Apple mouse
2002

Photo
Courtesy of Apple

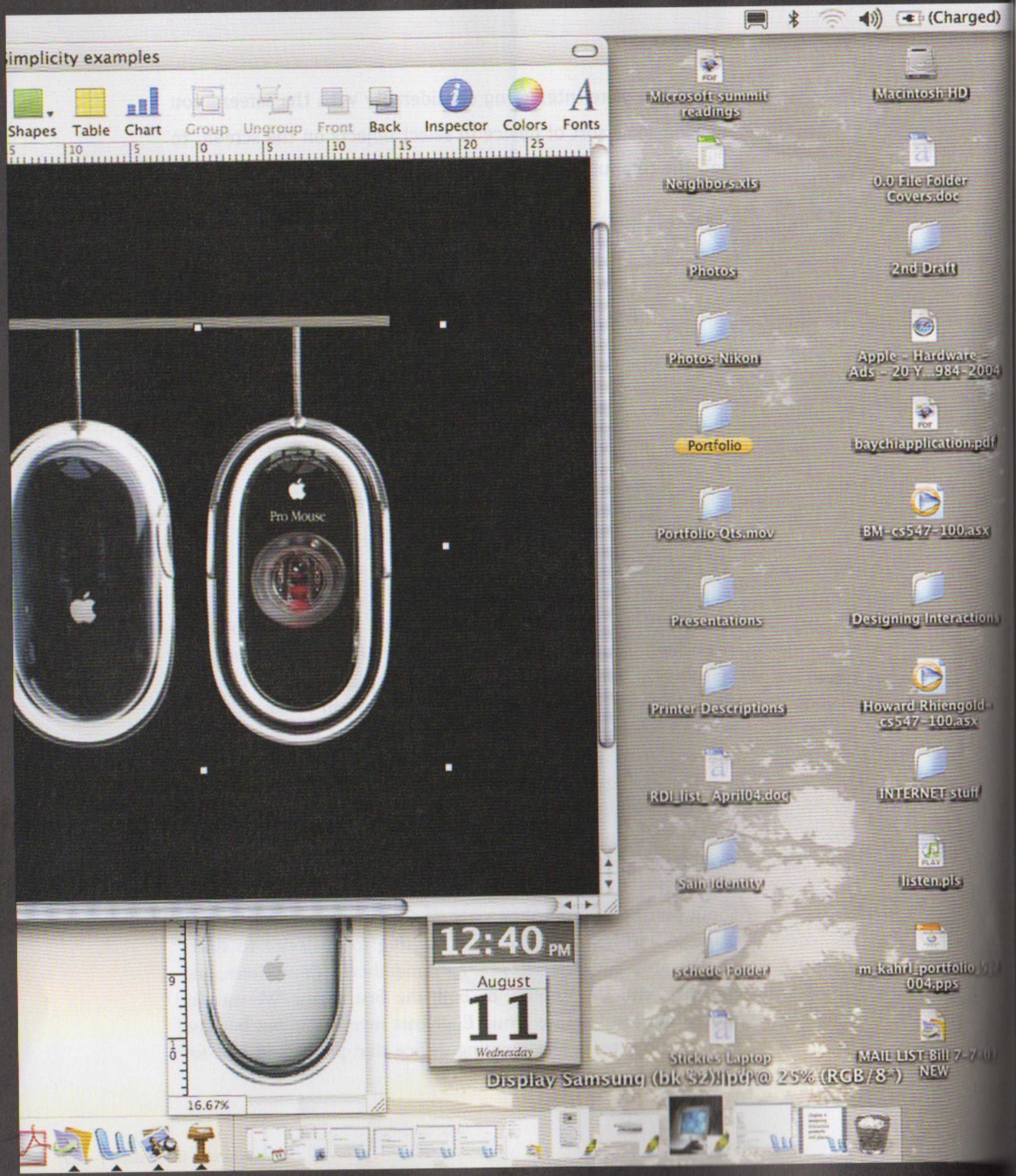
When you were interacting considerably with the screen, you needed some sort of device to select objects on the screen, to tell the computer that you wanted to do something with them.

Douglas C. Engelbart, 2003, referring to 1964

Why a Mouse?

Who would choose to point, steer, and draw with a blob of plastic as big and clumsy as a bar of soap? We spent all those years learning to write and draw with pencils, pens, and brushes. Sharpen the pencil to a fine point and you can create an image with the most delicate shapes and write in the tiniest letters; it's not so easy to do that with a mouse.

Doug Engelbart[1] tells the story of how he invented the mouse. When he was a student, he was measuring the area under some complex-shaped curves, using a device with wheels that would roll in one direction and slide sideways in the axis at ninety degrees. He was bored at a conference, and wrote in his notebook about putting two wheels at right angles to track movement on a plane. Years later, when he was searching for a device to select objects on a computer screen, he remembered those notes, and together with Bill English, he built the first mouse. We use the mouse not just because Doug Engelbart invented it, but because it turned out to be the pointing device that performed best for



Apple Mac OSX desktop with images of Apple mouse

Photo Author's screen capture

pointing and clicking on a display, outperforming light pens, cursor keys, joysticks, trackballs, and everything else that was tried in early tests with users. The mouse won because it was the easiest to use.

