golden opportunity here to turn people off to computers in unprecedented numbers.

LOWERING THE COST OF SOFTWARE DESIGN

Actually, software costs will be in the same order of magnitude as hardware costs, both for the manufacturer and the user. As is well known, software management, in the face of large quantities, is a less well known art than the management of large-scale manufacturing. There are a few secrets that have been written up many times, but are rarely heeded: use a single higher level language for all program development, use clean and unified design, use a management technique like the chief programmer concept and so on. But for computers by the millions, some new ideas may come into play.

The mass sales of personal computers started out with the S-100 bus machines. The concept was that you could buy all kinds of devices and plug them in. What was not said was that you then had the rather terrible task of writing software to support these new "boards". Even the more sophisticated operating systems still required detailed understanding of the add-ons. Even the second generation personal computers (Apple II, TRS-80, Pet, etc.) allow plug-ins and add-ons. This creates a software nightmare.

The third generation personal computers will be self-contained, complete, and essentially un-expandable. As we shall see, this strategy not only makes it possible to write complete software, but makes the hardware much cheaper and produceable. The kinds of options that do not give programmers nightmares are things like case color, kind of screen (so long as size, aspect ratio and resolution are unaffected), power supply and the model name.

Programmers will be very happy when the computer for which they are writing software is a constant environment. This is not to say that good programming cannot handle the problems inherent in a varying environment, and on a machine which allows many different peripherals. What is being said is that it is more difficult to program for such machines, and that such programs take up much more space. We need every help we can get in simplifying the programming problem.

ANSWERING CUSTOMER QUESTIONS

Most responsible computer companies have people who can be called when a customer has a question. If 30% per year of a million customers call, then you have to handle 300,000 calls per year. Given that there are less than 300 working days per year, that means over 1000 calls per day. It is inhuman to have a person answer even 25 calls per day, so you have forty full-

time people just answering questions. This seems to be a small number, compared with the numbers we have been considering, but in practice these people need management and support, there are questions coming in the mail, and the calls don't arrive evenly spaced in time. This last means that you need extra people to handle peak loads. Or, you can alienate many customers.

In addition, the people answering the phones must be very highly trained, both technically as well as in telephone decorum (you get a lot of angry calls). It is hard to find such people, slow and expensive to train them, and they usually don't last long before having to rotate to some other job. I challenge you to try to find a cadre of, say, 100 such workers. I wouldn't look forward to such a recruiting effort. Yet, it will have to be done.

Or, you can train the dealers to handle the problem. Later, we will see how many dealers are necessary.

HARDWARE PROBLEMS

The present crop of personal computers are nowhere near having what it takes to sell in the quantities we are discussing. The simplest problem is in manufacturability. The computers we have now were not designed for true large-scale mass production. They are full of connectors, separate printed circuit boards, nuts and bolts, and require too much hand labor. One part, say a bolt, that takes seven seconds to attach, will cost the company (besides paying for a million bolts) a full time assembler's salary each year. With various overheads (such as space, bookkeeping and more) the elimination of one bolt can save the company $40,000 per year.

That bolt wasn't nearly so interesting when you are making only 10,000 or so computers a year, and even at current rates in the 100,000 per year category it may cost only $4000 per year. But at a million computers per year, each little piece can be very expensive. If re-tooling a portion of the case so that the bolt is unnecessary costs $30,000, the re-tooling can be amortized very quickly.

Another consideration is that hardware can no longer be the usual conglomeration of little boxes held together with cables. For user convenience, there should be nearly zero set-up time. Besides, manufacturing costs are far less for one box than for many. Cables are also expensive and failure prone.

We have seen, above, that software may well demand strong restrictions on the variability of hardware. This only reinforces the same decision made on the basis of cost reductions in hardware per se.

SERVICE

Assume that only 1% per year of the million computers fail in such a way that they cannot be fixed at a dealer. That's 10,000 computers per year or some 500 per working day. (Many present-day personal computer companies don't ship 500 computers per day, much less are prepared to handle that many coming in and going out of a service bay.) One technician, with elaborate diagnostic equipment, might be able to fix (on the average) a dozen computers a day. With one thing and another, you will require a minimum of 50 technicians. Auxiliary clerical and shipping personnel will also be required.

This also implies that there might be some good reasons to make the computer itself more reliable--and again to minimize model changes so that automatic diagnostic equipment can be used fully. When you allow a variety of attachments to be placed inside the computer (e.g. on the bus), especially when they are from other manufacturers, it is hard at times to pinpoint problems. Since the user will not typically send in foreign devices with the computer to be repaired, the problem may not be detectable at all at the service center.

Another suggestion that the problems of service bring to mind is that the computer should be highly modular (conceptually--e.g. all timing circuits in one place on the board), and designed with strong advice from the service organization.

Consider also the number of warranty cards that you might have coming in each day--that requires a department in itself. Of course, if each computer had a built in modem and could thus "talk" over the telephone lines, and if each computer were given an electronic serial number, then you could have the purchaser merely call a toll-free number to register their new computer. You would have to set up a considerable automatic facility to handle the some 1000 to 2000 calls per day that will come in.

SALES AND DEALERS

Assume that an salesperson can sell an average of 2 computers per day, considering both quantity and individual sales. Make it 400 computers per year (at, say, $3,000 each that means sales of $1,200,000 per salesperson). Even with such prodigious selling, and assuming 5 sales persons in the average store (so that each store sells 2,000 computers annually) we need merely 500 stores. Since that many stores at present manage to sell only about one tenth as much as we have estimated here, we probably need at least 5000 outlets. Only one manufacturer has that kind of structure at present. It doesn't manage 2 computers per store per day, much less per salesperson.

The truly personal computers will probably have to be sold from a set of chains and other stores with a maximum of about 10,000 total locations nationwide. A good percentage of sales will be through direct mail outlets.

Many "personal" computer manufacturers at this time have barely produced 10,000 computers. Only the largest companies have produced ten times this many. This is mentioned just to keep things in today's perspective: right now there are few brands of computers with which the distribution network can even be provided with one computer in each store of our projected number.

ADVERTISING

Advertising will continue on its present path, moving from the hobbyist and specialist magazines to the large-circulation general readership periodicals, to newspapers, radio and television. Eventually the personal computer companies will be sponsoring prime-time programs and special events on television, prominently backing major sports figures and other personalities, and generally making as much of a nuisance of themselves as do beer, automobile, camera, soap and tire companies.

MANUFACTURING

Since some idea of the volumes required has been discussed (and left to the imagination and pocket calculator of the reader), there is not too much to be said on this score, except that many of the techniques used by the large quantity manufacturers of television sets, washing machines and automobiles will have to be adopted, including considerable automation. Unionization may occur, and if automation is not strongly under way before this happens, it will become difficult to accomplish. Some manufacturing may have to be done on an international basis.

AVAILABILITY OF PARTS

At the present time, most electronic components are scarce, with long delivery times. Many components are on allocation from their manufacturers. Plastics and metals, due to current trends in oil pricing and metals speculation are rising rapidly in price, with some shortages already apparent. There is no assurance to be had that parts and materials for building millions of computers can be obtained.

In response to these pressures, manufacturers will use conservative design which allows second sourcing. When new technology is imperative, some vertical integration may be necessary--although this often just puts the problem on raw material supply (e.g. high-purity silicon) instead of finished goods (e.g. ICs).

If the computer we are discussing uses any chips in quantity (e.g. it might have 16 memory chips) then the company will have to buy 16,000,000 of those chips per year. Not many suppliers are prepared to manufacture, let alone sell to one customer, that kind of quantity. Incidentally, the computer company will have to receive 80,000 of those chips every working day.

COMMUNICATIONS NETWORK IMPACT

We can pretty safely assume that future personal computers will have communications facilities built in. Since the only bi-directional network currently available is the phone system, we can assume that this system will be the first used extensively for inter-personal-computer communications.

We can expect the telephone company to attempt to create tariffs in response to these usages since, statistically, computer calls are rather different than ordinary voice communications. Two fundamental measures, average length of call and time density of information are much greater for computer communication.

There are two kinds of usages that can be easily foreseen. One is the sending of blocks of previously prepared data. The other is real-time computer conversations.

If we assume that the average transfer of information will be about 30,000 characters (a long letter) and that data will (for the time being) be transferred at a rate of 300 characters per second, then a typical transfer call will be 100 seconds in length. This is not an unusually long call, but the data transfer is continuous. In normal speech, there are brief pauses. On major phone routes, calls are "interleaved", with the sound from one put into the pauses of the other (this is called "time division multiplexing"). With data transfers, there are no pauses. Thus, where you might fit five 100 second voice messages into one 100 second time slot, you could only fit one computer transfer.

A real-time conversation involves two (or more) people with terminals carrying on an exchange. Such a conversation could easily last for hours. Or two computers could be co-operating on a problem, with the same duration of contact. Such usage could, in the face of a million users, tie up large portions of phone company equipment all out of proportion to the numbers using the system.

There are many technological solutions, but almost all involve major changes in telephone company equipment and/or changing the standard ways computers communicate over telephone lines. Something will have to give--either through restrictive legislation, or the quality of phone service, or through increased cost of phone service (presumably going, in part, to increasing the amount of equipment the phone company is using), or through a change in technology.

SOCIAL IMPACT

It is impossible to accurately assess what changes any new technology (or any policy or political decision) will cause. It is clear that truly massive use of a technology is quite different than the mere introduction of a new class of devices. No superhighways were created for the first few automobiles.

It is easy to anticipate more of what is now underway: new legislation, new data services (e.g. the phone book in computer-accessible form), new programming companies, new computer service firms, various kinds of clubs and organizations... It is easy to anticipate that many of these computers will end up on shelves alongside of unused tennis rackets, trumpets and fondue pots. Nobody questions that small improvements in the quality of life of people who do a lot of writing, filing and scheduling will occur.

But will the average person's circle of acquaintances grow? Will we be better informed? Will a use of these computers as an entertainment medium become their primary value? Will they foster self-education? Is the designer of a personal computer system doing good or evil?

The main question is this: what will millions of people do with them?

I have a memo that is typical of what was going on at Apple in early 1980. It also mentions Computers by the Millions and the reactions to it. Just a slice-of-life included for background.

From: Jef Raskin

To: Tom Whitney

Date: 3 April 80

Re: Bi-weekly report

These past two weeks seem to have been particularly fruitful:

1. The meeting with Noritake electronics took place, as covered in a previous memo. The DM256x26 display, erroneously stated to cost $150 in quantities of one in that memo, will actually cost $242. I apologize for the error. I have started the ordering process for the sample.

2. [Three candidates for a programming position] have been interviewed.

[the reports on the interviews are personal and have been deleted.]

3. Starting with a tutorial session with me last Saturday, Burrell Smith has been coming up to speed in Pascal. He is writing a program to calculate transfer functions in passive circuits of up to 16 nodes. The program solves the simultaneous equations without excessive precision loss by Gaussian elimination. Such a program is needed to solve the 7-pole filter required for Justice's modem design.

In the process Smith has discovered a bug in the ATAN function, which he has duly reported to Haynes.

4. I am pleased to report that Byte magazine has placed my article in the April issue in the lead position. Thus it appears only a few pages after the Apple ad. I also received a pleasant letter from Hal Roach of the CECC, stating that they found my presentation at that conference effective and enjoyable.

5. According to their advertising, AMI will be second-sourcing the 6809E. I have tried to contact them, but my calls have not yet been returned.

6. A more unified design for the Macintosh software is well under way. Tom Malloy [LisaWrite] is quite interested, and has made some very valuable suggestions. Bruce Daniels [Lisa software] is arranging for me to present the design to the Lisa Group, and John Couch [Lisa division head] plans to attend as well.

7. With regard to the Macintosh disk interface, Smith is investigating the Interdesign MUA undedicated array. His current thinking leans toward a three- chip approach, with Justice's analog read/write chip, followed by an MUA to handle the random logic and some analog functions, driven by a shift- register/PROM state machine.

8. Ron Hochsprung, of the Lisa Group, has considerable 6809 experience, and I will be discussing with Bruce Daniels the possibility of Hochsprung's modifying the TLA assembler (under Pascal) for us. This is not an extensive job, and it will be up to Daniels whether he can spare any time for this-- considering how critical Lisa software is at this point, I will not be disappointed if it does not come about.

9. I have been testing Wozniak's 1:1 interleave, and cannot find any problems with it. This represents a most pleasant improvement in the Apple II Pascal system.

10. M&R enterprises has loaned me their SUP'R'TERM board (equivalent to Mauro's board). It interfaces instantly with Pascal, and can successfully replace a SOROC. Aside from its poor hardware design (too many SSI chips, very high power drain, 5V supply boost hack), this kind of board can cut down considerably on engineering equipment expense--and furniture expense as well. I will be testing some other brands as well. I am supplying the results of these tests to publications.

11. I am reviewing all Pubs document designs, and giving them feedback. This process has been formalized recently, and is working well. In addition I am cooperating closely with Bruce Daniels and have promised to give him comments on any Lisa documents he chooses to show me. In the last two weeks I have worked on Stein's OS, and am looking at Franklin's Query Processor.

12. A lot of feedback has come in on the "Computers by the Millions" draft, with all readers apparently enjoying it, with the exception of Steve Jobs. Many substantive comments were received, which will help guide my thinking about Macintosh. These comments will eventually result in a second draft. Markkula said that it should not be published, in order to not get our competition thinking in these directions, and that I should distribute the revamped version around the company.

