CHAPTER ONE

Alan Kay's universal media machine

Medium:
8.a. A specific kind of artistic technique or means of expression as determined by the materials used or the creative methods involved: the medium of lithography.
b. The materials used in a specific artistic technique: oils as a medium.

American Heritage Dictionary, 4th edition
(Houghton Mifflin, 2000)

"The best way to predict the future is to invent it."

Alan Kay

Appearance versus function

Between its invention in the mid-1940s and the arrival of PCs in the early 1980s, the digital computer was mostly used for military, scientific, and business calculations and data processing. It was not interactive. It was not designed to be used by a single person. In short, it was hardly suited for cultural creation.

As a result of a number of developments of the 1980s and 1990s—the rise of the personal computer industry, adoption of Graphical User Interfaces (GUI), the expansion of computer

networks and the World Wide Web—computers moved into the cultural mainstream. Software replaced many other tools and technologies for creative professionals. It has given hundreds of millions of people the abilities to create, manipulate, sequence and share media—but has this led to the invention of fundamentally *new* forms of culture? Today media companies are busy promoting *e-books* and interactive *television*; the consumers are happily purchasing music *albums* and *feature films* distributed in digital form, as well making *photographs* and *video* with their digital cameras and cell phones; office workers are reading PDF *documents which imitate paper*.

In short, it appears that the revolution in the means of production, distribution, and access of media has not been accompanied by a similar revolution in the syntax and semantics of media. Who shall we blame for this? Shall we put the blame on the pioneers of cultural computing—J. C. R. Licklider, Ivan Sutherland, Ted Nelson, Douglas Engelbart, Seymour Paper, Nicholas Negroponte, Alan Kay, and others? Or, as Nelson and Kay themselves are eager to point out, does the problem lie with the way the industry implemented their ideas?

Before we blame the industry for bad implementation—we can always pursue this argument later if necessary—let us look into the thinking of the inventors of cultural computing themselves. For instance, what about the person who guided the development of a prototype of a modern person computer—Alan Kay?

Between 1970 and 1981 Alan Kay was working at Xerox PARC—a research center established by Xerox in Palo Alto. Building on the already accomplished work of the pioneers of cultural computing, the Learning Research Group at Xerox PARC headed by Kay, systematically articulated the paradigm and the technologies of *vernacular media computing*, as it exists today.[1]

[1] Kay has expressed his ideas in a few articles and a large number of interviews and public lectures. The following have been my main primary sources: Alan Kay and Adele Goldberg, *Personal Dynamic Media*, *IEEE Computer* 10, no. 3 (1977); Alan Kay, "The Early History of Smalltalk," The 2nd ACM SIGPLAN Conference on History of Programming Languages (New York: ACM, 1993), pp. 69–95; Alan Kay, "A Personal Computer for Children of All Ages," *Proceedings of the ACM 1972 National Conference* (Boston, 1972); Alan Kay, *Doing with Images Makes Symbols*, videotape (University Video Communications, 1987), http://archive.org/details/AlanKeyD1987/; Alan Kay, "User Interface: A Personal View," in *The Art of Human-Computer Interface Design*, ed. Brenda Laurel (Reading, Mass: Addison-Wesley, 1990), pp. 191–207; David Canfield Smith *et al.*, "Designing the Star user Interface," *Byte*, issue 4 (1982).

Although selected artists, filmmakers, musicians, and architects were already using computers since the 1950s, often developing their software in collaboration with computer scientists working in research labs (Bell Labs, IBM Watson Research Center, etc.) most of this software was aimed at producing only particular kinds of images, animations or music, congruent with the ideas of their authors. In addition, each program was designed to run on a particular machine. Therefore, these software programs could not function as general-purpose tools easily usable by others.

It is well known most of the key ingredients of personal computers as they exist today came out of Xerox PARC: the Graphical User Interface with overlapping windows and icons, bitmapped display, color graphics, networking via Ethernet, mouse, laser printer, and WYSIWYG ("what you see is what you get") printing. But what is equally important is that Kay and his colleagues also developed a range of applications for media manipulation and creation that also all used a graphical interface. They included a word processor, a file system, a drawing and painting program, an animation program, a music editing program, etc. Both the general user interface and the media manipulation programs were written in the same programming language, Smalltalk. While some of the applications were programmed by members of Kay's group, the users that included seventh-grade high-school students programmed others.[2] (This was consistent with the essence of Kay's vision: to provide users with a programming environment, examples of programs, and already-written general tools so the users would be able to make their own creative tools.)

When Apple introduced the first Macintosh computer in 1984, it brought the vision developed at Xerox PARC to consumers (the new computer was priced at USD $2,495). The original Macintosh 128K included a word processing and a drawing application (MacWrite and MacPaint, respectively). Within a few years these were joined by other software for creating and editing

[2] Alan Kay and Adele Goldberg, "Personal Dynamic Media," in *New Media Reader*, ed. Noah Wardrip-Fruin and Nick Montfort (The MIT Press, 2003), p. 399.

different media: Word, PageMaker and VideoWorks (1985),[3] SoundEdit (1986), Freehand and Illustrator (1987), Photoshop (1990), Premiere (1991), After Effects (1993), and so on. In the early 1990s, similar functionality became available on PCs running Microsoft Windows.[4] And while Macs and PCs were at first not fast enough to offer true competition for traditional media tools and technologies (with the exception of word processing), other computer systems specifically optimized for media processing started to compete with these technologies in the 1980s. (The examples are the NeXT Workstation, produced between 1989 and 1996; Amiga, produced between 1985 and 1994; and Paintbox, first released in 1981.)