We understand the reasons for the triumph of the mouse much more clearly from the story of developing the early designs told by Stu Card,[2] who joined Xerox Palo Alto Research Center (PARC) in 1974 and has spent much of his time there perfecting scientific methods to integrate with creative design. He has developed a process to predict the behavior of a proposed design, using task analysis, approximation, and calculation. His idea is to accelerate the movement through the design space by a partnership between designers and scientists, by providing a science that supports design. He tells the story of applying this science to the development of the mouse.

Why a Desktop?

It seems surprising to find a "desktop" on the spherical surface of a glowing glass display, housed in a bulky plastic box that in itself takes up half your desk. Or is it on the cramped flat screen of your laptop? Who came up with that idea? What were they thinking about, and why did they choose to design a desktop rather than a floor, or a playing field, or a meadow, or a river? Why does this desktop have windows in it? You usually think of windows being on the wall, not all over the surface of your desk. Why does it have a trashcan on it? It would seem more natural to put the trashcan on the floor.

In 1974 Tim Mott[3] was an outsider at Xerox PARC, working for a Xerox subsidiary on the design of a publishing system. He describes how the idea of a desktop came to him as part of an "office schematic" that would allow people to manipulate entire documents, grabbing them with a mouse and moving them around a representation of an office on the screen. They could drop them into a file cabinet or trashcan, or onto a printer. One of the objects in the office was a desktop, with a calendar and clock on it, plus in- and out-baskets for electronic mail.

There were lots of other people at Xerox PARC at that time thinking about desktops and other metaphors for use in the design of graphical user interfaces (GUIs), but Tim was working most closely with Larry Tesler,[4] and the two of them worked out processes for understanding users by talking to them, using guided fantasies, participatory design, and usability testing. Larry describes how he developed these processes and how icons arrived on the desktop. Larry insisted on simplicity and designed interactions that were easy to learn as well as easy to use. He went on to Apple and formed another partnership in the development of the desktop, working with Bill Atkinson[5] to create the designs for Lisa, including the pull-down menus, dialog boxes, and the one-button mouse. These ideas stayed in place as the user's conceptual model for the Macintosh and all of the GUIs that followed, stretching the desktop metaphor almost beyond the breaking point.

NLS, Alto, and Star

When Doug Engelbart invented the mouse, he arrived at the dominant design for input devices in a single leap from the light pen, and the development of the mouse since has been more in the nature of evolution than revolution.[6] Engelbart also invented the point-and-click text editor for the NLS system (oNLine System) that he developed at the Stanford Research Institute (SRI), and that system migrated with members of his design team to the fledgling Xerox PARC and became the foundation of the Alto, the first computer with a GUI. In the versions of NLS that were built at PARC for the Alto, the text-editing demonstrations were impressively fast, with a clattering of keystrokes that sounded businesslike and productive. Direct manipulation made it so easy to pick things up and move them that people would often find an instance of a word they wanted in the text and move it into place, instead of typing it. For programmers, this was a wonderful interaction, but the patience needed to acquire the skills proved a

fatal barrier for novice consumers when computers became accessible to ordinary people. It took the influence of Larry Tesler and Tim Mott to create a text editor and page layout design system that was really easy to use, and this was based on rigorous user testing and rapid iterative prototyping. They came close to the desktop metaphor that survives today.

One of the major innovations of the Alto was the bitmap display, making it easy to show graphics. You could make any dot be black or white, allowing you to put any image, whether a font or a picture, onto the whole screen. The memory to do that for a full-page display was exorbitantly expensive at the time, but it meant that you could translate the tradition of graphic and typographic design to the computer. Earlier computers had primitive and pixelated type fonts, but the bitmap display meant that what was on the display of the computer could look exactly like a book, except that it was only 72 pixels per inch instead of about three times as high a resolution for the printed page. A white background and dark text was used on the screen, so that the displayed image was like the printed result, and the screen size was chosen to be exactly the same as a page of text, enabling the concept of WYSIWYG (What You See Is What You Get).