Joanna Hoffman said, My contribution to the Mac was primarily in keeping it open to international issues. jef never fit the mold of getting particular kinds of people to do particular kinds of jobs, he got people who had broader horizons. Not only that, almost everyone in the group was a musician. They did not have the c.v. that most people think was necessary. To this day I look at the oddities, I got that from you. Where is the wink, the sense of humor, that edge.

END OF DOCUMENT

Jef Raskin's

The Mac and Me: 15 Years of Life with the Macintosh

Serialized in
The Analytical Engine

Journal of the Computer History Association of California (CHAC)

Part I of II

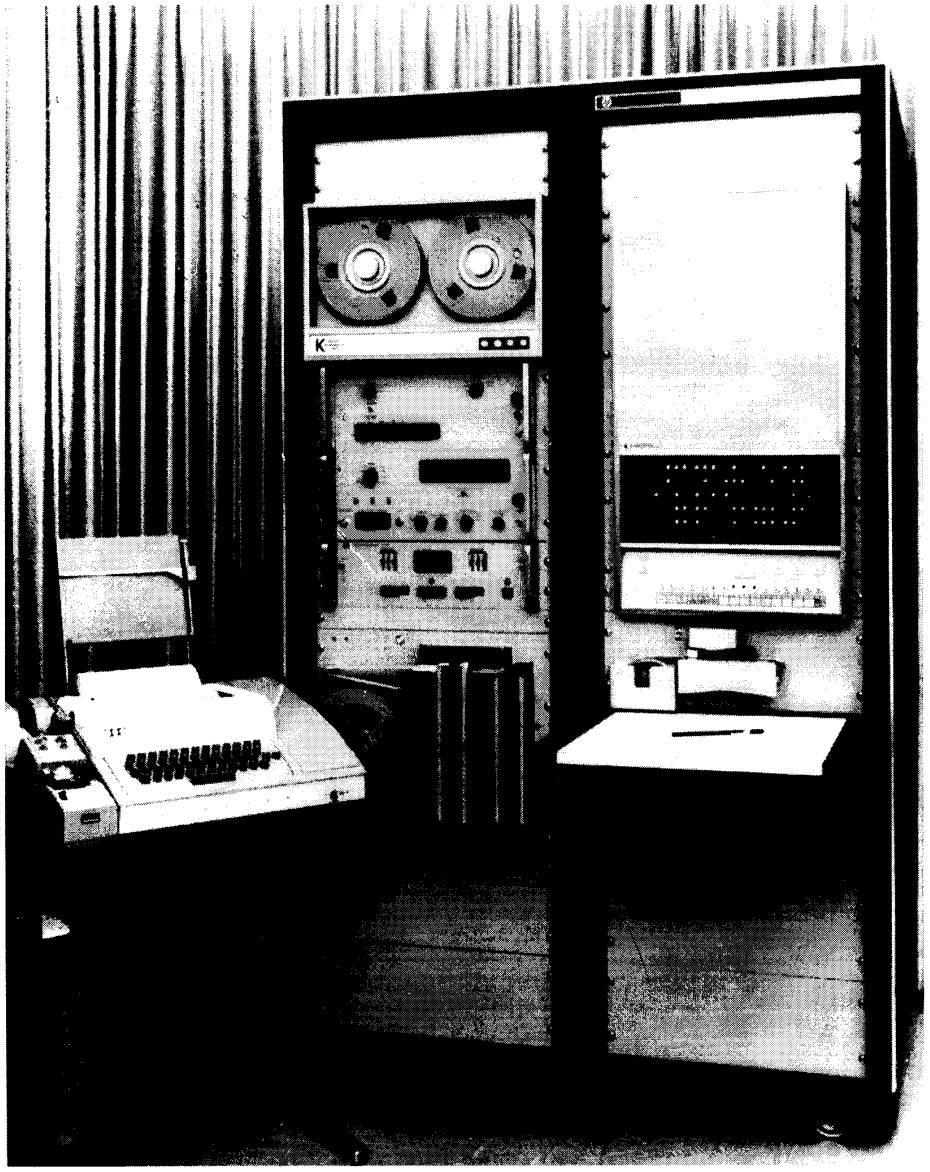
August 1995

Analytical Engine 2.4

EARLY HP: 21xx/2000
RASKIN: The MAC and ME

Analytical Engine

...AL OF THE COMPUTER HISTORY ASSOCIATION OF CALIFORNIA



Hewlett-Packard 2018 *HP Archives*

Volume 2.4 August 1995

THE MAC AND ME: 15 Years of Life with the Macintosh

by Jef Raskin

INTRODUCTION

The success of the Macintosh cannot be credited to any one person. I gave it its human-oriented, graphics-based, compact-sized nature from the very first, invented some now-universal interface concepts, and made many decisions that proved fundamental to its success. I hired a crew of unknowns who have become, almost without exception, men and women known throughout the industry for their continued innovation. It was not just me, but my original Macintosh crew of four, then a dozen or so, and finally hundreds of people, who created that first Macintosh. Now thousands at Apple continue to create and expand the Macintosh line of computers and the machines that will follow in its footprint. And even so it would have been a dead-end product, after all this effort, without the work done by thousands of software developers who give tens of millions of Macintosh users the tools they need. In this logarithmically spiraling cascade of numbers we come today to something over a hundred million people who use — at their desks at home or at work, in their schools and libraries, at the beach, in airplanes, everywhere — systems that look and feel much like Macs. Amplified by all this effort and the sincerest form of flattery, the influence of the Macintosh may well have touched the lives of over one percent of the world's population.

The phenomenon that I have just described represents the expansion of one person's stream of ideas into a flood, but the stream had to first gather force from numerous tributaries. It was not just my own inspiration, but the flowing together of the genius of Ivan Sutherland and Douglas Englebart, the scientists at Xerox PARC, the development of the microprocessor, the success of the Apple II, the efforts of many other people whose work I studied and learned from (I will never be able to thank them all) — and a lot of luck — that led to that one nexus in space-time, in the spring of 1979, when I went to the CEO of Apple and told him that I wanted to design a new product I had been dreaming of for a while, and that I wanted to call it Macintosh.

In 1994 the 10th anniversary of the introduction of the Macintosh was celebrated with a rash of articles — some of dubious accuracy — and parties at Apple and elsewhere. But it was also the 15th anniversary of the origin of the Macintosh project. This is the story of how the Mac, a product that has changed the face and interface of computing, first came into being.

THE HUMAN-ORIENTED COMPUTER SCIENCE STUDENT

It's hard to say when the conceptual framework for the Mac began. Parts of it can be discerned as early as 1965 when I was a graduate student in computer science at Penn State. Already steeped in the technicalities of computer design and programming, I nonetheless found computers aggravatingly — and unnecessarily — difficult to use, and always looked for ways to make them less intimidating. I soon earned a reputation as being sympathetic and helpful to our least technical users, especially those in the arts and humanities. Since I was (and am) as comfortable in the humanities and the fine arts as I am with science and mathematics, I never forgot our shared pain and frustration with the nonsensical ways computers were (and are) operated. In contrast, most of my fellow students celebrated their detailed knowledge and seemed to enjoy the power and status that distinguished them from "ordinary users." They preferred to work with hard-nosed programming students with whom they could attack problems in full jargon.

In my 1967 thesis, "The Quick Draw Graphics System," I took issue with the display architecture then in vogue. At this time, input was mostly via punched cards, and output took the form of extravagant quantities of oversize paper sheets from massive and noisy "line printers." Those who wanted pictures turned to expensive plotters designed to do engineering drawings. There were only a few CRT terminals at the Penn State computer center, and these could display only letters and symbols, usually in green or white on a black background. Hamstrung by specialized electronics — in particular a circuit called a "character generator" — that permitted no other use, they could not display graphics. One display at the center could draw thin, spidery lines on its large screen. With it you could do drawings that now seem crude, annotated by child-like stick-figure lettering.

In this milieu my thesis was radical in suggesting that computer displays should be graphics- rather than character-based. I argued that, by considering characters as just a particular kind of graphics, we could produce whatever fonts we wished, and mix text and drawings with the same freedom as on the drawn or printed page. To prove my point, I wrote a program that generated the complex, two-dimensional notation of music. To accomplish this, I needed to enter graphic data into the computer system.

Commercial digitizers were then as expensive as a small house; my only choice was to design and build one. Its input was somewhat indirect, in that as I pointed here and there, it produced punched cards which had to be read into the computer later. With only limited access to a machine shop and within the tiny budget a graduate student might worm from the university, I found it hard to achieve the required precision and repeatability; but my digitizer, although mechanically and electronically Rube-Goldbergish, was an inexpensive and practical one-point-at-a-time Graphic Input Device (GID). I did not know of [Douglas] Englebart, on the West Coast, and his recent invention of the mouse; even if I had, it would have been hard to hook it up to the mainframe we were using.

A mark of how much things have changed was my casual use of the word "fonts" in the paragraph above. Today, almost every computer user thinks of character display and printing in terms of fonts; but when I was a graduate student — and even when I started the Macintosh project — most people in the computer world did not think of fonts in connection with computers. When I talked about the merits of serif and sans-serif fonts, the advantages of variable- over fixed-pitch fonts, or the beauties of Bodoni's work, I got blank stares and people might mutter. "There goes Raskin with his odd art stuff again." To talk about fonts and drawing was to emigrate from computer science to the world of graphic artists, typographers and other "arts people." Now, everybody seems to have a few dozen fonts on their computer to play with; what I wished for has happened.

Another radical claim I made in my thesis was that ease of use should have a higher priority in the design of computers than speed and efficiency. Learning how to make code run in shorter time, or

less memory, or both, was central to computer science training; human interface was not given the slightest consideration. Computer time was expensive then, and pride of place for human convenience was an alien concept. It was not unimportant at that time to use internal computer resources efficiently, and it is still essential today. But efficiency should be neither an end in itself nor the highest ambition of the computer scientist — contrary to the impression one often got in graduate school in computer science.

Old-fashioned computer centers were an ideal breeding ground for pranks. Appalled by the daily waste of paper, a few friends and I once decorated the computer center building with a day's discarded output. Early arrivers the next morning found a band of white interrupting the red brick, and had to do some tearing to get into the building. The same stunt now could be considered a work of art. On another occasion a state-level dignitary was visiting the computer center. I set up the computer so that opening the massive printer cover (done by a remote command) would dump a wastebasket full of the punched-out paper chips from a card-punch into an air vent intake, giving the startling effect of a short-lived snow storm coming up from the floor. No one doubted the identity of the perpetrator (I guess I had a reputation) so sweeping and vacuuming were my lot.

My friend and office-mate Steve Zins and I engaged in a rubber-band gun arms race: my best designs were single-shot guns of unprecedented accuracy, needed to shoot the flies generated by the adjacent cow fields. Steve created a Gatling gun that could plaster my chest with some 60 rubber bands in less than a second (though the gun took five minutes to load for that one burst.) But I digress...

TO THE WEST COAST

The first truly interactive graphical computer system that came to my attention was Ivan Sutherland's Sketchpad. Though the hardware available at Penn State would not allow me to follow his lead, Sutherland's work was a revelation and an inspiration. It used a CRT display and had a light-sensitive "pen" for graphic input. In high school I had built a rudimentary light pen for an oscilloscope, so I immediately knew how it worked. One peculiarity is that you had to put up a mark of some sort (Sutherland used the word "INK") so the pen had some light to detect. You just could not start drawing without first pointing to the "ink," after which the computer could track the light pen. If you tilted the pen too far or moved it away from the screen, the computer "lost" the pen.

These details must be emphasized. Without them it is too easy to imagine, when you hear that Sutherland's ground-breaking system had "rubber band" lines and could do graphic input and output, that it all worked in the now-familiar Macintosh and Windows fashion. By present lights Sutherland's system was crude and limited. In its own day it was a wonder and an inspiration.

As I fretted with the details of getting my thesis approved, I began dreaming of a computer that would be graphical, easy to learn, easy to use, capable of everyday tasks such as word processing, and, above all, affordable. At the time this was not just a dream, but simply impossible. For a while, getting my degree also appeared an impossible dream; my thesis was rejected for not following the rules. One rule in particular was that you were to use only one font in a thesis; they didn't want you to turn out part of it on one typewriter and the rest on another. This had launched a local industry of typists who knew the university rules to the letter, and whom you paid to produce the final draft exactly to specifications, in the required number of copies.

My thesis had characters in several fonts, exactly to demonstrate that one could produce distinct fonts on a graphics-based system. The use of varied fonts was, I complained, part of the subject matter; to rule out their use was to attack the content, and not just the form, of the thesis. After months of verbal wrangling and a memo war, my thesis was accepted, fonts and all.

Tired of Pennsylvania winters and Penn State's cold bureaucrats, my wife and I drove west, until we ran out of land in La Jolla, just north of San Diego. We knew nobody in the area, but by luck had ended up at the University of California's Scripps Institution of Oceanography. Walking out onto the Scripps pier I saw, for the first time, a pelican abruptly fold its wings and splash into the ocean. I thought it had been shot.