By around 1991, the new identity of a computer as a personal media editor was firmly established. (This year Apple released QuickTime, which brought video to the desktop; the same year saw the release of James Cameron's *Terminator II*, which featured pioneering computer-generated special effects). The vision developed at Xerox PARC became a reality—or rather, one important part of this vision in which the computer was turned into a personal machine for display, authoring and editing content in different media. And while in most cases Alan Kay and his collaborators were not the first to develop particular kinds of media applications—for instance, paint programs and animation programs were already written in the second part of the 1960s[5]—by implementing all of them on a single machine and giving them consistent appearance and behavior, Xerox PARC researchers established a new paradigm of media computing.

I think that I have made my case. The evidence is overwhelming. It is Alan Kay and his collaborators at PARC that we must take to task for making digital computers imitate older media. By developing easy-to-use GUI-based software to create and edit familiar media types, Kay and others appear to have locked the computer into being a simulation machine for "old media." Or, to put this in terms of Jay Bolter and Richard Grusin's influential book *Remediation: Understanding New Media* (2000), we can say that GUI-based software turned a digital computer into a "remediation machine:"

[3] Videoworks was renamed Director in 1987.

[4] 1982: AutoCAD; 1989: Illustrator; 1992: Photoshop, QuarkXPress.

[5] See http://sophia.javeriana.edu.co/~ochavarr/computer_graphics_history/historia/

a machine that expertly represents a range of earlier media. (Other technologies developed at PARC, such as the bitmapped color display used as the main computer screen, laser printing, and the first Page Description Language which eventually lead to Postscript, were similarly conceived to support the computer's new role as a machine for simulation of physical media.)

Bolter and Grusin define remediation as "the representation of one medium in another."[6] According to their argument, new media always remediate the old ones and therefore we should not expect that computers would function any differently. This perspective emphasizes the continuity between computational media and earlier media. Rather than being separated by different logics, all media including computers follow the same logic of remediation. The only difference between computers and other media lies in how and what they remediate. As Bolter and Grusin put this in the first chapter of their book, "What is new about digital media lies in their particular strategies for remediating television, film, photography, and painting." In another place in the same chapter they make an equally strong statement that leaves no ambiguity about their position: "We will argue that remediation is a defining characteristic of the new digital media."

If today we consider all the digital media created by both consumers and by professionals—digital photography and video shot with inexpensive cameras and cell phones, the contents of personal blogs and online journals, illustrations created in Photoshop, feature films cut on Avid, etc.—in terms of its appearance digital media indeed often looks exactly the same way as media before computers. Thus, if we limit ourselves to looking at the media surfaces, the remediation argument accurately describes much of computational media. But rather than accepting this condition as an inevitable consequence of the universal logic of remediation, we should ask why this is the case. In other words, if contemporary computational media imitates other media, how did this become possible? There was definitely nothing in the original theoretical formulations of digital computers by Turing or Von Neumann about computers imitating other media such as books, photography, or film.

[6] Jay Bolter and Richard Grusin, *Remediation: Understanding New Media* (The MIT Press, 2000).

The conceptual and technical gap which separates the first room-sized computers used by the military to calculate the shooting tables for anti-aircraft guns and crack German communication codes, and contemporary small desktops and laptops used by ordinary people to create, edit and share media is vast. The contemporary identity of a computer as a media processor took about forty years to emerge, if we count from 1949 when MIT's Lincoln Laboratory started to work on first interactive computers to 1989 when the first commercial version of Photoshop was released. *It took generations of brilliant and creative thinkers to invent the multitude of concepts and techniques that today make possible for computers to "remediate" other media so well. What were their reasons for doing this? What was their thinking?* In short, why did these people dedicate their careers to inventing the ultimate "remediation machine"?

While media theorists have spent considerable efforts in trying to understand the relationships between digital media and older physical and electronic media in the 1990s and 2000s, the important sources—the writing and projects by Ivan Sutherland, Douglas Engelbart, Ted Nelson, Alan Kay, and other pioneers working in the 1960s and 1970s—remained largely unexamined. This book does not aim to provide a comprehensive intellectual history of the invention of media computing. Thus, I am not going to consider the thinking of all key figures in the history of media computing (to do this right would require more than one book). Rather, my concern is with the present and the future. Specifically, I want to understand some of the dramatic transformations in what media is, what it can do, and how we use it—the transformations that are clearly connected to the shift from previous media technologies to software. Some of these transformations had already taken place in the 1990s but were not much discussed at the time (for instance, the emergence of a new language of moving images and visual design in general). Others have not even been named yet. Still others—such as remix and mashup culture—are being referred to all the time, and yet the analysis of how they were made possible by the evolution of media software has so far not been attempted.

In short, I want to understand what is *"media after software"*—that is, what happened to the techniques, languages, and the concepts of twentieth-century media as a result of their computerization. Or, more precisely, what has happened to media after they have been *software-sized*. (And since in the space of a single book I can only consider some of these techniques, languages and concepts, I will focus on those that, in my opinion, have not been yet discussed by others.)

In this chapter we will take a closer look at one place where the identity of a computer as a "remediation machine" was largely put in place—Alan Kay's Learning Research Group at Xerox PARC, in operation during the 1970s. We can ask two questions: first, what exactly did Kay want to do, and second, how did he and his colleagues go about achieving it? The brief answer—which will be expanded below—is that Kay wanted to turn computers into a "personal dynamic media" which could be used for learning, discovery, and artistic creation. His group achieved this by systematically simulating most existing media within a computer while simultaneously adding many new properties to these media. Kay and his collaborators also developed a new type of programming language that, at least in theory, would allow the users to quickly invent new types of media using the set of general tools already provided for them. All these tools and simulations of already existing media were given a unified user interface designed to activate multiple mentalities and ways of learning—kinesthetic, iconic, and symbolic.

Kay conceived of "personal dynamic media" as a fundamentally new kind of media with a number of historically unprecedented properties such as the ability to hold all the user's information, simulate all types of media within a single machine, and "involve the learner in a two-way conversation."[7] These properties enable new relationships between the user and the media s/he may be creating, editing, or viewing on a computer. And this is essential if we want to understand the relationships between computers and earlier media. Briefly put, while visually, computational media may closely mimic other media, these media now function in different ways.