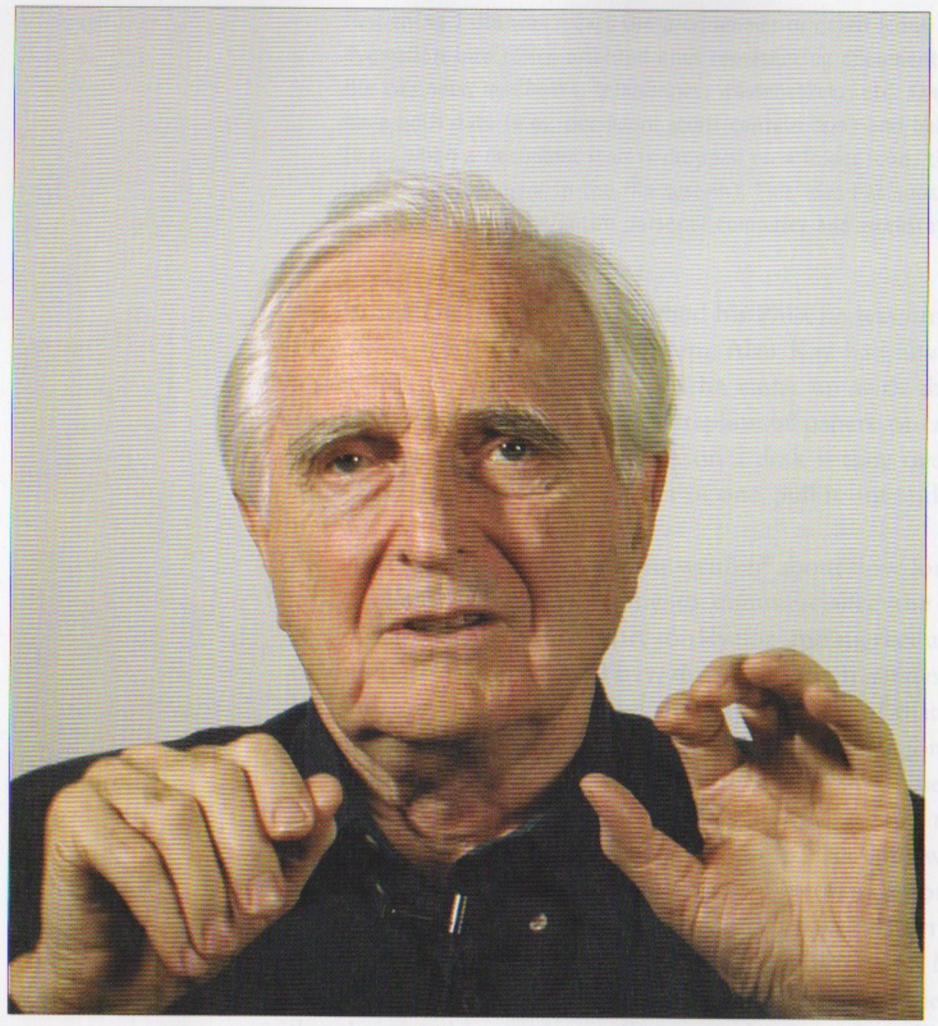
The idea of the desktop was floating around Xerox PARC as part of the communal consciousness. David Canfield Smith had devised an office metaphor as part of his PhD thesis, "Pygmalion: A Creative Programming Environment," published in 1975. His icons looked like mathematical symbols and boxes that you could type into but defined the characteristics of the icons that have become commonplace. Here is his explanation:

> The entities with which one programs in Pygmalion are what I call "icons." An icon is a graphic entity that has meaning both as a visual image and as a machine object. Icons control the execution of computer programs, because they have code and data associated with them, as well as their images on the screen. This distinguishes icons from, say, lines and rectangles in a drawing program, which have no such semantics. Pygmalion is the origin of the concept of icons as it now appears in graphical user interfaces on personal computers. After completing my thesis, I joined Xerox's "Star" computer project. The

first thing I did was recast the programmer-oriented icons of Pygmalion into office-oriented ones representing documents, folders, file cabinets, mailboxes, telephones, wastebaskets, etc. These icons have both data (e.g., document icons contain text) and behavior (e.g., when a document icon is dropped on a folder icon, the folder stores the document in the file system). This idea has subsequently been adopted by the entire personal computer and workstation industry.[7]