EARLIER INFLUENCES

Sometime in middle or early high school I was given a copy of Claude Shannon's marvelous *Information Theory*. I can remember no other books from that time, by both title and author, except Arthur Conan Doyle's Sherlock Holmes mysteries. It was eye-opening and completely wonderful to learn that this ephemeral, seemingly unquantitative stuff called information was amenable to a physics as rigorous as that for objects and motion — and that this physics was almost purely mathematical in its development. This was extremely appealing, as mathematics was by far my first love, queen of all the subjects I could command. It may seem premature for one at the age of 14 or so to be so smitten; but it was my good fortune that Ron Genise — a most wonderful teacher, and later friend — had begun, in my sixth grade, to make the beauty and power of mathematics as alive and vivid for me as the performance of sports figures and cars were for my classmates. I believe that he first pointed out the Shannon book to me, and if my memory is astray on that point, at least I know that he led me to the intellectual point of view from which I could appreciate it.

During this time I read an article about the rate at which information (measured, as Shannon had shown, in bits per second) could be communicated from the eyes to the brain. The number seemed much too low. For example, I could sight-read pieces by Chopin and Beethoven on the piano. As an early exercise in information theory I calculated the number of bits in each symbol. This is not difficult; for example, a note head specifies one of the 88 notes on the piano, and this takes a little over seven bits. It also specifies a duration, which for most practical purposes has one of 8 values, which is exactly three bits of information. So a note conveys approximately 10 bits of information.

Ignoring other symbols and some details here (I was more precise in the paper I wrote at the time) and considering music where one is reading four chords each of six notes in a second, implies a transmission rate of 2400 bits per second (2400 baud.) This exceeded the rate at which neurophysiologists believed the senses could transmit data. This made me realize that I wasn't really reading each note, but analyzing the chords into harmonies: "that's an E-flat major chord in the first inversion..." and so on. All my brain had to deal with was the single concept of a certain chord, and not all the details of each note. Later, reading about psychology, I found that I had re-discovered a phenomenon called "chunking" which allows us to grasp much more than would be indicated the slow data rates experiment shows our brains can handle.

This has been a paradigm typical of my entire life; supposedly different disciplines merge or interact, reinforcing each other. Studying math opens the path to a book on the physics of information which informs my classical musical studies, enhanced by my having ignored my music teacher's wishes and learned to play from jazz "charts" instead of sticking only to classical music. The speed at which I can read music seems to violate a fact I read in a science magazine (I am a compulsive reader, and will read almost anything,) the solution to which gives me an anchor of understanding when I am learning about something in psychology years later. Somehow it all fits together.

FAMILY AND FEMINISM

My parents brought my brother and me up to recognize oppression and to fight it. Popular, gregarious, and very active in civic events, they risked friendship and fortune in defending racial equality in the 1950s and 60s.

The following is part of the column I wrote about my father for the local paper:

My father, Bill, died last week. It was not at all unexpected. His health had been failing since my mother died a few years ago. Recently he had had a stroke and was also diagnosed with congestive heart failure. He could barely speak. My brother Michael had flown in from Boston and the three of us were together for what was to be the last time. Bill struggled for speech and repeated, "What can I say? What can I say?", a phrase he had always used when overcome with emotion. We told him that he didn't have to say anything, but he finally said, with evident effort, "I love you." and embarrassed us a little by taking our hands and kissing them All I could think to do was to return the gesture and kiss his hand. At this he smiled his delightful smile, made lop-sided by his stroke, yet a smile that reminded us for a moment of the father he had been.

If you had met him you'd have found a mild-mannered man, soft-spoken, well-liked and without self-interested ambition. The love between my parents was constant and evident to all who knew them. A responsible citizen, a merchant, a member of the school board after my brother and I had gone on to college — he would not be on the board while we were students to avoid tainting our achievements with suspicions of favoritism. For many years he was the secretary of the Lions Club. He never accepted the many nominations to be president. He liked to lead by example, by quiet persuasion, and with gentle humor from the sidelines.

On moral issues he was inflexible; I have space for only one example. In the '50's, long before the present civil rights movement was in full steam, he incurred the enmity of nearly the entire town by supporting the hiring of man of African descent as an English teacher, and backing another as a member of the Lions Club. I remember a long-time customer coming in to our store and saying, with true regret in her tone, "I can no longer shop here. You understand why."

We had a solemn family meeting. Our parents told us that we had a choice to make: if we continued to back our beliefs we would be very poor for a while, there would be no toys at year's end, no going to restaurants, and so forth. We knew what he meant, our family-owned store was too often quiet, the piles of layaways for Christmas were not building up in the basement as they had in previous years. The other choice, he said, was to hold onto our beliefs privately but not push matters. He would not impose his values and the attendant risks on his children. It was up to us and we knew he would abide by our decision. Michael and I had no patience with people who judged others on their race or cultural background, and we said (as Bill proudly recounted years later) without hesitation that we didn't care about presents but that we did care about our friends. To do nothing was to give tacit approval to racism.

Some would say that we lost. We had to sell the store across from the railroad station and set up shop in a poorer neighborhood. Instead of big Buicks and Packards we drove the cheapest Renault. We no longer had a summer house by the lake. And the lovely presents we had become used to came no more. The Lions club split in two, a large whites-only club and a tiny integrated one with my father as secretary. But we won. The high school had its first black teacher, and others followed. The white Lions club faded and Bill's survived.

There were no services; he was an atheist as righteous as any church-goer. He donated his body to science, and his love of humanity to his sons.

The messages were clear. One was: figure out what is right and then stick to your guns. Another: principle is more important than practicality. These were two of the beliefs that propelled the Macintosh as it came into being. I was not uncomfortable defying common wisdom.

This early training also made it easy for me to recognize girls and women as an oppressed class in our society. As a child I had seen my intellectually brilliant cousin, Miriam, given dolls while I would get the far more interesting Erector sets and chemistry labs we both preferred. My feminist leanings were deepened by some of the things that later happened to my friend Karen Kalinsky. For example, when we were undergraduates at S. U. N. Y. at Stony Brook, I held a job in the computer center. Karen, also a math major, became interested in computers and decided to take the programming course; the head of the computer center there had offered jobs to the people who got the three highest scores on the final and Karen was rarely outscored on any test. Looking at the posted list, we saw that she had received the top score, and we anticipated working together at the computer center. The jobs, however, went to the three top men in the class! We were mad, she raised a ruckus, and I quit in protest.

We went to Penn State next. As usual, I got a job at the computer center, and as usual, they wouldn't hire her — in spite of credentials better than mine in some ways, such as grade point average — because her "boy friend" worked there. After we were married, there was no way she could be hired. Nepotism, you know.

By the time we reached the west coast, degrees in hand, we were smarter about jobs. She took a position first, running the computer at the Institute for Geophysics and Planetary Physics at University of California at San Diego (UCSD.) Shortly thereafter, I got a job at the University Computer Center.

They didn't think to ask a man if his wife worked in a professional capacity — on the form it only asked if your husband worked at the University. I also saw some of the sexist hurdles my cousin Miriam had to face, such as being ignored by the professors in classes. (Miriam is now a professor and researcher at the University of Michigan at Ann Arbor.) These experiences are relevant to our story; for one thing, they influenced my choice of the name "Macintosh" for my favorite computer.

While working at the UCSD computer center, I became familiar with other parts of the school. The music department, filled with avant garde composers and performers, was intrigued by my background in both computers and music (I had designed and built the first electronic music studio at Penn State.) It looked like a good fit and I became a graduate student there, working toward a Ph.D. in music. UCSD, like some English universities, is divided into colleges; at the time, the first two were named Revelle and Muir Colleges, and the newest was simply called "Third College." By a peculiar series of events (that would, for once, take us too far afield if I described it here,) I soon became the computer center director and a professor of Visual Art at Third College, positions I held from 1969 through 1974.

The main computer center at Revelle College was noisy, antiseptic, and lit by fluorescent lights that glared off white vinyl floors. It was punch-card-oriented and built around a physically huge, multimillion dollar mainframe computer. The computer center I designed for Third College, located in a war-surplus Quonset hut, was very different. It used a pair of Data General Nova minicomputers with 16 interactive terminals. The decor was beanbag chairs and Japanese paper lanterns, giving my center a friendly, funky feel. It became the natural home for people with what were then seen as "odd" computer applications, like music and art. Some of the campus's computer aficionados found that they preferred the unhurried, interactive context of the minicomputers to the fluorescent, buzzy mainframe environment on the other side of campus. In light of today's personal computers, which operate in homes, cars, and at the beach, it is hard to remember that in the early 1970's a computer center such as the one I created was countercultural, and perhaps unique.

THE THIRD COLLEGE COMPUTER CENTER

My computer center was funded primarily by the National Science Foundation and the University of California. I suspect that if Senators Proxmire or Helms had ever visited it they would have mistaken it for a typical waste of taxpayer's money. It looked more like a place to get stoned than to get educated, a hippy haven.

But looks are just looks, and I have always made my courses friendly in spirit while I demanded hard work from the students. The new computer center was an effective educational facility. Only ten per cent of the undergraduates who learned programming at UCSD went through my courses, but nearly half of the students who held paying jobs at the main computer center had done so. There was no doubt that the Third College computer center was doing a good job at creating future computer scientists. Better still, it was reaching students who would otherwise never have gone near a mainframe, just as the Macintosh would someday be used by people who thought they'd never touch a computer. It was not the technology that made my teaching so effective, but the interface! Students learned more and better in a pleasant environment where, to test their programs, they simply pressed a key and got results. The other side of campus was batch-oriented; you presented a deck and went to the printer to await your output. Fortunately Ken Bowles, the director of the Revelle center, had an enlightened attitude for someone in his position at the time; he did not regard a second, student-oriented computer center as a challenge to his hegemony. Years later Bowles would spearhead development of the computer language and operating system called UCSD Pascal, which would be essential to the success of Apple and the Macintosh.

In retrospect, the Third College Computer Center was all that a grant administrator could wish for: it met its educational aims, resulted in appropriate publications, and later went on to inspire commercial products that have boosted the GNP (gross national product) to the tune of billions of dollars. It is definitely possible to see precursors of the Macintosh in the Third College center's low tables with small rectangular monitors and detached keyboards. Though they had to be tied to a common system, the effect was as if each student had a per-

sonal computer. Resources had to be shared in 1973, when a 4K byte (enough to hold about 800 words of English) random access memory (RAM) memory unit cost nearly two thousand dollars. As this is written, each 4K bytes of RAM in my Macintosh computer costs less than 10 cents.

During the summers, I used one of the Novas as my personal computer. My mostly volunteer staff and student friends (notably Jon Collins, Barbara Zakarian, Bill Atkinson, and Steve Clark) helped me put the computer into the back of my truck on a wheeled dolly, and we used it wherever we went. One memorable time we took it with us into a restaurant, using it to figure the bill and the tip, to the amazement of the waitresses and patrons who crowded around. A computer outside of a lab was an absolute novelty. These experiences with a "portable" computer system gave me a foretaste of what it would be like to own a personal computer. Like the crocodile in Peter Pan, I would never forget that taste, and craved it for years.

SAIL AND SILICON VALLEY

In 1972 I visited the Stanford University Artificial Intelligence Laboratory (SAIL,) which had an established reputation as a center for advanced research in computer science. I was also introduced to another magical place that had recently opened and was a short bicycle ride away. The Xerox Palo Alto Research Center (PARC) was to become even better known than SAIL in the coming years. Thanks to a strong common interest in early music as well as computers, I soon found a close friend in the person of Doug Wyatt, a tall, thin man who is as quiet as he is technically brilliant and musically talented. Doug took a leave of absence from PARC and came down to San Diego for a while to write new software for my computer center.

A programming language I designed, "FLOW," was implemented and improved by Doug. The human interface used in this system, as well as the design of the language itself, were somewhat ahead of their time. It proved so effective that it came to the attention of the cognitive psychologist Don Norman, later to become a leader in the fields of cognitive psychology and man-machine interfaces, who is now an Apple Fellow and a writer of popular books on the subject. Norman did some of his first computer-interface-related work investigating why students learned faster and better with the "FLOW" computer language.

The next summer I was invited to become a Visiting Scholar at SAIL. It was a great place to be. I remember fondly the memorable daily, end-of-the-day volleyball game, after which most of us would retire to the lounge and watch *Star Trek*. After which a lot of us would get supper and go back to work. There are a number of reasons why I remember watching *Star Trek*. I enjoyed the show, and once it led to a remarkable incident.

To understand what happened, you have to know that an experimental robot occasionally roamed the halls and parking lots at SAIL. The rambling robot (in case you were picturing C3PO walking across the desert with R2D2) looked like a wheeled table full of surplus electronics. It was not at all humanoid, or even robotoid. On this occasion we were sitting down to watch Captain Kirk and his enterprising crew when the robot wandered in, stopped, swiveled its TV eye at the set and sat there throughout the show. At the end, it whirred into life, rolled itself around, and left as we did. Later I learned that Hans Moravec was working at a terminal that did not have TV feed (most terminals at the AI lab did — another development way ahead of its time that I was exposed to) and had sent in the robot to beam the picture and sound back to his monitor. It occurred to me that we were probably the only people in the universe watching *Star Trek* in the company of a robot.

While at SAIL I used the early Defense Department progenitor of the now-popular Internet. ARPAnet allowed us to communicate with and use remote computers. It felt like magic to be sitting in California and running a computer at MIT. I became an early e-mail junkie, a habit that I have yet to kick after 20 years.