For instance, consider digital photography, which often imitates traditional photography in appearance. For Bolter and Grusin, this is an example of how digital media 'remediates" its predecessors.

[7] Since the work of Kay's group in the 1970s, computer scientists, hackers and designers added many other unique properties—for instance, we can quickly move media around the net and share it with millions of people using Flickr, YouTube, and other sites.

But rather than only paying attention to their appearance, let us think about how digital photographs can function. If a digital photograph is turned into a physical object in the world—an illustration in a magazine, a poster on the wall, a print on a t-shirt—it functions in the same ways as its predecessor (unless it has augmented reality features, like IKEA's 2013 catalog).[8] But if we leave the same photograph inside its native computer environment—which may be a laptop, a network storage system, or any computer-enabled media device such as a cell phone which allows its user to edit this photograph and move it to other devices and the Internet—it can function in ways which, in my view, make it radically different from its traditional equivalent. To use a different term, we can say that a digital photograph offers its users many "affordances" that its non-digital predecessor did not. For example, a digital photograph can be quickly modified in numerous ways and equally quickly combined with other images; instantly moved around the world and shared with other people; and inserted into a multimedia document, or an architectural 3D design. Furthermore, we can automatically (i.e., by running the appropriate algorithms) improve its contrast, make it sharper, and even in some situations remove blur.

Note that only some of these new properties are specific to a particular medium—in our example, a digital photograph, i.e. an array of pixels represented as numbers. Other properties are shared by a larger class of media species—for instance, at the current stage of digital culture, all types of media files can be attached to an email message. Still others are even more general features of a computer environment within the current GUI paradigm as developed forty years ago at PARC: for instance, the fast response of a computer to a user's actions which ensures "no discernible pause between cause and effect."[9] Still others are enabled by network protocols such as TCP/IP that allow all kinds of computers and other devices to be connected to the same network. In summary, we can say that only some of the "new DNA" of a digital photograph is due its particular place of birth, i.e., inside a digital camera. Many others

[8] Roberto Baldwin, "Ikea's Augmented Reality Catalog Will Let You Peek Inside Furniture," July 20, 2012, http://www.wired.com/gadgetlab/2012/07/ikeas-augmented-reality-catalog-lets-you-peek-inside-the-malm/

[9] Kay and Goldberg, *Personal Dynamic Media*, p. 394.

are the result of the current paradigm of network computing in general.

Before diving further into Kay's ideas, I should more fully disclose my reasons for focusing on him as opposed to somebody else. The story I will present could also be told differently. It is possible to put Sutherland's work on Sketchpad in the center of computational media history; or Engelbart and his Research Center for Augmenting Human Intellect which throughout the 1960s developed hypertext (independently of Nelson), the mouse, the window, the word processor, mixed text/graphics displays, and a number of other "firsts." Or we can shift focus to the work of the Architecture Machine Group at MIT, which since 1967 was headed by Nicholas Negroponte (in 1985 this group became the MIT Media Lab). We also need to recall that by the time Kay's Learning Research Group at PARC fleshed out the details of GUI and programmed various media editors in Smalltalk (a paint program, an illustration program, an animation program, etc.), artists, filmmakers and architects were already using computers for more than a decade and a number of large-scale exhibitions of computer art were put in major museums around the world such as the Institute of Contemporary Art, London, The Jewish Museum, New York, and Los Angeles County Museum of Art. And certainly, in terms of advancing computer techniques for visual representation enabled by computers, other groups of computer scientists were already ahead. For instance, at the University of Utah, which became the main place for computer graphics research during the first half of the 1970s, scientists were producing 3D computer graphics far superior to the simple images that could be created on computers being built at PARC. Next to the University of Utah, a company called Evans and Sutherland (headed by the same Ivan Sutherland who was also teaching at the University of Utah) was already using 3D graphics for flight simulators—essentially pioneering the type of new media that can be called "navigable 3D virtual space."

While the practical work accomplished at Xerox PARC to establish the computer as a comprehensive media machine is one of my reasons, it is not the only one. The key reason I decided to focus on Kay is his theoretical formulations that place computers in relation to other media and media history. While Vannevar Bush, J. C. R. Licklider and Douglas Engelbart were primary concerned

with augmentation of intellectual and in particular scientific work, Kay was equally interested in computers as "a medium of expression through drawing, painting, animating pictures, and composing and generating music."[10] Therefore if we really want to understand how and why computers were redefined as a culture machine, and how the new computational media is different from earlier physical and electronic media, I think that Kay provides us with the best theoretical perspective.