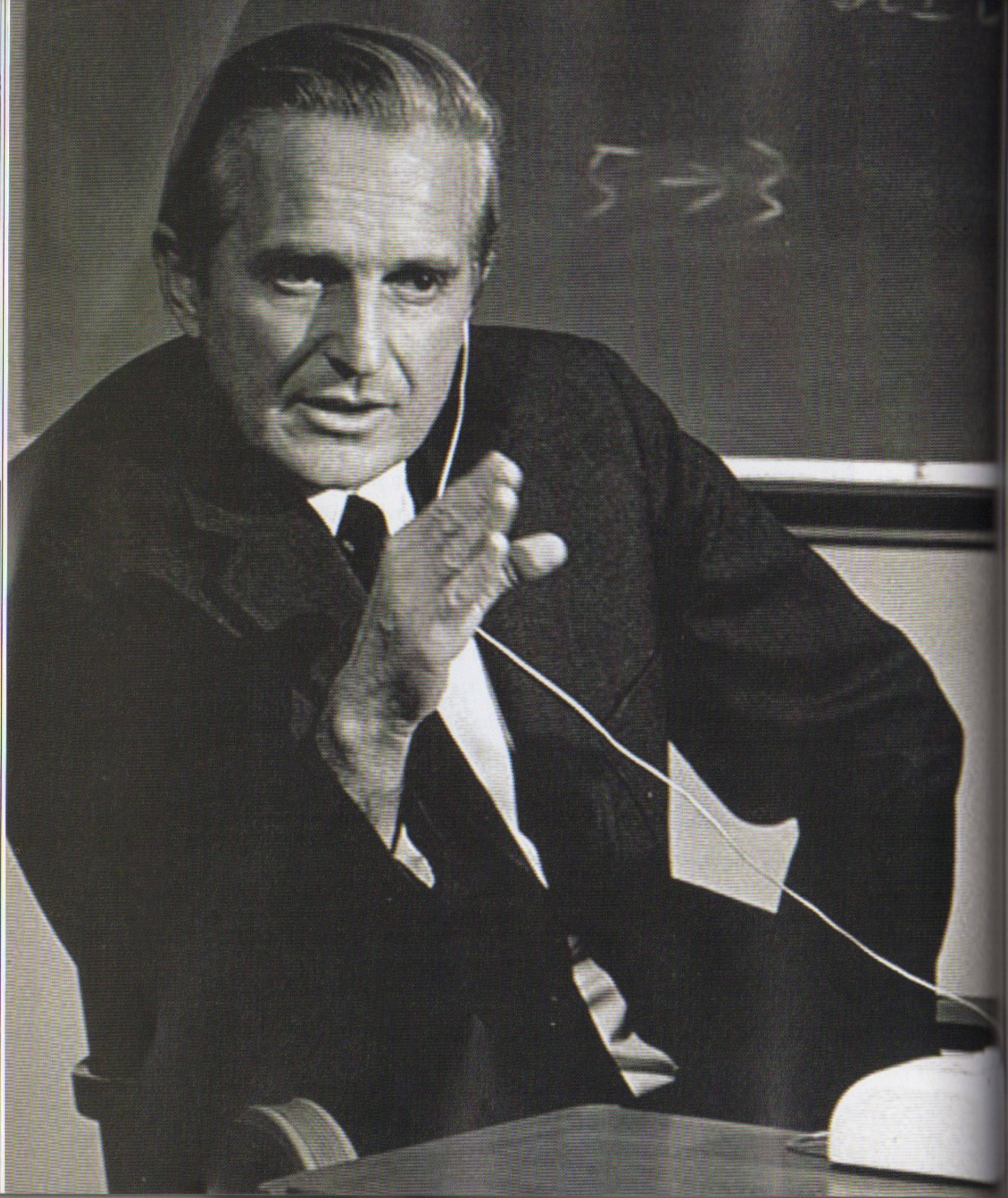
When Dave Smith had finished his PhD, he was hired to join PARC to work on the Star system. Alan Kay had always talked about the desktop metaphor and he came up with the idea of overlapping windows, from a metaphor of papers on a desktop. "You can have a lot more papers on a desk if they overlap each other, and you can see the corner of one, pull it out and put it on top."

Detailed accounts have been written[8] about how long it took to convince a copier company to commercialize the Alto, but Xerox did make a major effort to develop and market an "Office of the Future" machine. They did a thorough analysis of the business potential to find what the requirements of that machine should be, concluding that people would pay a lot for the Alto technology. Star was conceived in response to this; based on the combination of Smalltalk and Alto, it became a design that fit all of the requirements. Star was a very futuristic machine; when they were asked in the market research surveys, people responded that they would not give up that advanced interactive performance for a much inferior but less expensive machine. After it was launched, the IBM PC came out, and the people who said in the research that they would pay a lot for Star proved to be only willing to pay much less for an inferior interface.



Dr. Douglas C. Engelbart

Doug Engelbart is best known as the inventor of the mouse. At the time of writing, in 2004, at the age of seventy-eight, he is still going to work at his "Bootstrap Alliance," trying to persuade us of the value of his ideas about augmenting the human intellect, and evangelizing the virtues of his "Augment System." His office is located in the headquarters building of Logitech, the world's largest manufacturer of mice. Taking credit as the inventor of such an ubiquitous product is not enough for Doug; in fact, he is charmingly modest about that achievement, preferring to discuss other ideas that have not met with such spectacular success. He has always wanted to create designs that enhance human performance and is not interested in ease of use for the novice. He grew up in Oregon, and his electrical engineering studies were interrupted by service in World War II. During his stint in the Philippines as a naval radar technician, he came across an article by Vannevar Bush in the *Atlantic Monthly*[9] and was inspired to think of a career dedicated to connecting people to knowledge. This idealistic ambition led him to the Ames Laboratory (later NASA Ames Research Center) on the edge of the San Francisco Bay, where he worked on wind tunnel research, and then to the Stanford Research Institute, where he invented the mouse and built up the Augmentation Research Center (ARC) with funding from ARPA. In the early seventies he took several members of his team to Xerox PARC, where he helped put the mouse and the desktop together.