PARC

The populations at SAIL and PARC intermingled freely and I found myself gravitating more and more toward the beanbag chairs at PARC. A significant portion of their work was based on the same goals as my own, to make the power of computers accessible to non-specialists. I suspect that I fit in easily and well because they didn't have to start by converting me to their point of view; I was already there. I meanwhile felt at home since, for

the first time, I was among computer scientists who were on the same wavelength as I. They had accomplished independently what my thesis had called for a few years earlier: computers that were graphic-based, without impediments such as character generators. What was more exciting was that the people I spoke with were concentrating on interface design, which I also saw as the area of computer science most in need of development.

By 1974 I was fed up with the politics in the UCSD art department and left the University, making my point in artistic fashion by ascending in a huge hot air balloon, playing the sopranino recorder, and announcing my resignation from 100 feet up in the air. I was also tired of the directions the computer industry was taking. It was all "more" and "bigger" and "faster," but not really better (this history is being repeated with personal computers today.) Nobody seemed interested in what I was preaching about usability. I sold my house near San Diego and moved to Brisbane, a town just south of San Francisco. I tried the life of a street musician and music teacher, started a company that made radio-controlled model airplane kits (a business that continues today,) and became the conductor of the San Francisco Chamber Opera. I also worked briefly as a packaging designer, but left when I couldn't convince the owner that we could design boxes faster and better with a computer. In the 1990's the owner's son, who better understood what I had proposed, wrote a set of computer programs to do box layout, and now has a successful business making boxes — and an even more successful one selling the software.

I also worked as an advertising and portfolio photographer (having been taught a bit about the art by my former student and forever mentor David Wing, now a professor of art at Grossmont College east of San Diego.) During this period of wandering, I started a company called Bannister & Crun to write software and manuals. The company was named after two characters (Minnie Bannister and Henry Crun) featured on the BBC's beloved Goon Show. Between the way the Goon Show's players mangled English and the spotty reception of my shortwave radio, it was sometimes hard following their humor; they were to radio what Monty Python was to become to TV.

Our first job at Bannister & Crun was to computerize the South San Francisco sewer billing system, a job that required me to visit the sewage treatment plant from time to time and work on one of the most dreadfully designed computers I had ever seen, an early Qantel model. We also worked on other software projects and wrote manuals for companies that included National Semiconductor and Heathkit.

ENTER THE MICROCOMPUTER

In late 1974 the general purpose microprocessor chip was put on the market and I remember discussing its incredible potential with Doug Wyatt and a mutual musical friend, a talented and extraordinarily pleasant man named Brian Howard. Of broad learning, with a degree in Electrical Engineering from Stanford, he was working for the preventive medicine department at the university, doing a characteristically wide range of things including building test equipment. Brian was to become a central intelligence in the development of the Macintosh and, later, one of the designers of Apple's first laser printer — another product that changed the face of computing. He continues as a respected engineer at Apple.

When the first microcomputer kit, the MITS Altair, was announced in 1975, Brian, Doug, and I just had to have one. With soldering iron, oscilloscope, and logic probe in hand, Doug and I built the Altair and (somewhat to our surprise) got it working. This was a non-trivial endeavor, but Doug's combination of methodical care and clever insight solved many a problem. I was experienced with a soldering iron and felt comfortable with the construction because electronics had been a hobby of mine as a child. I had won a science fair prize for a computer I made while in high school. Building a computer is, perhaps, nothing to crow about now, but in 1960 individuals just didn't have computers and kids didn't use or program them. I was still a hardware jock as an undergraduate, designing and building a computer from scratch for the Biology department at the State University of New York, then at Oyster Bay (now Stony Brook.) This background proved useful when creating the Mac, since I had a realistic idea of what could and could not be done with electronic components. Having done electronic design and testing myself, I could communicate with electronic wizards, and not be snowed when they spoke of impedance or logic levels.

We got the Altair running a program that did stock market analysis, and we sold it for about $5,000 to Jim Hurst, a stock market guru. He'd been paying $10,000 per month in time-shared computer charges for the same work, so the micro system amortized out in two weeks — a great savings to him. The computer had cost us a few hundred dollars and had served well as an introduction to microcomputing. With part of our profits we bought a slightly more sophisticated IMSAI, and I built the first modem kit that became available. I made back a bit of the money I spent on that kit by authoring a review of it for *Dr. Dobb's Journal*. I liked reviewing kits and I was soon writing the "Consumer Notes" column for that magazine. I became a reporter on the early personal computer scene; pieces I wrote appeared in *Personal Computing, Interface Age,* the *Silicon Gulch Gazette, Kilobaud, Datamation,* and *Byte* magazine.

Jim Warren, who ran the wonderfully-named *Dr. Dobb's Journal of Computer Calisthenics and Orthodontia* (Running Light without Overbyte,) is a delightful, jovial, and unconventional man who sparked much in the industry. He created the West Coast Computer Faires and now works on creating political enfranchisement through technology. He often managed the Faires by cruising their huge exhibit halls on roller skates. One of the assignments he gave me in 1976 was to interview two fellow-members of the now-legendary Homebrew Computer Club centered in Palo Alto. The club, many of whose members went on to become prominent in the computer industry, was moderated by the very funny and genial Lee Felsenstein. This was also the man who designed the modem I reviewed (it's a small valley.) Doug Wyatt and I got an ovation one night when I announced that we had run our IMSAI for over a month without once taking the top off to fix something; it was a real milestone (and a tribute to clean soldering.)

THE TWO STEVES

The members I was sent to interview were building a new computer in their garage. By coincidence both were named "Steve" and their project was the Apple I. I was impressed by Steve "Woz" Wozniak's brilliant and efficient design and predecoded bus concept, and his exposition of the advantages of the 6800 and 6502 architecture over the competing 8008 and 8080-based machines. (Incredibly, this competition between architectures continues to this day.) I remember Woz explaining how the pre-decoded bus made peripherals simpler, that you could send information to peripherals the same way you wrote to memory, and that memory wasn't paged — unimportant details in this essay perhaps, but indicative of the kinds of considerations that Woz paid careful attention to. And I loved the name "Apple" instead of the techie names everybody else was using; it fit my kind of iconoclastic spirit. Now we have become so accustomed to it that it is hard to remember how joltingly countercultural that name was at first. A computer company named "Apple"?

The other Steve, Steve Jobs, was a delight to talk to about less technical aspects of computers. His enthusiasm and business orientation were exciting. They were just starting on the design of the Apple II, and I tried to convince them that they should employ bit-mapped graphics and not have a character generator, but Woz thought that software couldn't handle the character generation task fast enough and Steve Jobs didn't understand why I thought it so important. I had a different vision of what a microcomputer should be like, and PARC's programmers and my own work had convinced me that software could do the job. I tried to convince Woz by working out the code to put bit-mapped characters on the screen and calculating timings by counting cycles, but the Steves were not open to the idea. The concepts I espoused were far from the mainstream of computer design and for all their mold-breaking thinking, Steve and Steve were very strongly conditioned by the minicomputers they had seen. To do them justice, Woz was absolutely correct in stating that a character generator was much faster and its software less memory intensive than my all-graphics approach. But had I been able to convey my vision better, I suspect he could have made bit-mapping work fast enough back then.

It was by the slimmest of chances that the Apple II had a high resolution graphics mode (Hi-Res) on which bit-mapped graphics could later be explored. Woz was not going to include it but Jobs asked Woz how many chips it would take to add the feature. Woz said that it would take only two, so Jobs insisted that they could afford it. Sometimes history stumbles along from accident to accident, things are done that seem like a good idea at the time, and every now and then they are.

I tried to convince Jobs and Woz to visit PARC, which was a very academic and open place (Xerox may later have felt that PARC was too open,) but did not succeed. Jobs repeatedly told me (and anybody else he could get hold of) that a large corporation like Xerox couldn't do anything interesting. Hewlett-Packard's rejection of Woz's proposal for a personal computer, when he worked there, was a prime example of such corporate blindness, and ever after remained part of their psychological motivation. If I could have told them then that Hewlett Packard would someday make millions of dollars simply by selling peripherals to Apple computer products (as has happened,) the Steves would have been ecstatic, and rightfully so.

APPLE MANUALS

I worked with Jobs and Woz and, under the aegis of Bannister & Crun, wrote a user-oriented portion of the manual for the Apple I. There was a tiny misunderstanding about the price: I was talking about $50 per finished page and they thought I had said that it would cost $50 for me to write the whole manual. We resolved our differences amicably and work proceeded.

At about this time Jobs made a decision crucial to the history of personal computers. Paul Terrell ran the Byte Shop in Mountain View, one of the first retail computer stores in the world. When Jobs and Woz asked his advice he insisted that the Apple II must have a non-rectilinear, consumer-oriented, plastic case and — unlike the Apple I and many of its competitors — should never be sold as a kit for which potential users had to scrounge parts at local surplus or electronics stores. I well remember the hassle of finding keyboards that worked properly with the very early microcomputers, and connectors to fit the idiosyncratic circuit boards. The Apple II, Terrell suggested, should circumvent these problems by being factory-built with an integral keyboard and power supply.

Terrell was not quite alone in recognizing these desiderata. The SOL, designed by Lee Felsenstein and manufactured by Processor Technology, had a typewriter look. A Utah company, Sphere, sold a complete little machine with a programmer's hexadecimal keyboard (base 16 numbers only) and included a video screen even before the Apple I was released. The Commodore PET and TRS-80, both designed with much attention to consumer needs, followed soon after the Apple II. But though Apple was not alone, it had important advantages. One was Woz's remarkable BASIC interpreter with color graphics commands embedded in it. Another was that Apple was led by a raving firebrand in the person of Steve Jobs — which was just what the industry needed.

Bannister & Crun was engaged to write the manual for Apple II's BASIC. This gave me the chance to put some of what I had learned as a computer science professor into a vehicle that I believed would reach tens or hundreds of thousands of people in a few years. It had been hard to give up teaching, which I love and yet hope to get back to, but (as I wrote to my parents) I felt that I could do more good for education by working at Apple than in any other way open to me.

The BASIC manual, first published in 1978, turned out to be a trend-setter. Instead of starting off with the then-customary explanation of the internal architecture of the computer, it got right to what people had to do to get the product working. It first explained in a step-by-step manner how to hook up the computer and use the keyboard. It then quickly moved the learner into doing color graphics (in 1978!). Another first: the manual used color illustrations and photos.

Today, when computer products are graded by magazines on the quality of their documentation, it may be surprising to learn that the nascent company barely saw the need for an Apple II manual at all. Mike Scott (Scotty,) a large man whose occasionally high-handed manner and gruff speaking style could be intimidating, had come from National Semiconductor to be Apple's president. He said, half seriously, that at National they had done very well with one-page data sheets, and I could save the company a lot of money by doing likewise. I soon learned that in spite of his manner, he was open to cogent arguments. Later he was to protect and nurture my Macintosh project when it was at a delicate stage.

Writing user documentation was a perfect prelude to creating the Macintosh at Apple. Doing manuals forced me to look at each product — in excruciating detail — from the customer's point of view. It is an experience I wish all computer and interface designers could share. Any design flaw that interferes with learning or using the product becomes painfully apparent as you struggle to

explain the quirk to the user. Time after time Brian Howard and I would wrestle with these problems, our frustrations coming out as subtle, snide remarks about design errors — remarks that, in those innocent days, often appeared in our manuals. This sometimes annoyed marketing people, but it actually served the purposes of Apple's products. The comments told the truth, showed sympathy for the customer's plight, and created credibility for the rest of the manual and the company as a whole. Too many manuals are fairy tales about how a product is supposed to work, or how it worked in the previous version.

When we wrote the Apple II manuals at Bannister & Crun the product was already finished — we weren't trying to write a manual from specifications or rough prototypes. Trying to document a product still in development is an often-made mistake which guarantees a second-rate result. Since almost everybody now does this, customers have come to accept such manuals as standard. It can't work well, since you are documenting something different than what the customer will get, you are writing about a fiction, a planned product (and we all know that products always turn out exactly as planned.) To be sure, the manual can be edited to conform with changes, but that is not nearly as good as having the whole picture in mind from the beginning. Besides, you never catch all the changes, as the customers eventually find out.

We weren't the only group writing good manuals; another example was the superb HP 35 manual. The HP 35 was the first scientific pocket calculator, another fabulous product that opened up an industry. Its manual was an inspiration in terms of writing, use of color and layout, and informal conversational style. Instead of a lecture about the calculator's remarkable stack architecture or revolutionary custom electronic chips, the manual started you out punching buttons and seeing what happened. Later, when the topic had some value and experiential basis, you were given a mental model of what was going on inside. Along with my Apple I and Apple II (serial number 2) I keep my HP 35, still in working condition, in my office. The inspiring manual is displayed alongside it.

The Apple II manuals also worked because they were tested with typical users and rewritten as necessary, a concept nearly unique at the time in the computer industry, and one now regularly abandoned (to the detriment of users everywhere) under the excuse of time pressures. The time thus "saved" is not always a win; what the manufacturer gains by a few weeks' shorter product cycle is lost doubly — to customer dissatisfaction, and as a continual drain on the bottom line attributable to increased support costs. As CEO of Bannister & Crun I demanded that I have a real product in its packaging before I wrote a manual, and in those days I got what I asked for. Errors in the manuals were extraordinarily rare thanks to these procedures. Brian Howard turned out to be a great editor, along with his other talents; his comments and those of Doug Wyatt were my education in how to write clearly and simply. My writing has never achieved the standards they set, but inasmuch as it is better than it was, they — and the many other people who have since bravely waded through my first drafts and let me know in no uncertain terms that a lot more work was required — deserve a lot of credit.