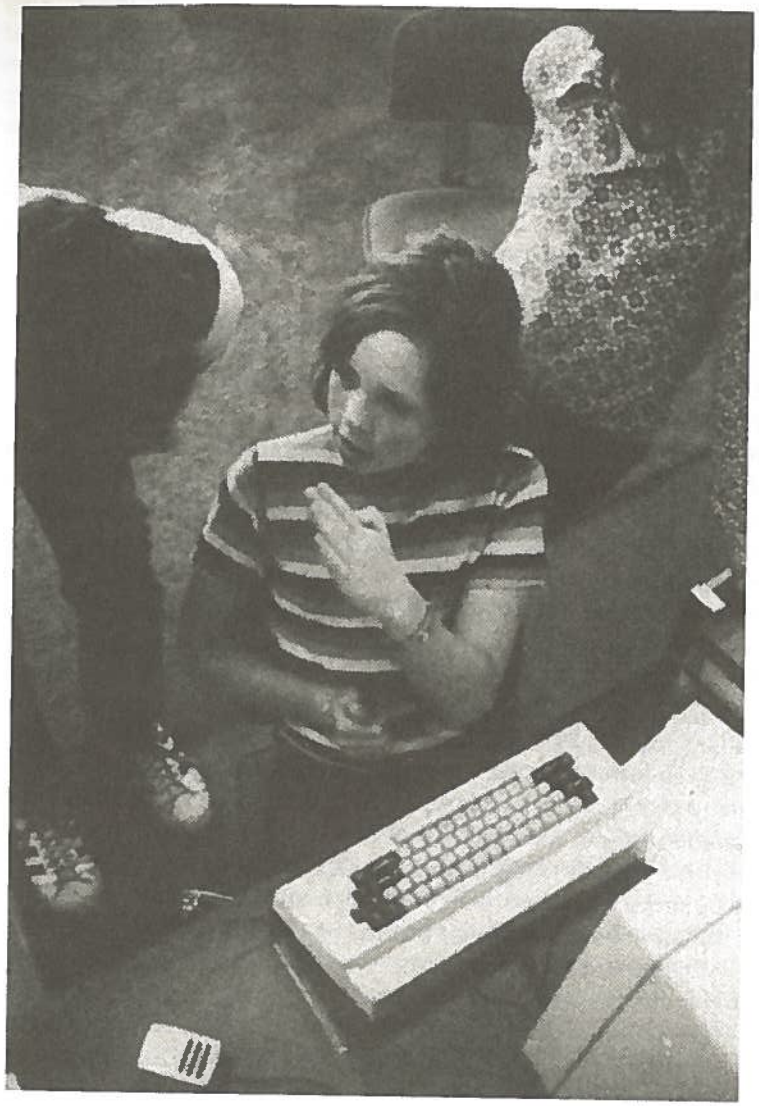
"Simulation is the central notion of the Dynabook"

While Alan Kay articulated his ideas in a number of articles and talks, his 1977 article co-authored with one of his main PARC collaborators, computer scientist Adele Goldberg, is a particularly useful resource if we want to understand contemporary computational media. In this article Kay and Goldberg describe the vision of the Learning Research Group at PARC in the following way: to create "*a personal dynamic medium* the size of a notebook (the Dynabook) which could be owned by everyone and could have the power to handle virtually all of its owner's information-related needs."[11] (The actual Alto computer built at Xerox PARC was the size of later PCs; the article strategically refers to it as "interim dynabook.") Kay and Goldberg ask the readers to imagine that this device "had enough power to outrace your senses of sight and hearing, enough capacity to store for later retrieval thousands of page-equivalents of reference materials, poems, letters, recipes, records, drawings, animations, musical scores, waveforms, dynamic simulations and anything else you would like to remember and change."[12]

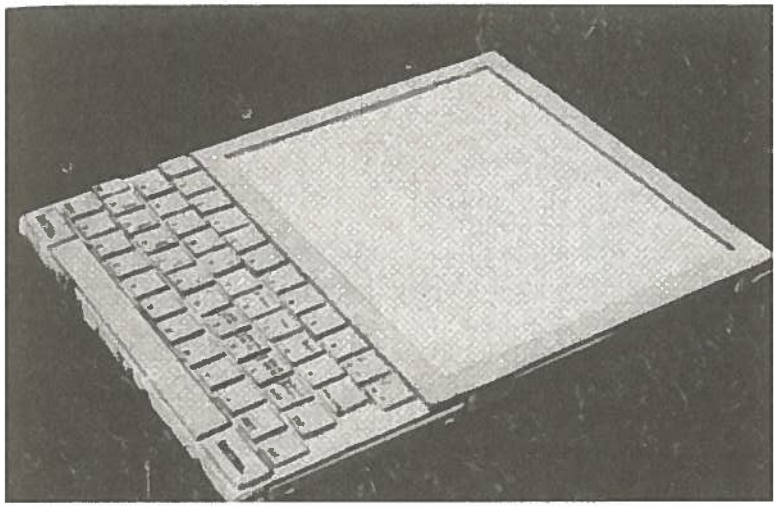
In my view, "all" in the first statement is important: it means that the Dynabook—or computational media environment in general, regardless of the size of a form of device in which it

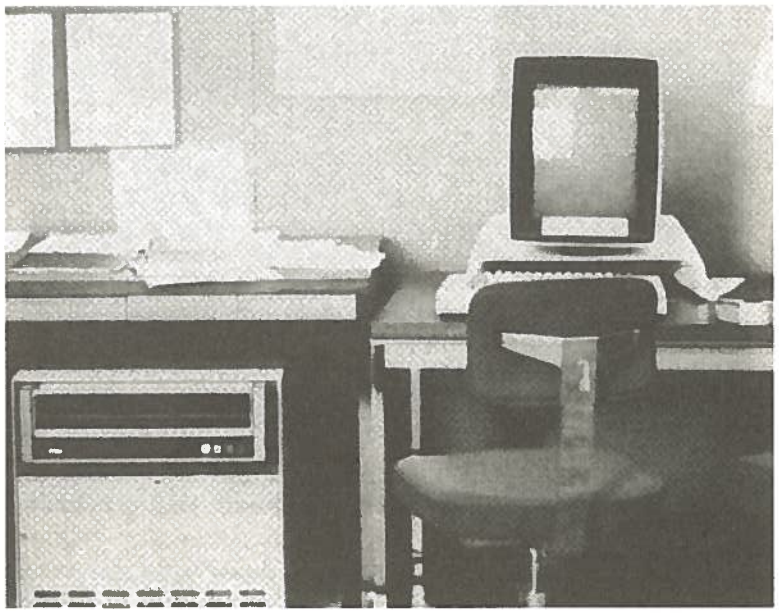
[10] *Ibid.*, p. 393.
[11] *Ibid.*, p. 393. The emphasis in this and all following quotes from this article is mine—L. M.
[12] *Ibid.*, p. 394.