Doug Engelbart conducting a workshop circa 1967–68

Photo Bootstrap Institute

Doug Engelbart

Inventing the Mouse

In 1995 the International World Wide Web Conference committee presented the first SoftQuad Web Award to Dr. Douglas C. Engelbart, to commemorate a lifetime of imagination and achievement and to acknowledge his formative influence on the development of graphical computing, in particular for his invention of the mouse. This is the contribution for which Doug is universally acclaimed. In spite of this, he himself is inclined to give credit to the trackball:

> When I was a senior in electrical engineering, some of the experiments we had to do in the laboratory would end up resulting in funny shaped curves that curled back on themselves, and it was the area under the curve that we were experimenting with. They had an elbow shaped device there, a platform sort of thing that would set down on the table. You would run the pointer around that area, and there was a small wheel next to the pointer, resting on the tabletop, and the other side of the joint there was another one. I couldn't figure out how that would produce the area under the curve.
>
> "I'm not sure myself about the mathematics," said the professor, "but it is a fact that the little wheels will only roll in the axis of the rolling direction, and will slide sideways."

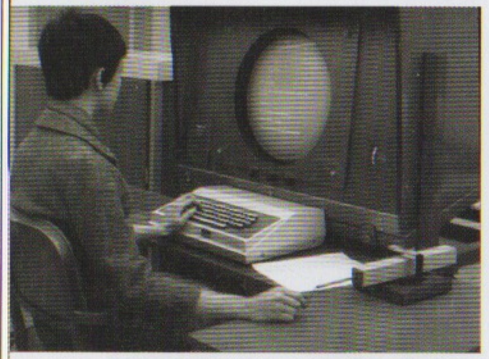
I began to understand that the wheel would roll only as far as you went in the one direction, irrespective of how many sideways movements you made.

I thought of that again one time during a conference on computer graphics, when I was feeling rebellious and bored, so I wrote in my notes about putting two wheels at right angles to each other, so that one would always be measuring how far you went north and south, and the other east and west. It would be easy to convert that into potentiometers and things so that the computer could pick up that signal. It was a very simple idea. At the time I was unaware that that same thing was sitting underneath the tracking ball, and that was how a tracking ball worked. Later on the manufacturers put the wheels against the table, which is exactly like an upside down tracking ball. There should be credit given to the tracking ball, except for my ignorance about it at the time.

It says a great deal about Engelbart's extraordinary modesty that he makes so light of his achievement. It also says a lot about his methodical persistence that he used his moments of boredom at that conference to fill a notebook with ideas and that he remembered what was in that notebook when he was looking for the best input device solutions many years later:

When you were interacting considerably with the screen, you needed some sort of device to select objects on the screen, to tell the computer that you wanted to do something with them. We got some funding in the early sixties, I think it was from NASA, and set up an experimental environment with several different kinds of devices; a tracking ball, a light pen, and things of that sort that were available at the time.

As we were setting up the experiments, I happened to remember some notes that I had made in a pocket notebook some years before, and sketched that out to Bill English, who was the engineer setting up the experiments, and he put one together, with the help of a few draftsmen and machinists. That one was put in the experiments, and happened to be winning all the tests. That became the pointing device for our user interface. Somebody, I can't remember who, attached the name "mouse" to it. You can picture why, because it was an object about this big, and had one button to use for selection, and had a wire running out the back.


■ Graficon experimental pointing device

"It looks like a one-eared mouse!" someone said. Soon all of us just started calling it a mouse.

Thinking of the possible relevance of those orthogonal wheels was the first step; working with Bill English to design an object to contain them was the second. The recognition that this idea might be important for interaction came from the tests that compared the mouse with other possible input devices; it was the people who used it in the tests who proved the point. The designers came up with as many alternatives as they could that seemed plausible, built prototypes and created tasks in the relevant context, and then ran the tests. Here's what Doug says about the testing process:

We listened to everybody who had strong ideas, and it seem to us worth just testing everything that was available. The light pen had been used by radar operators for years and years, and that seemed to most people would be the most natural way to do it. I couldn't see that, but why argue with them; why not just test and measure? The time it takes to grope for it and lift it up to the screen seemed excessively large, so it didn't do well in the tests.

For the test we had naive users coming in, and we explained everything that would happen so that they weren't surprised. We asked them to put their hands on the keyboard, and all of sudden an array of three-by-three objects would appear at an arbitrary place on the screen, sometimes small objects and sometimes large, and they had to hit a space bar, access the pointing device and go click on it. The computer measured time, overshoot, and any other characteristics we thought were valuable. The assessment just showed the mouse coming out ahead. It was many years later that I heard from Stuart Card, a friend at Xerox PARC, what the human factors explanation was.

There is an objectivity in this process of letting the user decide, the value of which is a recurring theme in this story of designing the desktop and the mouse. Come up with an idea, build a prototype, and try it on the intended users. That has proved, time and time again, to be the best way to create innovative solutions.