APPLE'S MANAGEMENT

Having already approved the use of high-quality, coated paper, four-color illustrations, two-color printing throughout, and full-length manuals, management was reluctant to support the use of a wire binding so that the manual would lie flat (at least we had gone beyond flat lies.) I had often observed that most users didn't have a third hand to hold the manual open as they typed. During 1977, when I was merely a vendor to the company, two key people supported my point of view: co-founder Steve Jobs and chairman "Mike" Markkula, also known by his initials "ACM." Markkula had profited significantly from his experience at Intel, not only financially, but as a well-polished manager. Not having been in industry, I had never worked with a person of his extraordinary business skills, and I am still striving to live up to some of the examples he set. For one, he always gave the impression of having all the time in the world to hear what I had to say. He proved that he was listening, either by acting on my suggestions, or by taking the time to explain to me why they were not good ideas. One of my not-very-good ideas was to lower the price of the Apple II. I had been upset when I figured out how large Apple's margins were. Markkula patiently explained that while Apple's products were more expensive, the resultant financial strength of the company meant that

Apple would be there for its customers in the future. It would have the money to develop new products and successfully market them while its competitors, who seemed to be doing a favor to their customers in the short term, would soon be out of business. He was right.

Jobs, in the early days of Apple, was an adamant protector of my writer's prerogatives who championed the need for testing and revision when Scotty didn't see things my way. Thus protected, I did the manuals as I thought they should be done, and Apple got what the press — and even other companies — praised as the best manuals in the business.

DEPARTMENT BUILDING

In mid-1977 I was still running Bannister & Crun, but my writing for Apple and the magazines had made me pretty well known in the industry, and I had lots of job offers. Chuck Peddle, leader of the PET computer project at Commodore, wanted me for a position there. Steve Jobs, who is as persistent a person as I've ever met, kept on asking me to join Apple as head of their publications department. I repeatedly declined, and he eventually asked what it would take to get me to join Apple. To put him off I made an impossible list which included an office with a window and a musical instrument, time to play gigs (I didn't want to let my musician friends down,) flexible hours, Apple's hiring everybody at Bannister & Crun who wanted a job at Apple, and so on. He simply agreed to all my conditions, which I then wrote down; as it turned out, I should have done this with more of his promises. Bannister & Crun became Apple's publications department with me at its helm. I joined on the 3rd of January, 1978 as Apple's 31st employee. I presented no resume and signed no forms. Apple did no checking on my background. That I had led a team that had produced the nascent industry's best manuals was enough.

One day I heard that a new product, called the Apple II Pro, was being put together in the lab. From some technical details, I surmised that it could not possibly work as expected. So I snuck into the laboratory and turned on the prototype. Sure enough, it didn't do what it was supposed to. I went to Mike Markkula and told him that the machine wasn't up to snuff, and he replied that I couldn't be right, his engineers had assured him that all the problems had been solved. He was actually making plans for marketing and shipping the product and was about to start taking orders from dealers.

I took him to the lab and demonstrated my discovery. Upon talking to the people working on the project I had discovered a classic management nightmare (though it was new to me at the time): the engineers working on the project said that while the project was mostly going OK, there were still some unresolved problems. The next level reported to their bosses that a handful of problems would no doubt be rapidly fixed; they in turn told Scotty that it was nearly done, and Scotty told Markkula that it was just about ready to roll. In a more mature company I would probably have been fired immediately for my end-run around the hierarchy, but this time I was able to make a case for a "New Product Review" department. This would do for systems and software what QA (Quality Assurance) programs did for circuit boards and mechanical assembly. Suddenly, I was managing two departments.

Computers are not terribly useful without software (my definition of a computer is "a box for running software".) I argued that Apple would need to provide something new, application software, if we were to sell computers more widely. I created what may have been the first application software department at any microcomputer company. I tried to convince Apple to buy Visicalc, the first spreadsheet, when it was offered to us, but was outgunned by Jobs and Markkula. It was Markkula's theory — at least as he expressed it years later — that to become a major application provider would have put a damper on third-party software developers, in the long run hurting Apple. What he said at the time I do not remember, but I do remember remaining unconvinced. With Markkula's approval I took a brief leave from Apple, arguing that if I could help make Visicalc a winner, Visicalc would sell a lot of Apple II's. As a result, I got to write the tutorial portion of the Visicalc manual, reporting to Dan Fylstra. Visicalc did sell a lot of our computers, established a new category of software, and — since it was a business application — greatly helped the credibility of microcomputers in general.

In 1979 I found managers for two of my departments and became manager of Applications Software. Meanwhile, I was chafing at the limitations

of the Apple II. The publications department managed to keep a secret that would have been embarrassing to the company had it been revealed at the time: the publications department was using not Apple IIs but Poly 88 computers. We were running a word processor I had designed and which had been implemented at Bannister & Crun. The Polys were a competing microcomputer that could handle both upper and lower-case letters — a necessity in manuals. Due to mediocre design (they had no Woz,) poor marketing, and less imaginative management, they were soon out of business. Back in the garage days, in 1976, I had argued that the Apple II must have lower-case letters, but Woz disagreed. I held that the single biggest use of microcomputers would be word processing, he claimed that they would be used for game playing and programming in BASIC. But he had the ultimate argument: upper-case-only character generators were lots cheaper.

Though I often felt they were on the right track, I still could find myself at odds with Apple's founders, who were a strange mix of the radical and the conservative. They wanted to create personal computers, but expected them to work much like the hard-to-use minicomputers from DEC, HP, and Data General. Dragging the two Steves into the interface future was preaching in an unknown tongue, and from my perspective, they didn't appear to be the advanced thinkers that they were made out to be in the press. They were visionary, and working like mad to drag the world into the personal computer future, it's just that I was a few years further out in the future. In spite of these differences I was the typical way-over-100%-effort and totally Apple-oriented employee. This extended into my personal life. Apple's Cupertino phone number was 996-1010. When I moved to Cupertino, I chose my home phone number, symbolically, to be just one step ahead of the rest of Apple: it was 996-1009.

BITMAPPING

Apple employees were a diadem of the brightest and best cut jewels of Silicon Valley, some well known and some newly discovered. I was amazed at the competence of the people, whether in financial management, marketing, manufacturing, engineering or whatever; and they all seemed willing to share their knowledge and points of view with me. Competence clustered at Apple, partially thanks to the many contacts men like Markkula and Scott and our investors had in the industry, and partly as a result of Steve Jobs' incredible persistence. When Jobs was convinced he wanted someone, that person would be hounded to death, complimented, provided blandishments suited to his or her nature, and offered the world. Soon enough, Apple could deliver many of these promises. At NeXT, Jobs was to continue making similar promises, repeating the ploys he had developed in Apple's first years, to the disappointment of investors, employees, and customers alike. Too often we mistake the randomness of the universe as our own accomplishment when things go our way. Still more often we take that same randomness, when it goes against us, and regard it as punishment for our sins. In a complex world it is often impossible to tell accident from design.

When Ken Rothmuller was hired from HP to start the Lisa project (which was after I had proposed the Macintosh, but before it was officially approved as a research project) I saw a new opportunity to get my computer interface and architecture ideas accepted. I argued again that the screen architecture of this new product should be bitmapped. But where I had failed with Woz and Jobs, I managed to convince Ken and his crew — probably to Ken's detriment as Jobs found him difficult to work with (i.e. had strong opinions and didn't kowtow) and fired him. Jobs probably found me equally difficult, but I had already proved myself and my very productive and cost-effective publications department was one of Apple's many gems; it would have been hard to justify getting rid of me.

In spite of the loss of Ken Rothmuller, the bitmapped screen survived. This was a key win for me and (though they didn't know it at the time) for Apple, because it would force the software I was dreaming of to be implemented. No longer would computers be restricted to whatever font was in the character generator, and have to treat characters and graphics as fundamentally different kinds of things. Another major battle that I fought was to have black characters on a white background instead of the then-conventional white (or green!) lettering on a black background. The Lisa hardware designers were, like Jobs and Woz, dead set against this idea, noting that it took too much power, would require a higher refresh rate to avoid

flicker, was not the way computers usually worked, and so on. I argued that people often printed computer output on white paper, and that was black-on-white, and that if you wanted it to look the same on screen and print (the WYSIWYG, or What You See Is What You Get principle) you had to do it black-on-white. But my industrial-strength argument had to do with something the Lisa crew (like the whole micro-computer industry) was just not thinking about: grayscale, or dithered, graphic images. If you worked in white-on-black and had a part-text and part-graphics image on the screen, which got reversed on printing, then either the screen or paper image would have to be a negative, and nobody wants to be forced to look at negatives.

Again, after many memos, meetings, and informal and formal discussions, I managed to sell the idea. It was another key to the future.

Jef Raskin's

The Mac and Me: 15 Years of Life with the Macintosh

Serialized in
The Analytical Engine

Journal of the Computer History Association of California (CHAC)

Part II of II

May 1996

Analytical Engine 3.3

The Analytical Engine

JOURNAL OF THE COMPUTER HISTORY ASSOCIATION OF CALIFORNIA

Volume 3.3

Maiga Wolf, Bookkeeper and Programmer

May 1996

THE MAC AND ME:
15 Years of Life with the Macintosh (Part 2)

by Jef Raskin

ANTI-MICRO ATTITUDES

The computer industry in the middle 70's tended to ignore or minimize the microcomputers that I saw as the future of computing. (The term "personal computer" was to come later.) Nonetheless, I was invited to chair the National Computer Conference session on documentation in 1979 — but this was mostly on the basis of my presence in the large-computer world.

At first, those who asked to exhibit microcomputers were turned down. By 1978 they were given a room in the basement. A few of my friends at the large computer companies asked me why I was throwing away my career by working for a microcomputer company.

At one of the National Computer Conferences I was on a panel where I was expected to uphold the proposition that microcomputers were useful. Many mainframers thought and said that micros were - and would remain - toys. We each gave our little talks, but I didn't score until the discussion session.

To show the superiority of large computers, one of the speakers challenged me to some "benchmarks." The exchange went something like this:

"Anything your little Apple can do, my mainframe can do, and do it better," he boasted. "For one thing, microcomputers don't have the speed of a mainframe!"

"OK," I replied, "name your speed benchmark."

"Invert a 100 by 100 matrix! It will take me about 40 seconds."

"You win," I conceded. "It would take my machine hours to do it."

The audience gave a bit of applause for the mainframe.

Then it was my turn; "We both have to run across the hall. The person getting to the other side first, carrying his computer, wins."

There was laughter as people pictured him trying to pick up and run with his mainframe, larger and heavier than a refrigerator, and then there was a solid round of applause as I raised my Apple II with one hand.

"For my next benchmark, let's discuss power," he said. "Have each of our machines create an index to a thousand page book."

I had to concede. My computer couldn't even hold that much text. This admission got a few guffaws from the audience.

Then I proposed my second benchmark: "You take $100 out of your salary every month and I'll take $100 out of mine. The person who can pay for his computer first wins."

There was a lot of laughter and applause. "But," argued my opponent, "that's not computer power!"

"A computer," I answered, "has no power at all if you can't afford it." From the audience reaction, it was clear I had won the debate.

MACINTOSH PROJECT PRELIMINARIES

Early in 1979, probably in March, I talked with [Mike] Markkula about my idea for a new computer. He had had an idea for a $500 game machine, which he called "Annie."

I thought that a game machine, although a good idea, was not something that I'd feel comfortable doing. So I counter-proposed a general-purpose, low-cost computer based on my own ideas - and dreams - for an interface. Markkula agreed to it.

I picked "Macintosh" as the name for my project, since Macs were my favorite apples. I changed the spelling because I wanted to avoid conflict with the name of an electronics manufacturer - an attempt that proved to be in vain.

Most of all, I didn't want to call the project "Annie," since I felt that the trend in the company to give new products feminine names was sexist — and if you had spoken to the namers you would agree.

Markkula's "Annie" project would, besides games, have allowed the user to program in BASIC. But it was not intended for business, and I thought any new product should be able to handle a much wider range of applications.

I also said that using a TV set or a third-party monitor was playing Russian roulette with one of the most important selling points of a system - how the screen looked.

With these wants and limitations in mind, Markkula sent me off to do design and cost studies. Working with my friends at Apple, notably Brian Howard, I came back with an absolute minimum selling price of $1,000, far from Markkula's goal.

The machine I designed was based on the 6809 chip and had a 256 by 256 bit-mapped screen. I came up with a proportionally-spaced character set that would display 25 lines with an average of over 80 characters per line on the little display. (To put this into perspective, the Apple II displayed only 40 upper-case characters per line. The idea of proportional fonts on a display was then unknown at Apple, though commonplace at PARC.)

My choice of the 6809 was dictated by the tight price constraint imposed initially by Markkula. The better 68000, when it first became available a little later, was $400 - if we bought it in quantity. That would have made the product have an introductory price of about $3000.

My original concept was biased toward the inexpensive and memory-efficient. I noted that a 256 by 256 display could be addressed in exactly two bytes, making fast software easier to write - speed is of the essence in a good interface.

To convey one of the Macintosh design features to others in the company, I built an Apple II with a monitor incorporated into the lid. I used it at lectures and demos and it had great appeal wherever I demonstrated it.