is implemented—should support viewing, creating and editing all possible media traditionally used for human expression and communication. Accordingly, while separate programs to create works in different media were already in existence, Kay's group for the first time implemented them all together within a single machine. In other words, Kay's paradigm was not to simply create a new type of computer-based media that would co-exist with other physical media. Rather, the goal was to establish a computer as an umbrella, a platform for *all* existing expressive artistic media. (At the end of the article Kay and Goldberg give a name for this platform, calling it a "metamedium.") This paradigm changes our understanding of what media is. From Gotthold Ephraim Lessing's *Laocoon; or, On the Limits of Painting and Poetry* (1766) to Nelson Goodman's *Languages of Art* (1968), the modern discourse about media depends on the assumption that different mediums have distinct properties and in fact should be understood in opposition to each other. Putting all mediums within a single computer environment does not necessarily erase all differences in what various mediums can represent and how they are perceived—but it does bring them closer to each other in a number of ways. Some of these new connections were already apparent to Kay and his colleagues; others became visible only decades later when the new logic of media set in place at PARC unfolded more fully; some may still not be visible to us today because they have not been given practical realization. One obvious example of such connections is the emergence of multimedia as a standard form of communication: web pages, PowerPoint presentations, multimedia artwork, mobile multimedia messages, media blogs, and other communication forms which combine multiple mediums. Another is the adoption of common interface conventions and tools which we use in working with different types of media regardless of their origin: for instance, a virtual camera, a magnifying lens, and of course the omnipresent copy, cut and paste commands. Yet another is the ability to map one media into another using appropriate software—images into sound, sound into images, quantitative data into a 3D shape or sound, etc.—used widely today in such areas as DJ/VJ/live cinema performances and information visualization. All in all, it is as though different media are actively trying to reach towards each other, exchanging properties and letting each other borrow their unique features. (This situation is the direct opposite

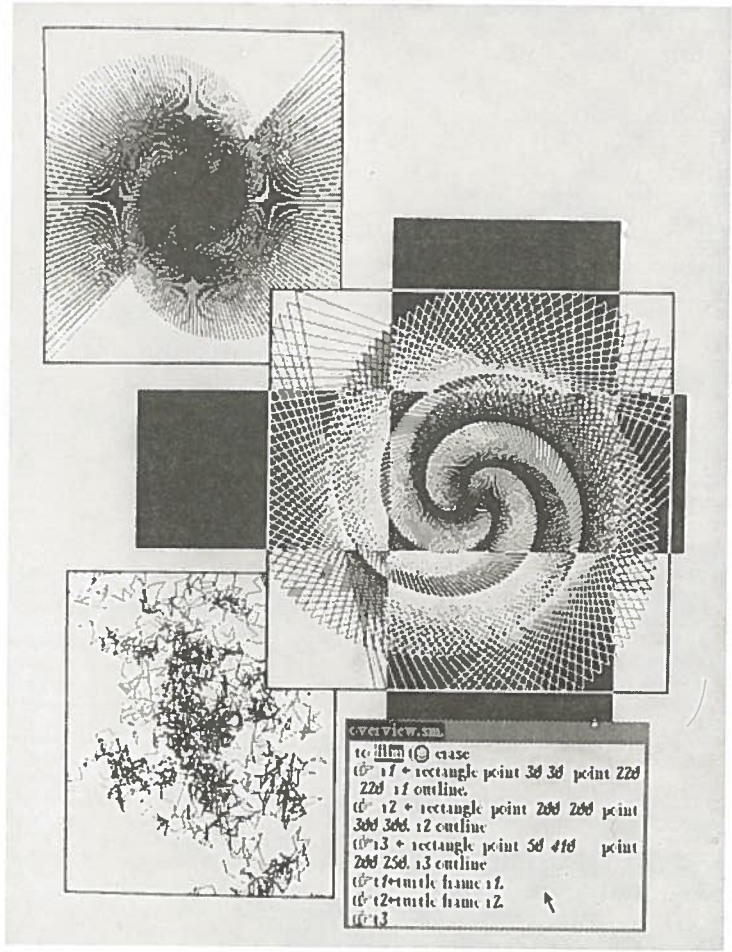


"Kids learning to use the interim Dynabook." (The original caption from the article.)

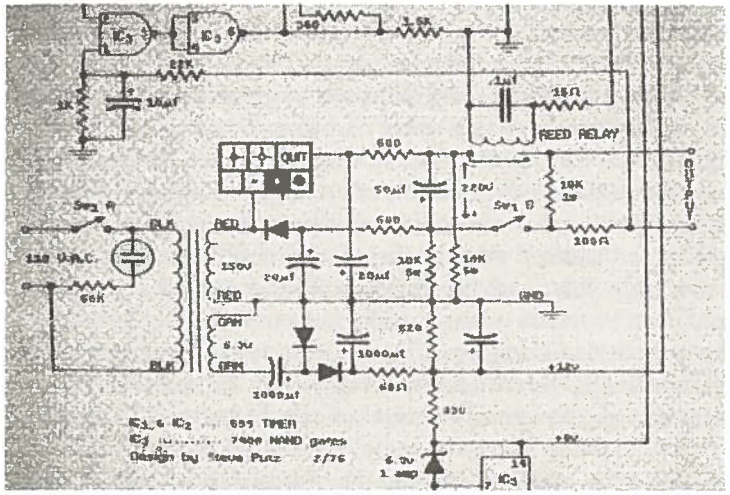


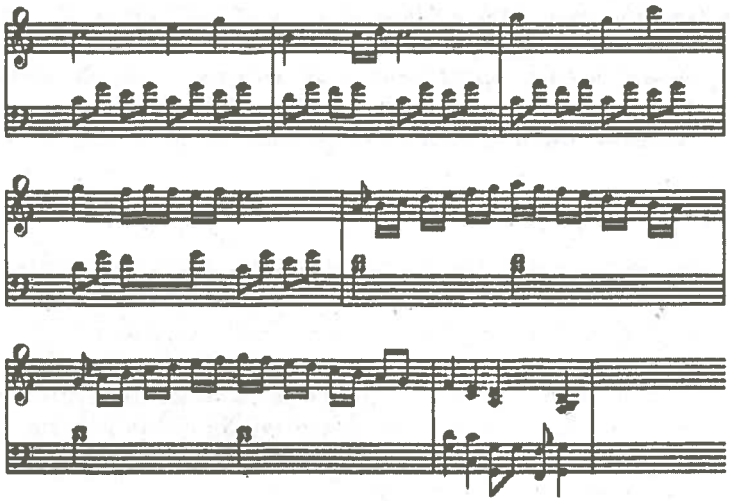


"The interim Dynabook system consists of processor, disk drive, display, keyboard, and pointing devices." (The original caption from the article.)