First mouse in hand, 1963–
First mou
First production mou

The Demo that Changed the World

There is a gentle modesty, even diffidence, in the way Doug Engelbart talks, but he holds your attention much more firmly than you would expect from his manner of speech. His passion for philosophy and ideas shines through, with an underlying intensity, almost fanaticism, that is charismatic. He remembers his early motivations:

> My initial framework for thinking about these questions got established in 1951. I had realized that I didn't have any great goals for my career. I was an electrical engineer with an interesting job, recently engaged to be married, but had no picture of the future, and I was embarrassed about this.
>
> "What would be an appropriate career goal for me?" I asked. "Why don't I design a career that can maximize its benefit to mankind?" I ended up saying.
>
> I was an idealistic country boy. Eventually I realized that the world is getting more complex at an ever more rapid rate, that complex problems have to be dealt with collectively, and that our collective ability for dealing with them is not improving nearly as fast as the complexity is increasing. The best thing I could think of doing was to try and help boost mankind's capability for dealing with complex problems.

By this time he had been working for a couple of years at the Ames Laboratory, in what is now the heart of Silicon Valley but was then still a pleasant agricultural countryside full of orchards. His job researching aerodynamics and wind tunnel testing was interesting and enjoyable, he was engaged to the girl of his dreams, and life might have been good enough; then that idealistic itch to change the world took over, and he started his lifelong search to develop electronic systems that would augment the human intellect. He remembered the "Memex" that Vannevar Bush had described as an "enlarged intimate supplement to a person's memory" that can be consulted with "exceeding speed and flexibility." He felt a kinship for the vision and optimism that Bush communicated and set out to find his own way of realizing an equivalent ambition.

> When I was half way through college, I was drafted for World War II, and had the good fortune to get accepted in a training program that the navy was running for electronic technicians, because the advent of radar and sonar had changed the aspects of navy problems immensely. They had a year-long program which taught me a lot of practical things about electronics and exposed me to the fact that the electronics of radar could put interesting things on the screen, so I just knew that if a computer could punch cards or send information to a printer, then electronics could put anything you want on the screen. If a radar set could respond to operators pushing buttons or cranking cranks, certainly the computer could! There was no question in my mind that the engineering for that would be feasible, so you could interact with a computer and see things on a screen. That intuitive certainty made me change my career path totally to go after it, but I had an extremely difficult time conveying that conviction to anybody else successfully for sixteen years or more.

His first step in that sixteen-year path of dogged determination was to leave his job and go to the graduate school at the University of California at Berkeley, where one of the earliest computers was being constructed. His fixed idea that people should be able to interact with computers directly did not fit with the prevailing view, so he started to get a reputation as an eccentric. Once he had his PhD, he started to look around for a place that would be more accepting of his vision than the UC Berkeley community. He talked to Bill Hewlett and David Packard, but although they were enthusiastic about his ideas, they were determined to focus on laboratory instruments rather than computers.

He finally landed a job at SRI, whose leaders were interested in researching possible uses for computers in both military and civilian applications. He started there at the end of 1957, soon after Sputnik had been launched, and the space race was getting under way. After learning the ropes at SRI for a year and a half, he started to lobby for the opportunity to start his own lab to experiment with new ways of creating and sharing knowledge by combining man and machine. His wish was granted when the U.S. Air Force Office of Scientific Research provided a small grant, and he settled down to the task of articulating his views.

"I wrote a paper that was published in 1962 called 'Augmenting the Human Intellect: A Conceptual Framework'[10] that steered my life from that point forward." In his paper he defined four areas in which human capabilities could be augmented:

1. **Artifacts**—physical objects designed to provide for human comfort, the manipulation of things or materials, and the manipulation of symbols.

2. **Language**—the way in which the individual classifies the picture of his world into the concepts that his mind uses to model that world, and the symbols that he attaches to those concepts and uses in consciously manipulating the concepts ("thinking").

3. **Methodology**—the methods, procedures, and strategies with which an individual organizes his goal-centered (problem-solving) activity.

4. **Training**—the conditioning needed by the individual to bring his skills in using augmentation means 1, 2, and 3 to the point where they are operationally effective.

The system we wish to improve can thus be visualized as comprising a trained human being, together with his artifacts, language, and methodology. The explicit new system we contemplate will involve as artifacts computers and computer-controlled information storage, information handling, and information display devices. The aspects of the conceptual framework that are discussed here are primarily those relating to the individual's ability to make significant use of such equipment in an integrated system.

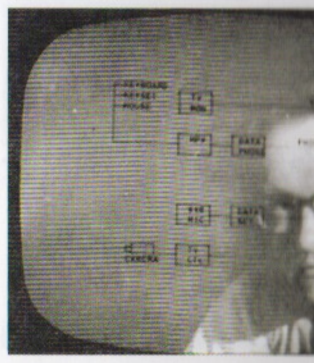
In this short quote one can see the seeds of triumph and tragedy. The triumph is Doug's powerful vision of a complete system, where people and computers are engaged in a symbiotic relationship for human benefit, working cohesively as an integrated system. From this came the mouse and the other elements of interactive computing that he pioneered. The tragedy is that training is a necessary component of the system. He developed concepts for experts, and the pursuit of the highest capability drove the design criteria; it was therefore inevitable that training would be needed to reach the level of proficiency that

would let people benefit from this capability. That proved a barrier to acceptance by ordinary people, and as computers became less expensive and more accessible, the barrier got in the way more and more.