(To this day I don't know why Markkula - to whom I pitched the idea the strongest - Jobs, and all the other people in management didn't use my idea in the II. The Apple II had a pop-off lid, and we could have sold a replacement lid with an angled CRT built in.) My very happy experience with this prototype settled it: the first Mac would have a built-in display.

FRICTION WITH JOBS

While the company was thinking about manufacturing tens of thousands of computers a year (another unheard-of idea), I wrote an internal document called "Computers by the Millions." In it I looked at questions of design, manufacturing, marketing, and general social and economic impact of computers in those quantities. Management found the paper valuable, and would not allow me to publish it for three years, to avoid letting the competition know what we were thinking. It was still years ahead of its time in 1982, when I published it in the ACM's SIGPC Bulletin (Vol. 5 No. 2).

Jobs, unaccountably, did not at all agree with my views of the future, nor with my distributing them internally at Apple, even though I was doing so at Markkula's request. By proposing new strategic ideas and products independently of Jobs, I began to get on his "wrong" side. By this time Jobs had begun to have people who were "in" and those who were "out;" if you were "in," everything you did was golden, if you were "out" everything you did was rotten. By the time Jobs had started NeXT this had become a major trait of his, according to Randall Stross's book, "Steve Jobs & the NeXT Big Thing." My take on this book, and its view of Jobs, appeared in 1994 as "Hubris of a heavyweight" (*IEEE Spectrum,* July 1994, pp. 8-9).

But as I began work on the Mac, I didn't recognize the Jobs phenomenon. Thinking I was still "in," I kept on trying to get Jobs to go see what PARC was doing; since I was actually "out," he resisted the idea strongly.

I , of course, remained oblivious to what was going on. I thought that he would turn around as soon as he saw the quality of what I was doing. Besides, we had been friends, and our disagreements were purely technical.

PASCAL

Early in 1979, I tried very hard to convince the company that we should move away from using BASIC and assembler as our main languages for applications and system software. After presenting the case for and against a number of major computer languages, from FORTRAN to APL, I argued that we should base our work on Pascal. I hired a clever and inventive ex-student of mine, Bill Atkinson, who implemented a Pascal developed under Ken Bowles at UCSD. They had it running on the 6502 processor, the same processor used in the Apple II, and Atkinson suggested porting it to our product.

In the process, Bill had to write graphics routines, an experience that proved extraordinarily valuable for Apple. Many in the company had rejected PASCAL as impossible to put on an Apple II, contradicting several technical memos I had written showing how it could be done.

As Atkinson later said, "We had a bunch of self-trained amateurs who didn't really understand modern software development. The system software team actively resented a new language. Once we had it up enough to demonstrate the word processor, and Markkula saw that, it was clear sailing."

I supported Bill's implementation, and then wrote a PASCAL manual with Brian Howard. Pascal, as I had predicted, allowed us to hire more professional programmers, and later became the main development language for "Lisa" and the Mac. At the time, I personally paid a license fee to UCSD so that Apple could use their Pascal system. Apple never reimbursed me, since Jobs insisted that Apple didn't need and would never use Pascal. Almost all Mac and Lisa software was written in Pascal derived from UCSD. I remain amused by the thought that in some vague sense, it was all owned by me.

THE MAC BECOMES OFFICIAL

By September 1979, Mike Markkula had - over Steve Jobs's objections - approved the Macintosh project. But by going around Jobs I had unknowingly set up a dynamic that made the project far more difficult politically than I could have anticipated.

From the first, Jobs opposed it, calling the Macintosh the "dumbest idea" he'd ever heard of. He would often recite a list of imagined advantages that the Lisa project had over the Mac and put obstacles in the way of my obtaining staff or supplies. His interference eventually became so overt that Mike Scott had me move the entire Mac project to some buildings behind a Texaco gas station across De Anza Boulevard, so that we would be able to develop the Mac in peace. Since we were on the second floor, we called it "Texaco Towers." Later, when Jobs took over the project, he put up a pirate flag and claimed that he moved the Mac out of Apple headquarters so that it would remain pure and uninfluenced by the stodgy company engineers. To me, the pirate flag really indicated a pirate within: as I see it, Jobs took over the project by fiat and lies, and was nearly successful in stealing the credit for having originated it as well.

From the beginning, to keep the project on track, and so that we would not lose good ideas (and the reasons for abandoning others) in the press of development, I created a document numbering system and put the collected documents in the "Book of Macintosh," which grew to some 400 pages. I wrote most of the book, since I liked to write - and was the fastest typist in the group - but the ideas were generated by everybody, and everybody got credit in the text.

Here's one example of the standardized heading format:

MACINTOSH PROJECT DOCUMENT 18 VERSION 0 DATE: 20 OCTOBER 1979 TITLE: DELIMITING STRINGS. AUTHOR: JEF RASKIN.

I asked all the participants to explain the reasons for their conclusions, their right turns and wrong turns, as we went along. I wrote most of the documents late at night at home; we were too busy during the day to get around to it.

I believe Jobs's opposition was partly due to his not understanding what I was trying to accomplish, though at the time I incorrectly thought of him as the supportive friend he had been for so long. For example, when I insisted on bit-mapping and square dots, he would retort that Woz had put a character generator in the Apple II and it didn't have square dots and its sales were paying my salary.

THE FAMOUS ANACHRONISM

It has been often said in the computer and general press that the Mac was a straightforward copy of the work done at PARC. It was not, and the idea does a disservice to the hundreds of people at Apple who developed the hardware, software, marketing, and interface concepts. This erroneous belief turns what was a significant intellectual debt into the appearance of moral bankruptcy. I can't blame people for making the mental leap from hearing that the Mac resembled - and was partially inspired by - the interfaces at PARC to guessing that it was largely "stolen." But at the same time I cannot forgive those who write on the subject (and

make the same claim) for not doing their homework. Most of them never saw or used an Alto, a Dorado, or a Star — the systems developed at PARC. They simply assume that the earlier systems were much the same as today's Macs and Windows machines.

Then there is that apocryphal story about Steve Jobs visiting PARC, having an "Aha!" experience and coming back to Apple in full cry to create the Macintosh project.

Well, he did go, he did see, and he did come back enthused, but the Macintosh project was well under way at that time, having been officially started months earlier. The trip was set up to convince him of the value of the Macintosh project. I'm not sure how the story got reversed, but I later learned that Apple's PR department repeatedly told the false tale to anybody who asked.

Nearly a decade after the introduction of the Macintosh, Xerox took Apple to court over the issue. I was briefly invited to be an expert witness, not by Apple - as I might have expected - but by Xerox. The Xerox attorneys soon learned that the main thing I could testify to was the originality of the work done on the Macintosh. (The Lisa group did do what I consider some shameless copying of the Xerox Star, down to the names for some individual fonts, but that is a different story arising from the fact that a lot of key people on the Lisa project had been hired from Xerox, something that was not true of people in the Mac group.)

The case did give me a chance to use a Star and an original Lisa, each for the first time, an experience that taught me how much further the Mac was from its predecessors than I had remembered.

INTERFACE INNOVATION

One of the substantive differences in the "look and feel" of the Mac interface was the one-button mouse. The one-button paradigm has become so pervasive that many applications for IBM-compatibles ignore the second button that clutters most IBM-compatible mice; the third button that was part of the Engelbart and PARC mice has also disappeared almost completely from popular use. My own difficulties with the three-button mouse - and watching other people have trouble learning it - led me to rethink the design.

With one button, I reasoned, you could not get confused about which to use. It took a while, but I was able to find methods that in every case required the same or fewer operations than those required by the PARC system; it was faster, easier to learn and use, and it was far less "modal."

Of the methods I invented, the most fundamental was the idea of pressing and holding a button while dragging, and using the release of the button to indicate that the operation was complete. This differed from the method — used at PARC and dating back to the work of [Ivan] Sutherland — of click, drag, and click again.

When Larry Tesler came from PARC to join Apple he was naturally resistant to the one-button mouse. Larry was comfortable with the three-button implementation and had long touted its advantages over non-mouse systems. It took considerable effort to convince him, point by point, that my solution was not only workable (which he and others doubted at first) but in fact superior.

In any case, the interface we developed was a distinct and new creation, though it shared many elements with and owed a very real debt to what had been done at PARC. A major part of that debt, of course, is that I was able to use PARC's work as a living demonstration of a highly evolved graphical interface.

The one-button mouse was not the only major difference between the Mac and the systems at PARC. Another interface improvement that made the Mac feel so much easier to use was the way a user selected something or engaged a menu. At PARC, menus were relatively static lists of limited length that the user could summon and dismiss. Bill Atkinson — later to become an Apple Fellow — proposed that we instead extend my method of selection and drawing so that just the title of a menu would be shown, but when you pointed to it, clicked and held down the mouse button, the menu would appear! Then you would release the mouse button when the cursor was pointing at the desired item. This made menus appear when you needed them and disappear without apparent effort. Furthermore, as we both pointed out, having the menus at an edge of the screen and having the cursor position confined at the edge meant that you had to point accurately in only one dimension, which made the menus easier to use. The design of Microsoft's Windows and similar

interfaces does not have this useful "pin to the edge" idea.

Atkinson was led, by analogy with my point and drag methods, to pulldown menus that you can drag across to your desired item. Probably because it worked much as typewriter SHIFT keys do and as a pencil does — you put it down at the beginning of a line and lift it up at the end — my method of using a mouse has prevailed.

I extended this idea to drawing lines and to creating rectangles and other shapes by pointing and sweeping across the diagonal. My "hold and sweep" concept was then applied to making graphical selections. We created a rectangle that surrounds or touches the items to be selected while the button was held. The methods I devised are now so universal that some people who worked on the earlier systems have forgotten how they worked. They tend to "remember" them working as the Mac does now. What I remember is the effort it took to convince my fellow engineers that what I was proposing was better.

I suggested that Apple patent the one-button mouse and the new way of using such a pointing device, but Jobs nixed the idea in favor of patenting Atkinson's pull-down menus. Apple missed this opportunity simply because Jobs didn't want my name to appear on any Apple patents (though I have about a dozen of my own). I was still "out."

A more subtle difference between the Mac and the work at PARC is this: in the Mac you point to something and then tell the system what to do with it. It is the "noun-verb" paradigm that is now nearly universally recognized as desirable by interface designers. As Bill Buxton of PARC has reminded me, the Xerox products used a more complex noun-verb-noun method involving a bunch of function keys (like the current IBM compatibles).

To quote Buxton, "Both the concept and the operation were quite different... it is remarkable how few people who teach and talk about GUIs even seem to understand the differences to even this degree of subtlety." Buxton and his colleagues also published research in the 1990s (on what they term "kinesthetic feedback") that showed why my click-and-drag paradigm worked so well.

Another fundamental part of the Mac from the very beginning was the insistence that unifying software would be built in. Knowing the time constraints of the real world and the inherent laziness of all humans, I suspected that if we built in an interface, programmers writing applications would use it, grudgingly, for their first mock-up as it was much faster and easier than writing the interface themselves - standard practice in all products prior to the Mac.

I knew that writing a rule book would only antagonize the independent spirit of software developers, who are inherently entrepreneurial. They had to be tricked into using the Mac interface. I could depend on their time constraints, and the likelihood that our interface would be far superior to what they planned, to insure that the details enforced in the software prototype would appear in the final product.

It worked. When the Macintosh was released, users found that learning new applications on a Mac required far less effort than the same task on any competing system. This gave third-party software developers added incentive to do things in the Macintosh manner, and Mac users have reaped the benefits.

The success of the Mac led other companies to copy its interface, and one can now move without too much difficulty from the Mac to Windows, to Geoworks, to most workstations, and even to some mainframe front-ends without retraining and with barely a glance at a manual or help screens.

My unifying software originally was to be a graphics-and-text editor within which applications could run as additional commands (via menus), all input and output being through the interface designed for the editor. Later, the PARC desktop metaphor was adopted from the Lisa group, who had adapted it from the Xerox Alto and Star computers. The incredible work of the Mac software team designed and squeezed the necessary code into a "Toolbox" within a relatively small ROM (Read Only Memory) that we could afford to put into the product.

The interface concepts I wanted to implement required fundamental hardware changes. One example was the way the electronics of keyboards were designed, not in keyboard layout - which obviously affects the interface - but in the way the keyboard works at the chip level. Before the Mac, and excepting PARC which was at that time not a commercial manufacturer, the makers of commercial keyboards built each key to put out a signal

when pressed. By the middle 1970's a special "encoder" chip took a signal from a key and produced the code for the symbol that key represented. There were usually a few exceptions: the SHIFT key could be pressed and held and while other actions took place; the same was often true of other state-shifters such as the "control" key.

But these exceptions were built into the encoder chip; what I wanted was a keyboard whose keystates — whether any keys were up or down — would be "known" by the computer. By analogy with pianos and organs, which can use any combination of keys simultaneously to play what musicians call a *chord,* this was known as a chord keyboard. I had long believed that this was an essential step toward improved interfaces and when I first went to PARC I was delighted to learn that they had come to the same conclusion.

Burrell Smith, our hardware designer, participated avidly in these discussions, and often suggested ways in which hardware changes could help the interface, sometimes also proposing changes in software design that could simplify hardware requirements. In each case the interface requirements took precedence, but this was probably the first time a commercially successful computer was designed with hardware and software subservient to the issue of usability.

The Mac succeeded because the initial impetus for its creation came from a humanitarian impulse, rather than a hardware dream or a marketing study.