The Alto Screen showing windows with graphics drawn using commands in Smalltalk programming language.





Top: "An electronic circuit layout system programmed by a 15-year-old student" Bottom: "Data for this score was captured on a musical keyboard. A program then converts the data to standard musical notation." (The original captions from the article.)

of the modernist media paradigm of the early twentieth century, which was focused on discovering a unique language for each artistic medium.)

Alan Turing theoretically defined a computer as a machine that can simulate a very large class of other machines, and it is this simulation ability that is largely responsible for the proliferation of computers in modern society. But as I have already mentioned, neither he nor other theorists and inventors of digital computers explicitly considered that this simulation could also include media. It was only Kay and his generation that extended the idea of simulation to media—thus turning Universal Turing Machine into a *Universal Media Machine*, so to speak.

Accordingly, Kay and Goldberg write: "In a very real sense, simulation is the central notion of the Dynabook."[13] When we use computers to simulate some process in the real world—the behavior of a weather system, the processing of information in the brain, the deformation of a car in a crash—our concern is to correctly model the necessary features of this process or system. We want to be able to test how our model would behave in different conditions with different data, and the last thing we want to do is for computers to introduce some new properties into the model that we ourselves did not specify. In short, when we use computers as a general-purpose medium for simulation, we want this medium to be completely "transparent."

But what happens when we simulate different media in a computer? In this case, the appearance of new properties may be welcome as they can extend the expressive and communication potential of these media. Appropriately, when Kay and his colleagues created computer simulations of existing physical media—i.e. the tools for representing, creating, editing, and viewing these media—they "added" many new properties. For instance, in the case of a book, Kay and Goldberg point out "It need not be treated as a simulated paper book since this is *a new medium with new properties*. A dynamic search may be made for a particular context. The non-sequential nature of the file medium and the use of dynamic manipulation allow a story to have many accessible points of view."[14] Kay and his colleagues also added various other

[13] *Ibid.*, p. 399.
[14] *Ibid.*, p. 395.

properties to the computer simulation of paper documents. As Kay has referred to this in another article, his idea was not to simply imitate paper but rather to create "magical paper."[15] For instance, the PARC team gave users the ability to modify the fonts in a document and create new fonts. They also implemented another important idea that had already been developed by Douglas Engelbart's team in the 1960s: the ability to create different views of the same structure (I will discuss this in more detail below). And both Engelbart and Ted Nelson had already "added" something else: the ability to connect different documents or different parts of the same document through hyperlinking—i.e. what we now know as hypertext and hypermedia. Engelbart's group also developed the ability for multiple users to collaborate on the same document. This list goes on and on: e-mail in 1965, newsgroups in 1979, World Wide Web in 1990, etc.

Each of these new properties had far-reaching consequences. Take Search, for instance. Although the ability to search through a page-long text document does not sound like a very radical innovation, as the document gets longer this ability becomes more and more important. It becomes absolutely crucial if we have a very large collection of documents—such as all the web pages on the Web. Although current search engines are far from being perfect and new technologies will continue to evolve, imagine how different the culture of the Web would be without them.

Or take the capacity to collaborate on the same document(s) by a number of users connected to the same network. While it was already widely used by companies in the 1980s and 1990s, it was not until the early 2000s that the wider public saw the real cultural potential of this "addition" to print media. By harvesting the small amounts of labor and expertise contributed by a large number of volunteers, social software projects—most famously, Wikipedia—created vast and dynamically updatable pools of knowledge which would be impossible to create in traditional ways. (In a less visible way, every time we do a search on the Web and then click on some of the results, we also contribute to a knowledge-set used by everybody else. In deciding in which sequence to present the results of a particular search, Google's algorithms take into account which

[15] Alan Kay, "User Interface: A Personal View," p. 199.

among the results of previous searches for the same words people found most useful.)

Studying the writings and public presentations of the people who invented interactive media computing—Sutherland, Engelbart, Nelson, Negroponte, Kay, and others—makes it clear that they did not produce the new properties of computational media as an afterthought. On the contrary, they knew that they were turning physical media into new media. In 1968 Engelbart gave his famous demo at the Fall Joint Computer Conference in San Francisco before a few thousand people that included computer scientists, IBM engineers, people from other companies involved in computers, and funding officers from various government agencies.[16] Although Engelbart had only ninety minutes, he had a lot to show. Over the few previous years, his team at The Research Center for Augmenting Human Intellect had essentially developed the modern *office* environment as it exists today (not be confused with the modern *media design* environment which was developed later at PARC). Their NLS computer system included word processing with outlining features, documents connected through hypertext, online collaboration (two people at remote locations working on the same document in real-time), online user manuals, online project planning systems, and other elements of what is now called "computer-supported collaborative work." The team also developed the key elements of modern user interface that were later refined at PARC: a mouse and multiple windows.

Paying attention to the sequence of the demo reveals that while Engelbart had to make sure that his audience would be able to relate the new computer system to what they already knew and used, his focus was on new features of simulated media never before available previously.[17] Engelbart devotes the first segment of the demo to word processing, but as soon as he briefly demonstrated text entry, cut, paste, insert, naming and saving files—in other words, the set of tools which make a computer into a more versatile typewriter—

[16] M. Mitchell Waldrop, *The Dream Machine: J. C. R. Licklider and the Revolution That Made Computing Personal* (Viking, 2001), p. 287.