But we are getting ahead of ourselves in the story. Let's go back to 1964 when, to the surprise of the SRI management, the Defense Advanced Research Projects Agency (DARPA) offered to fund the Augmentation Research Center (ARC) to the tune of half a million dollars a year, as well as providing a new time-sharing computer system, worth another million. Engelbart's energetic lobbying for funding and his flow of papers describing the high-level potential of automation had not been ignored by everyone, so now he had the resources he needed to move from theory to practice. He put together a stellar team of engineers for both hardware and software and set about developing NLS. Bill English was a partner for Doug in much of the work they did, leading the hardware development as the team grew to seventeen people. He joined ARC in 1964 and was the perfect complementary talent, having the technical ability to implement many of the ideas that were expressed by his boss as high-level abstractions. After four years of development, Doug took a chance to show the computer science community what he had been doing:

For the Fall Joint Computer Conference in 1968, I stuck my neck out and proposed giving a real-time demonstration, if they would give me a whole hour-and-a-half conference session. We had a timesharing computer supporting our laboratory, and small five-inch diagonal screens that were high resolution, but worked by moving the beam around (vector graphics). We put a TV camera in front of the screen and used a TV display for a larger size image. We rented microwave links (from the Menlo Park civic auditorium) up to San Francisco, and borrowed an enormous video projector, three feet by two feet by six feet, to project onto a twenty-foot screen for the audience to see, using a very novel way of converting the video sweep into modulated light.

Bill English, the genius engineer that I worked with (on the mouse also) managed to make all this work. He built a backstage mixing booth, where he could select from four video feeds, one from


Bill Englis

each of the linked displays, one looking at me, and one overhead showing my hands working. He could select from the feeds so you could see a composite image in real time. He had experience doing the stage work for amateur plays, so he was the director. I had written a script for different people to come onto the stage, so he put a speaker in my ear to let me hear his cues: sometimes it was so distracting that I would fumble words.

We were able to show high-resolution links, graphics for schematic diagrams of what was going on, the faces of members of our team in the Menlo Park Laboratory, as well as the screens that they were looking at. We had cursors controlled by two people simultaneously interacting on the screen; one guy started buzzing at my cursor as if in a fight. The audience all stood up and applauded at the end of the demo.

This was the demo that changed the world. The computer science community moved from skepticism to standing ovation in an hour and a half, and the ideas of direct manipulation of a graphical user interface became lodged in the communal consciousness. This was not punched cards and Teletypes. It was something entirely different. Doug sat alone at a console in the middle of the stage, with the twenty-foot screen behind him showing the view from the video feeds. He was wearing a short-sleeved white shirt and a thin tie, with a microphone dangling on his chest, and a headset like an aircraft controller's. The overhead camera showed his right hand using a mouse, seen for the first time by most of the audience, to point and select with; a standard typewriter keyboard in the center and a five-key command pad under his left hand. In his calm but mesmerizing voice, he described and demonstrated an amazing array of functions on the NLS system. Words were manipulated with full-screen text editing, including automatic word wrap, corrections and insertions, formatting, and printing. Documents were planned and formatted using headings and subheadings. Links were demonstrated between one document and another, and collaboration between remote participants was demonstrated in real time:

One of the basic design principles that we started with was that you want to be able to talk about any other knowledge object out there. You want your links to point in high resolution, for example to a given word in another document, so you want a link addressing string that will let you get there. You also should be able to have optional alternative views.

"I just want to see that one paragraph," I might say.

"Okay! When I get there, I'd like to have certain terms highlighted."

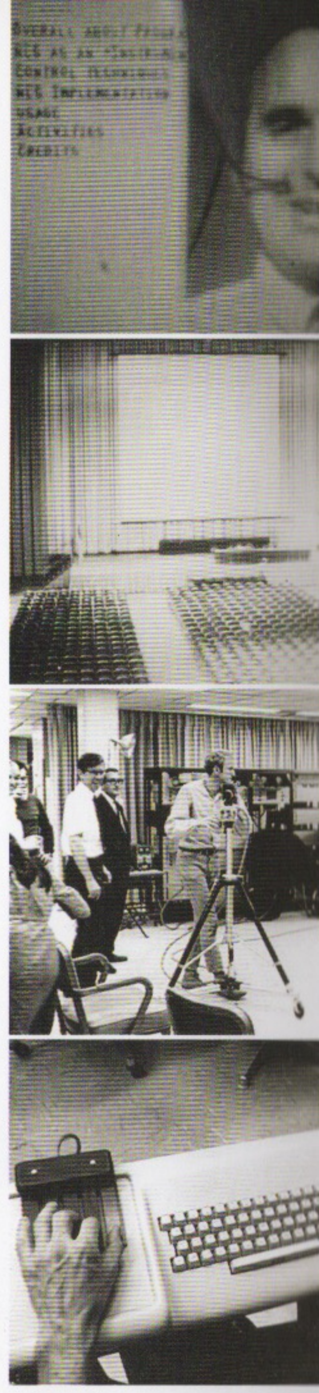
"Okay! I'd also like to know who else in my group has links to it, and what they say about it."