SELLING JOBS ON THE IDEAS

A popular description of Jobs is that he has a "reality distortion field." This phrase accurately described Jobs's ability to convince people that whatever he was saying at the time was inevitable. I'd seen him charm otherwise reasonable people into believing absolute nonsense.

Some of this is helpful when doing something new in the world, but - as I see it - Jobs lived at the center of this field and actually believed and acted not only on vision, but on the basis of his own falsehoods, sometimes with unpleasant consequences.

It seems to me that Steve Jobs was also mesmerized by the power of part of his key insight that had helped make the Apple II a success: 'make it look attractive' became a guiding principle. He continued to confuse appearances and quality ever after; years later at NeXT, his first major expenditure was to hire decorators for the new office complex. This passion for appearances would have been an asset, or at worst immaterial, in someone who also understood the products; but Jobs often did not.

THE PASCAL POSTER

In 1979 he botched the design of a poster that summarized the structure of the Pascal programming language. Programmers found sets of diagrams created by the originator of the language, Niklaus Wirth, a handy reference. In writing the Pascal manual I had discovered several errors in Wirth's diagrams and also disclosed some simplifications. Diagrams in the manual reflected these corrections and improvements. I thought that it would be good advertising, as well as a real benefit to programmers, to put the entire set into a decorative poster. Color would serve as a key to link items of the same syntactic type, making relationships among language elements clearer.

Jobs thought it was a great idea, and promptly hired a prominent graphic artist, Kamifuji, to produce the poster. Jobs asked me for a copy of my diagrams so that the artist could estimate the project, telling me that once we had a quote I would work with the artist. But the next thing I knew, Jobs proudly came into my office with the finished work. Thousands had been printed.

The poster was very good looking, with bold colors on a jet-black background. But some of the diagrams were no longer correct and the colors had been chosen purely for esthetic effect, making the chart unnecessarily hard to use. I told Jobs that it was very pretty but wrong; he didn't care, and blissfully went on to something else. The posters were sent to stores as advertising posters, but they couldn't be shipped with the Pascal product as planned. It was a waste of time and money. This example is not significant in the history of Apple *per se,* but does say a lot about how Jobs thought.

MAC POLITICS

By the end of 1979 it was clear to many people that unless Jobs had a better understanding of what was being attempted on both Lisa and the Mac, he

would continue to inadvertently sabotage the former and be antagonistic to the latter.

My friend Bill Atkinson knew a great deal about what was going on with the Macintosh even though Jobs had officially forbidden him to work with members of the project. This meant that Bill had to keep his involvement with the Mac secret, lest he lose his "in" status, while he worked on the Lisa. At the time he was writing his meticulously crafted QuickDraw graphics system - then called LisaGraph - for Lisa (Apple knowingly used the name of the list-structured graphics system I designed for this central piece of software - without permission or compensation).

With Bill's connivance and the help of Tom Whitney — who had given me the title of "Manager of Advanced Systems" to correspond with my work on the Mac — and by keeping my name out of the picture, we at last managed to convince Jobs to visit PARC.

Jobs later said that after he went to PARC, he returned inspired, and launched the Lisa and then the Macintosh. This story, once promulgated by Apple's PR department and often repeated in books, articles, and even by the generally excellent PBS series on the history of the computer, is inaccurate, to say the least. As Atkinson put it in a phone call to me, "You were instrumental in getting Jobs to go to PARC, and that was central to getting his support for new interfaces." Jobs pointedly did not invite me on this visit, and excluded me from the conversations about it when he got back; it would have been very hard for him to have admitted that I had been right about the value of the work done at Xerox.

In general, I remained oblivious to the politics going on at Apple, and concentrated on the design of the Macintosh. This left almost no room in my life for anything else, except practicing the piano and occasionally getting out to fly a model plane. I bought a house a few blocks from Apple so I could bicycle in and back on a moment's notice. The Macintosh project was my life.

CONCEPT AND COSTING

What was the Mac concept like in the early days? We researched many possibilities. For example, we considered a bit-mapped LCD display which had a resolution of 256 X 26 (yes, twenty-six) and a cost to us of about $240. At our usual five-to-one ratio of parts cost to list price, that part alone would have been $1200 at retail. A 256 X 256 or larger display with any technology other than the cathode ray tube (CRT) was then totally out of the question, since a CRT display cost between $35 and $50.

A drawing done by Brian Howard in 1980 shows a one-piece box with a built-in CRT, 5 1/4" drive, keyboard, and joystick. The joystick is in the same position occupied by the trackball in the later Mac Portable.

We also worked on a strain-gauge stick almost identical to the current IBM graphic input device. Embedded pointing devices have a long history at Apple; for example, in 1978 Woz came with the idea building a pair of orthogonal thumb wheels (one each for vertical and horizontal motion) under the Apple II keyboard. This was a response to my request that we build a pointing device into the box that could be operated without removing the hands from the keys. This seemingly obvious good idea reached fruition years later with the PowerBook series and was probably reinvented independently by the PowerBook group.

Graphic input was an essential element of the Macintosh from the first. I thought that the mouse in particular was a clumsy way of going about it - for one thing, it takes up too much desk space, and for another you have to find it anew each time you want to use it.

But Jobs was an adamant mouse-ist, (mainly, I think, because that's what PARC had), and until third party vendors supplied trackballs, the mouse was the only graphic input device available for the Mac.

TEAM BUILDING AND THE TOOLKIT

One of my basic concepts was of a software nucleus that would be built into ROM, and serve as a home port to the user, tossed about on the high and varied seas of application software. To write the software, I hired Bud Tribble, who had similar thoughts. He and the two other "B's," Brian Howard and Burrell Smith, were the first Macintosh team.

The Mac Toolkit was initially written by Tribble, who was in charge of Mac software; it was taken over by unstoppably hard-working UC computer

science dropout Andy Hertzfeld, Bill Atkinson, Bruce Horn (who, at fourteen, had been one of the usability testers of Smalltalk at PARC), and others.

Each member of the original "gang of four" came to the group through a different route. Bud Tribble was a medical student, a programmer and designer of genius who I had known at UCSD; he and Bill Atkinson had been good friends there. Atkinson pointed out the talents of a man working in repair, Burrell Smith, and after interviewing Smith, I hired him as head of hardware design. Brian Howard had been a friend of mine for years.

There were established hardware designers that I had tried to bring over to the Mac (and who wanted to work with me), but Jobs had forbidden them to join the project. Still, Smith proved a first-rate designer who was open to thinking from a software and human-interface point of view, and he was a delight to work with.

I brought MIT anthropology student Joanna Hoffman on as our marketing person. Her major contribution to the Mac was to make sure that design decisions didn't preclude international sales; this concern was unusual in the then-parochial microcomputer industry. Thus the Mac had, from the first, the accents, special characters, and diacritical marks needed in languages other than English. We had come a long way from the philosophy that upper-case letters were all you needed. (Joanna also introduced me to my future wife.)

Steve Clark (another UCSD student I brought to Apple, and whose Olympic-level-kayaking sister Candi was later to marry Woz - as I've said, it's a small valley), and a few others formed the nucleus of a team easily the equal of the much larger and better-funded Lisa group. I hired some, such as programmers (and musicians) Gareth Loy and Bill Schottstaedt, from SAIL; another, the remarkable poet Bana Witt, had been a music student of mine when I taught at the San Francisco Community Music Center. She later married Bruce Tognazzini, another Apple employee who worked with me and was to write and lecture extensively about interface design. (Being a minister, I had the pleasure of conducting their nuptials.)

Donald Reed, the very image of a bookish intellectual in appearance and manner, worked with me closely on documentation. Of course we enjoyed the under-the-table help of Atkinson and others on the Lisa team who believed in what I was trying to do, and the warm support of the late Tom Whitney, who had been hired to head engineering for all of Apple.

THE END OF THE BEGINNING

John Couch, a good manager and insightful computer scientist who was running the Lisa project, increasingly found Jobs a nuisance, and eventually managed to get him removed from the Lisa project. Jobs, at loose ends and hearing rave reports about the Macintosh, decided to have a hand in it. Apple's top management helped shunt him to the Mac project to get him away from Lisa, which was seen as the company's hope for profitability in the 1980's.

Jobs's attempts to undermine the Mac project now took the form of destroying my credibility. One of the more blatant incidents was the "brown bag" lunch at which I was to describe the Macintosh project to the company at large. It is discussed in a confidential memo that I wrote to Mike Markkula to explain why, though I was seeking someone to manage the Macintosh project so I could concentrate on technical issues, I didn't want Jobs to be in charge. The memo specified, in detail and in my judgment, Jobs's many and egregious failings as a manager.

I had asked that the memo be kept secret, and Markkula agreed, though he said that he didn't think he could do anything to control Jobs. I believed this assurance and, thus, felt betrayed a few days later when Jobs called me in to his office to "discuss" the memo. I dimly recall Markkula saying something about having had to discuss it with Jobs. But Apple was a very open company, doors were left unlocked, and people wandered freely into one another's offices. Any of a number of people might have seen the memo and made a copy for Jobs, or he may have noticed it himself. Markkula thinks that something of this nature is what must have happened, and it might well be.

The memo reflected the running joke that the way to get Jobs to agree to something was to tell him about it, let him reject it, and wait a week; when he came running to tell you about "his new" idea, you'd exclaim, "Great, Steve, we'll do it right away!" In the memo I also made a prediction that was to prove exact: "Jobs was wrong on his Apple

III schedule, wrong on the Lisa schedule, wrong on the cost and price estimates, and he will be wrong on Macintosh. He is a prime example of a manager who takes the credit for his optimistic schedules and then blames the workers when deadlines are not met."

The memo also related the incredible brown bag incident: "Jobs is often irresponsible and inconsiderate. An example is the brown bag seminar I was scheduled to give on 17 February. In January, he first cancelled the seminar, but then he agreed that I was to give it. Two hours before the talk he called me to say that he was canceling it again. His reason was: I cancelled it because of the reorganization in PCs.' However, Jobs did not tell the seminar's organizer about the cancellation, nor did he place any notices announcing the cancellation."

"At noon, fortunately, I made a last-minute decision to go over to the seminar site, where I discovered a crowd of over 100 employees waiting to hear the seminar. I announced the cancellation myself - and then I gave a talk on my current work and interests at Apple, instead."

I was careful not to mention Macintosh or give specifics since Jobs had forbidden it, but just explained the cognitive aspects of the interface and design principles my group and I had developed; it was - as everybody knew - a veiled introduction to the Macintosh project.

The talk was received very enthusiastically. The morning after the seminar Jobs called me into his office and told me that I had violated his explicit instructions and was fired. Ignoring what he said, since he often spoke without thinking things through, I told him that I'd come back in the afternoon, after I'd completed something I was working on, and we'd discuss the matter.

Later I went back and, as I expected, he had decided not to fire me but I was "given" an extended paid leave from Apple. The leave turned out to be a very important time in my life. For one thing, I went to a party at marketer Joanna Hoffman's house where I met my future wife, Linda Blum. I found a place to live on ten acres of land high in the foothills of the Santa Cruz mountains, offering a magnificent view of Silicon Valley. I rebuilt the dilapidated old house that stood there, adding a large music room for my piano and a small flying field for my model planes.

When I came back from my leave I was offered the position of head of Apple's research division. I had been offered this before, had accepted, hired a good group, and seen them whisked away to "put out fires." In those days Apple didn't know what research meant, and looked at the talented people I hired as resources wasted if they weren't working on current products. Besides, there was the matter of personal integrity. Steve Jobs had become impossible for me to work with.

Most people worked around him or sucked up to him or were in awe of him. In fact he was no genius; he resembled a planet shining by reflecting the light of others. Yet he thought of himself as the Sun King. He could not abide someone who was unimpressed by Steve Jobs, yet by his actions he had lost my respect, and I am incapable of being a sycophant.

Steve had *chutzpah* in the extreme; he said that the Mac would make "a dent in the universe," without the least idea how big the universe is, or how little a dent all our activities really make. And you can also explain Jobs with another Yiddish word, *mensch*. It is high praise to say of a person that he (or, in these enlightened days, she) is a mensch or "a real mensch." A mensch is cultivated without losing the common touch, upholds high principles while remaining practical, is kind and generous without short-changing himself, and is attentive to his responsibilities to himself, his family, his business, his associates, his community, and the world. If you understand the qualities that make a man a mensch, then you understand a lot about Steve Jobs. Everything a mensch is, he isn't.

At this point the only alternatives left to me were to leave or learn to toady to Jobs. I resigned from Apple and gradually watched my predictions about the Mac come true. Jobs took until 1984 to get the project out. Burrell Smith quipped that it was in "constant time to completion mode" and I was repeatedly told that Jobs did exactly what he said I would do, make endless mindless changes; I have many faults, but lack of direction is not one of them.

Steve was given to imposing absurd requirements on the project's designers. This was especially ironic since one of the arguments he used, to convince management that he should be given the Mac project, was that I was an "academic dreamer and a perfectionist" who would keep on changing his

mind and never bring the project to fruition on time. My detailed schedule showed release of the product toward the end of 1982. Jobs's verbal plan was something like six to eight months shorter.

There is some evidence to back up my perspective: the next project of comparable scope that I managed (a workstation for Canon) was completed on budget and on schedule. The next project Jobs tried to manage (the NeXT computer) was a disaster in both these regards.