[17] Complete video of Engelbardt's 1968 demo is available at http://sloan.stanford.edu/MouseSite/1968Demo.html. For the detailed descriptions of NLS functions, see Augmentation Research Center, "NLS User Training Guide," Stanford Research Institute: Menlo Park, California), 1997, http://bitsavers.org/pdf/sri/arc/NLS_User_Training_Guide_Apr77.pdf

he then goes on to show in more depth the features of his system which no writing medium had before: "view control." As Engelbart points out, the new writing medium could switch at the user's wish between *many different views of the same information*. A text file could be sorted in different ways. It could also be organized as a hierarchy with a number of levels, as in outline processors or outlining mode of contemporary word processors such as Microsoft Word. For example, a list of items can be organized by categories and individual categories can be collapsed and expanded.

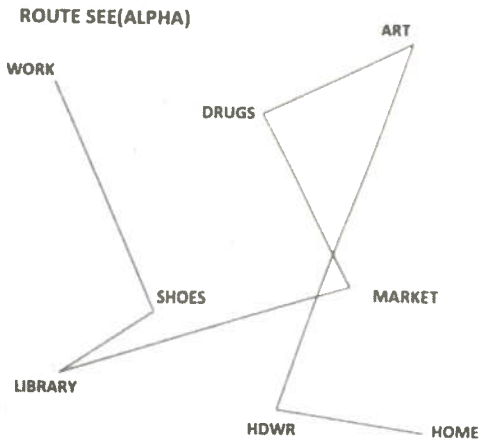
Engelbart next shows another example of view control, which today, forty-five years after his demo, is still not available in popular document management software. He makes a long "to do" list and organizes it by locations. He then instructs the computer to display these locations as a visual graph (a set of points connected by lines.) In front of our eyes, representation in one medium changes into another medium—text becomes a graph. But this is not all. The user can control this graph to display different amounts of information—something that no image in physical media can do. As Engelbart clicks on different points in a graph corresponding to particular locations, the graph shows the appropriate part of his "to do" list. (This ability to interactively change how much and what information an image shows is particularly important in today's information visualization applications.)

Next Engelbart presents "a chain of views" which he prepared beforehand. He switches between these views using "links" which may look like hyperlinks the way they exist on the Web today—but they actually have a different function. Instead of creating a path between many different documents *à la* Vannevar Bush's Memex (often seen as the precursor to modern hypertext), Engelbart is using links as a method for switching between different views of a single document organized hierarchically. He brings a line of words displayed in the upper part of the screen; when he clicks on these words, more detailed information is displayed in the lower part of the screen. This information can in turn contain links to other views that show even more detail.[18]

[18] For the detailed descriptions of these and other capabilities of NLS, see Augmentation Research Center, "NLS User Training Guide," Stanford Research Institute: Menlo Park, California), 1997, http://bitsavers.org/pdf/sri/arc/NLS_User_Training_Guide_Apr77.pdf

```
MARKET
  PRODUCE
      ORANGES
      APPLES
      BANANAS
      CARROTS
      LETTUCE
      BEANS
  CANS
      APPLE SAUCE
      BEAN SOUP
      TOMATO SOUP
  CEREALS
      BREAD
      NOODLES (ELBOW KIND)
      FRENCH BREAD
  COLD LOCKER
      MILK
```

```
1  ROUTE
2  (MARKET)   MARKET
3  (SHOES)    SHOE STORE
4  (HDWR)     HARDWARE
5  (ART)      ART SUPPLY
6  (DRUGS)    DRUG STORE
7  (LIBRARY)  LIBRARY
```

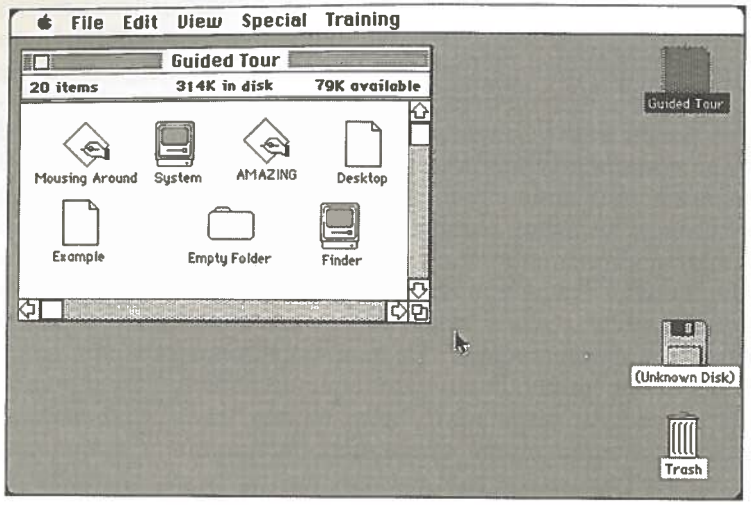


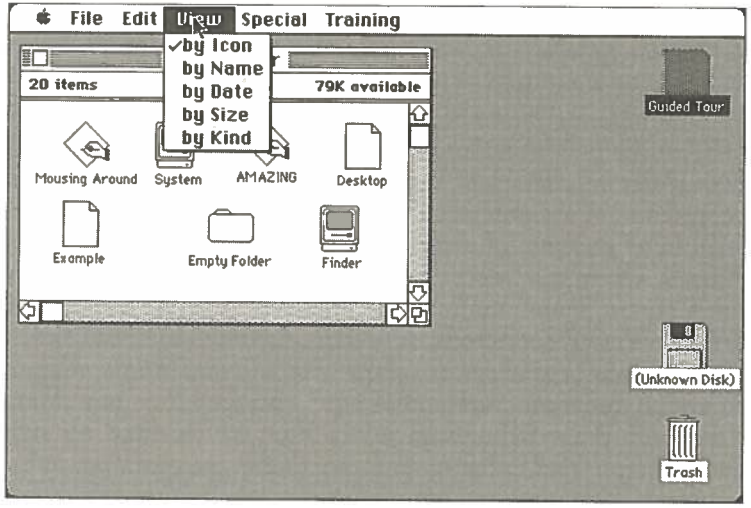
Examples of "view control" as implemented in NLS. Top left: a hierarchical view of a shopping list. Top right: a collapsed view sorted by location. Bottom: a graph view showing the sequence of locations. (Text and graphics were traced from the original video of Engelbart's 1968 demo.)