I wrote in 1962 that we are all used to the idea that we can diagram sentences, based on the syntactical rules for a properly structured sentence. Now we might want to see a similar set of rules for the structure of an argument, so the computer has a graphic diagram of the argument with nodes connected to other arguments, expressions and statements, and meaningful links connecting them.

The demo was truly amazing, proving that interactive computing could be used for real-time manipulations of information in ways that very few people had imagined before. The assumption that high levels of training would always be acceptable did not get in the way until ordinary people tried to become users of NLS.

The demo also positioned Doug and his band at ARC to receive continuing funding for their research until 1975. His team grew to thirty-five people at one point. In 1969 they were connected to ARPAnet as one of the original nodes of the military research connected network, which eventually developed into the Internet. NLS grew in sophistication and content as time went on but remained essentially the same in concept. In 1971 a group of the best people at ARC, including Bill English, were tempted away from SRI by the opportunities at the new Xerox PARC, where so many exciting things seemed to be about to happen. This was the start of a slide for Doug Engelbart, during which his long-held dreams seemed to have less and less influence. You can feel the frustration behind his words as he describes the determined pursuit of his ideals and bemoans the success of the desktop:

I've been pursuing for fifty years something that required higher and higher levels of capability. "How do you achieve capability," I was



Demo at Joint Computer Conference, 196

asking, "and when you achieve capability levels as high as you can get, then how do you reduce the learning costs in a reasonable way, but not try to set what I think of as an artificial level of learnability ease, and have to keep your capability enhancements within that level?"

In the business world, I understand, that is awkward to try to do, because you are competing with other people for sales, and people will try computer interfaces that will strike customers as easy to use early on when they purchase them. I don't object to having a difference, but I feel that the world should recognize that there are really high levels of capability there to pursue that will be very important for society to have reached. That's been my pursuit.

Many years ago it became clear to me that what you need to do is develop a basic software structure that will have file designs, capabilities, and properties that the very expert person could use. Then it is easy enough to support the beginner, or pedestrian user, by plugging a very simple user interface with simple operations on the front, but they can both work on the same materials.

Yes, you can point with a GUI, I admit, but our system had an indefinite number of verbs and nouns that you could employ.

There's no way that pointing and clicking at menus can compete with that. You wouldn't want to give someone directions by that limited means.

It is easy to understand the idea of going for the best, of catering to the expert user, and then providing a path to get there from a simple user interface designed for the beginner. In practice, however, this has proved to be the wrong way round, as it's not easy to get something right for the beginner when your design is already controlled by something that is difficult to learn. Look, for example, at the use of the five-key keypad for typing text. Like the stenographer's keyboard used for recording court proceedings, it enables impressive typing speeds when you have been trained long enough to become expert.

Quite a few users adopted the chording key set that I built for myself. You could type any of the characters in the alphabet with one hand, and give commands with the three-button mouse in the other hand at the same time as pointing. This gave a much richer vocabulary and a much more compact way to evoke it than the GUI.

This is how the interactions were designed. On the mouse, one button was to click, another was called *command accept*, and the third was called *command delete*. If you wanted to delete a word, you would hit the middle button on the keypad, which was the letter *d*. It was *d* because it is the fourth letter in the alphabet, and this was a binary coding, 1, 2, 4, 8, 16. If it was the letter *f*, it was the sixth letter, so you'd hit the 2 and the 4 keys at the same time. Then you pointed at and clicked the beginning of the thing you wanted deleted, then you pointed at and clicked the end of the thing you wanted deleted, and if you hadn't made any mistakes, you would hit the *command accept* key on the mouse. It was *d*, point/click, point/click, *command accept*. If you made a mistake at any point, you would hit the *command delete* key to back you up one step.

That process was complicated to learn, and it took a long time for most people to memorize the binary-based, five-finger alphabet. Alternatively, they could invoke commands with keypad, or even the keyboard, the mouse for pointing, and a conventional keyboard in between for typing text. This would have been a very good solution for people with four hands, but is not as fast as the chorded keyboard and three-button mouse, as it takes longer to move your hands to and from the keyboard.

Doug Engelbart strives consistently toward a goal of the best possible performance, and his intuitions and insights have set the scene for the dominance of the mouse and the desktop. His influence has been limited by his decision to design for people as determined and proficient as he is himself, rather than for those who require an easy-to-use system.