The resultant Mac not only took more time to come to market, but was a less coherent and — in some ways — less capable product than what I had been working towards. The interface was less consistent and harder to use, and there was no way to get at the hardware bus. Other parts of the Mac design were improved. Whether my version (as it would have matured as it approached production) would have been more commercially successful than the 128K Mac is an unanswerable question. The experiment cannot be done, and we will never know. I feel it would have been a somewhat better product that would have penetrated the market faster. I would guess that Jobs would disagree.

DREAM FULFILLED... ALMOST

Three decades ago I dreamed of a computer with which I could compose music and print it out in full musical notation, write properly formatted text in a panoply of fonts, have the ability to mix text and graphics, and do drawings with precision and ease.

Today I do all this and more at the tiny and capable Macintosh PowerBook that sits at my desk. It goes wherever I do. This much of the dream came true.

My reasons for deciding to abandon teaching for commerce proved correct. The Macintosh's profitability (as contrasted to Xerox's extensive published record) convinced companies such as Microsoft and IBM that the interface was the controlling element in most sales. Now a vast majority of the computers sold have an interface that looks much like the Mac's.

Because of the Macintosh project, computing has been made easier and more pleasant for hundreds of millions of people years before it might have happened otherwise. I made not a penny for my work on the Mac, beyond my salary at the time; but I helped change the world in accord with my own personal vision, and I have seen the effect of this in my own lifetime. This would be fully satisfying if I didn't know so well that we can do much better.

The desktop metaphor, used by everybody from PARC though Apple and Microsoft and extended almost absurdly by Apple spin-off General Magic, was a clever way of making the workings of an operating system palatable and learnable. It is far more fundamentally good to eliminate the need for an operating system altogether. The current paradigm of using application programs is inherently wrong from the standpoint of interface design. This is widely recognized, but the solution offered is to make them interoperable, which solves some of the problems but by no means all.

GUIs as presently designed and used are an interface dead end. They can be patched endlessly, but only a completely different approach can bring a large jump in usability. The Cat computer, which I developed for Canon, demonstrated that my alternate approach is implementable and both more productive and more pleasant than GUIs. Canon failed to market the product effectively, possibly because the moribund Electronic Typewriter Division had been selected for the task, and it is now a dead Cat.

The parts of computer interface design that I am working on now are not dependent on particular technologies; any advance in the basics of interface design will apply — however more powerful computers become, however broad the information networks of the future spread, and however the technology is melded into our everyday or even everymoment lives.

A CRITIQUE OF SOME HISTORIES

Many friends have suggested that I counter the numerous incorrect accounts of the history of the Macintosh with a true one of my own. It is difficult, for even though nobody was or could have been more closely involved with the initial creation of the Macintosh than I, I cannot eliminate the colors that tinge my memories. It was a very emotional time, full of strong feelings, massive egos in conflict, distinctive personalities, and many rights and wrongs. But I cannot do worse than some of what has been published.

There is a strange avoidance of scholarly seeking after truth. An egregious example of the anti-academic attitude occurs in a book by Robert Cringely, who writes a delightful column that appears weekly in *InfoWorld*. In his book he has the Mac and Lisa projects being created by Steve Jobs after Jobs made a visit to PARC in 1980 and came back inspired.

I wrote to Cringely and pointed out that his account - like those of several other authors - was wrong; Jobs had indeed made the visit in 1980 (some say in December of 1979,) but the Mac project was proposed in the spring and officially started in September of 1979. In other words, the project was well underway before the supposedly pivotal event took place.

Cringely was unabashed. He wrote back: "As for all the business of what project started when, whether Lisa started before or after Steve visited PARC, whether the Mac had already begun or not, well I don't think that it really matters very much. My attempt was to EXPLAIN (I say that at the front of the book), not to be a historian." How one can hope to explain what happened, without even knowing what happened, eludes me.

A PBS special on the history of computers made the same mistake of attributing the genesis of the Mac to Jobs's visit to PARC. When I wrote to Jon Palfreman, its producer at WGBH, he replied, "The part of the program you are referring to comes at the end of a lengthy segment about the highly innovative work done at Xerox PARC. This section was based on extensive interviews with Alan Kay, Bob Taylor and Larry Tesler. The purpose was to show that the key concepts of interface design which today are a feature of most PC's (if you count Windows) were first discussed at Xerox PARC. When those ideas were embodied in an affordable machine — the Macintosh — they began to change the world of personal computing. I was aware of your key role in the Macintosh project, and indeed of the contribution of people who developed Lisa. My aim in this particular program wasn't to detail the history of Apple, but to show how the key interface ideas found their way into consumer PCs."

Again the false scenario seems so plausible and story-like that the person in charge does not care to "detail the history." But it is in that history, and not only the history of PARC, that "the interface ideas found their way into consumer PCs." The people he interviewed were at PARC; their association with Apple began only after the Mac was well under way. Thus they could only tell him about the development of the ideas at PARC and, in the case of Larry Tesler, about the work on Lisa only after 1980 — that is after Apple was committed to the basic direction I wanted the company to take.

Larry was quite resistant to some of the non-PARC ideas that we had developed independently. He did not at first understand many of the improvements over Xerox's work — such as the one-button mouse — that I created. He did not work on the early Macintosh project at all but on Lisa, which was modeled closely on the Xerox Star, even to the point of having the same Xerox-created names for the fonts. The Macintosh proceeded for years much more independently and (significantly for the reports that used them as sources) out of the view of the people interviewed!

The years of thinking and experimentation on the early Macintosh project have gone unreported, even though the early work led to the breakthroughs that made the Macintosh and everything after so much of an improvement over what went before. Against this reality we have the powerful mythological image of Jobs going to PARC, having an "Aha!" experience and coming back at full cry to Apple to create a fantastic project.

The fabricated Jobs story is familiar — it parallels that of Archimedes jumping naked out of his bath crying "Eureka!," and a dozen other stories. That there was a little-known computer scientist who had been working on the concept for over a decade — who created the project, and then maneuvered Jobs to go to PARC, so that Jobs would begin to understand (and thus support) what was already going on at Apple — is a very different, more complex, and unlikely-sounding story.

Also, the appealing and basically true legend of two college drop-outs who created the profitable and excellent Apple II blends in easily with the fiction that one of the dropouts went on to create the even more revolutionary Macintosh. It is a less striking tale that a former college professor and computer center director with a degree in computer science instigate such a thing — but although it may not be as good a story, it is what happened.

Cringely and Palfreman were not being underhanded, only a bit careless and — in Cringely's case - cavalier. In some other cases, authors drew the wrong conclusion by lacking accurate information. Jeffrey Young, in his book "Steve Jobs," writes of the first time that Jobs (along with Atkinson and others) saw the work at PARC.

"Atkinson and the others were asking Tesler questions, one after the other." "What impressed me was that their questions were better than any I had heard in the seven years I had been at Xerox... Their questions showed that they understood the implications and the subtleties..." But Young did not ask why their level of instantaneous understanding was so impressive. The reason was that I had been explaining all this stuff to Atkinson and Jobs for years; Atkinson (a student of mine who had worked with me extensively prior to this meeting) had grasped it very well. Tesler didn't know about this background, wasn't told, and so was bowled over.

Atkinson couldn't very well say that Raskin had briefed him and some others, because I was out of favor with Jobs at the time, and anything I proposed was automatically rejected. Only after much planning and sleight of hand, during which it appeared that Atkinson supported the PARC trip and I opposed it, did Jobs agree to go.

There is also the halo effect. During the years that Steve Jobs was on top at Apple - and before NeXT showed his fundamental weakness - he was usually credited with inventing the Macintosh. Later, when his star was declining — as his company, NeXT, beat one strategic retreat after another, and as General Magic, co-founded by Bill Atkinson, Andy Hertzfeld, and Marc Porat, was about to announce its first product — the December 27, 1993 issue of *InfoWorld* included a story hailing Bill Atkinson and Andy Hertzfeld as the creators of the original Macintosh.

Their contributions were essential to the product and represent some brilliant work, but neither of them has ever claimed that they created the Macintosh. Again we find the heroes of today falsely credited with the achievements of others not currently in the limelight.

Steven Levy's history of the Macintosh, *Insanely Great* — published to ride the wave of publicity for the 10th anniversary of the Mac — is also occasionally at odds with historical fact. Levy retells the Jobs-at-PARC story. Strangely, he credits me with having paintings shown at a famous museum; in fact I have never done any paintings. (An adaptation of this book, published in the February 1994 issue of *Popular Science,* tells a story that is far more accurate, — although it still calls me a painter.)

John Sculley, in his ghost-written book *Odyssey,* refers to me as a "programmer" at Apple. I was never a programmer at Apple, and the rest of what he says is nearly as inaccurate. He got his misinformation about the history of the Mac primarily from Jobs, with whom he spent a lot of time. Like Cringely, Levy, and Palfreman, he chose never to interview me or even call me — or others who were there — to check on his facts.

My experience with Jeffrey Young was especially disturbing. I had agreed to the interview with the understanding that I would see and comment on the galleys before publication; he never sent them - his inaccuracies compounded by a breach of trust. By way of contrast, Owen Linzmayer's *The Mac Bathroom Reader* is more accurate; for example, it gets the order of events straight.

In any case, Jobs's own view of how things came about necessarily must have been distorted. For one thing, we often misattributed ideas deliberately when speaking to Jobs. If the group admired an idea by someone Jobs didn't like at the moment, we gave the credit to someone currently on Jobs's "good" list. It was also often necessary to use "reverse psychology" on Jobs; we got a lot of features into the Mac by having someone (usually me) suggest the opposite. Jobs would then see the problem in "my" approach and often tell us to turn it around.

Another technique was to tell him about something informally. Often he replied that the idea was dreadful. Then when he "proposed" that same idea after its merits had settled in, a few days or weeks later, we'd tell him he was a genius for having thought of it.

Thus, Jobs's recollections of the history of the Mac would often be far from what actually went on. Being very independent, I was often on Jobs's "bad person" list, so I had to rely heavily on these techniques. You'd think he would have caught on when one of "his" ideas turned up implemented later that afternoon or the next day, but he simply

believed that his great engineers — and his way of driving them — could get it done so quickly. He had little intellectual basis on which to judge the difficulty of software or hardware tasks, which often helped us pull the wool over his eyes. How it all looked to him I cannot say. Eventually his impossible management style became so well known that Sculley and the board of directors of Apple had to remove him from all functional duties in the company. My memo had finally been acted upon.

JOBS AND THE PIPE ORGAN

Here's another story that doesn't quite fit in anywhere, but gives some insights into the interpersonal dynamics of the time. When I first started working at Apple, Jobs and I would take long walks (probably like the much-reported walks he would later take with John Sculley). I remember giving him mini-lectures on the philosophy of science or the performance practices of early music. On one of these walks I shared with him my lifelong ambition to own a pipe organ. I explained that the valves to the pipes in many organs were driven electrically, and I hoped to hook up an Apple II to one which would turn it into a modern player organ. (I published an article with details in *Byte*.) He asked me why I didn't have one; I told him that it was mainly a matter of space, and that there was a secondary consideration of cost.

Jobs had a suggestion: if I could find an organ I could afford, I should buy it and Apple would let me put it up in the lobby of the new, large building on Bandley Drive, in Cupertino CA. I would hook it up to an Apple II, which would play it for visitors. After hours, I could practice on it, or even give company concerts.

Jobs was excited about the idea and told many people about the organ that was going to be installed. With this encouragement, I searched for an organ in earnest. In a few months I got lucky and found an abbey where the organ was being replaced. I told Jobs the good news and he congratulated me on the find. I purchased their old organ, had it crated and moved onto the abbey lawn (no small task in itself) and called Jobs to tell him that the crates with the organ would be there in a day or two.

"What organ?" he asked. "The pipe organ we're putting up in the lobby," I replied, thinking that he must be distracted to have forgotten. He first said that he had changed his mind, because it looked like space would soon be tight; when I suggested that it was a bit late to change his mind, since I had already purchased the organ, he retorted that he had never agreed to have the organ installed at Apple in the first place.

When I got back I reminded him that he had made a commitment, and that I had gone to some trouble and expense based on his assurances. He told me that he had never assured me that he would give me room for the organ, and refused to speak in my presence to the people who had heard his promises. I asked if, until the issue was resolved, I could store the crates in the still-empty buildings. The crates were outdoors, this was an imposition on the abbey, and if it rained, the organ could be seriously damaged. He simply said no.

I was stuck. The organ was too large to fit into my house or even a rental storage unit. I called every organ builder, organ teacher, and church I could find and, after much desperate work, found a church in Santa Clara that needed an organ. They, in turn, found a benefactor to purchase the organ from me for them. After long negotiations I succeeded in selling it for a fraction of its value.

This was my introduction to the new Steve Jobs, or perhaps a phase of the old Jobs I hadn't yet seen. Apple's first employee and his friend of many years, Dan Kottke, who had traveled with him in India and worked with him day and night to help Apple get started, was treated even more shabbily. In late 1980 Dan was surprised to find out that — in spite of their long friendship and the many uncompensated hours he had put in — he was not going to get any stock options. Later Woz gave some stock to Kottke, and to some other deserving people from the early days of Apple (such as Bill Fernandez, Chris Espinosa, Randy Wigginton, Cliff Huston and Dick Huston) all of whom Jobs had turned his back on. Woz's was an admirable act of pure generosity.

As for me, I still don't have a pipe organ.

This is a preliminary version of a portion of a book in progress. Comments and corrections are welcomed. Please send them to jefraskin@aol.com.