Rather than using links to drift through the textual universe associatively and "horizontally," we move "vertically" between more general and more detailed information. Appropriately, in Engelbart's paradigm, we are not "navigating"—we are "switching views." We can create many different views of the same information and switch between these views in different ways. And this is what Engelbart systematically explains in this first part of his demo. He demonstrates that you can change views by issuing commands, by typing numbers that correspond to different parts of a hierarchy, by clicking on parts of a picture, or on links in the text. (In 1967 Ted Nelson articulated and named a similar idea of a type of hypertext, which would allow a reader to "obtain a greater detail on a specific subject." He named it "stretchtext."[19])

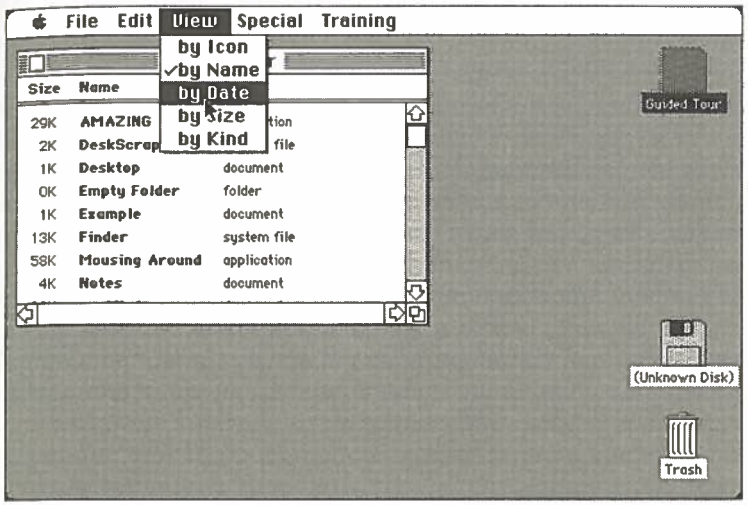
Since new media theory and criticism emerged in the early 1990s, endless texts have been written about interactivity, hypertext, virtual reality, cyberspace, cyberculture, cyborgs, and so on. But I have never seen anybody discuss "view control." And yet this is one of the most fundamental and radical new techniques for working with information and media available to us today. It is used daily by each of us numerous times. "View control," i.e. the abilities to switch between many different views and kinds of views of the same information is now implemented in multiple ways not only in OS, word processors and email clients, but also in all "media processors" (i.e. media editing software): AutoCAD, Maya, After Effects, Final Cut, Photoshop, InDesign, and so on. For instance, in the case of 3D software, it can usually display the model in at least half a dozen different ways: in wireframe, fully rendered, etc. In the case of animation and visual effects software, since a typical project may contain dozens of separate objects each having dozens of parameters, it is often displayed in a way similar to how outline processors can show text. In other words, the user can switch between more and less information. You can choose to see only those parameters which you are working on right now. You can also zoom in and out of the composition. When you do this, parts of the composition do not simply get smaller or bigger—they show less or more information automatically. For instance, at a certain scale you may only see the names of different

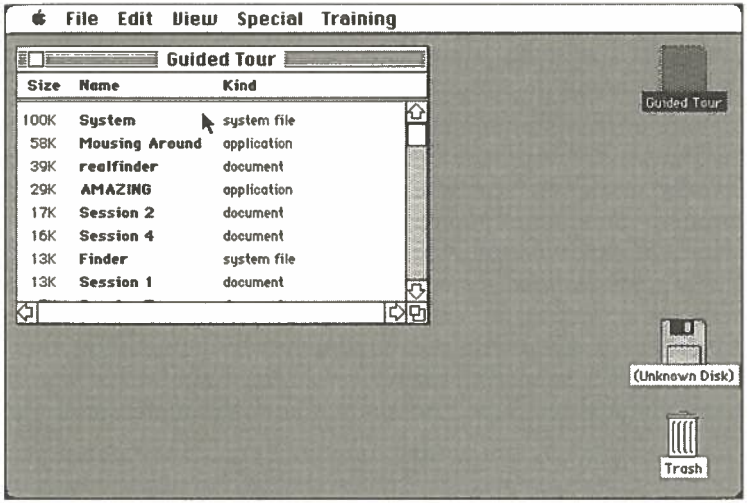
[19] Ted Nelson, "Stretchtext" (Hypertext Note 8), 1967, http://xanadu.com/XUarchive/htn8.tif





View control as implemented in Macintosh System software, 1984. Top: applications, folders, and files in "Guided Tour" floppy disk. Bottom: View of applications, folders, and files sorted by icon.





Top: View of applications, folders, and files sorted by date. Bottom: View of applications, folders, and files sorted by size.

parameters; but when you zoom into the display, the program may also display the graphs which indicate how these parameters change over time.

Let us look at another example—Ted Nelson's concept of hypertext that he developed in the early 1960s (independently but parallel to Engelbart).[20] In his 1965 article *A File Structure for the Complex, the Changing, and the Indeterminate*, Nelson discusses the limitations of books and other paper-based systems for organizing information and then introduces his new concept:

> However, with the computer-driven display and mass memory, it has become possible to create *a new, readable medium*, for education and enjoyment, that will let the reader find his level, suit his taste, and find the parts that take on special meaning for him, as instruction and enjoyment.
>
> Let me introduce the word "hypertext" to mean a body of written or pictorial material interconnected in such a complex way that it could not be conveniently presented or represented on paper.[21]

"A new, readable medium"—these words make it clear that Nelson was not simply interested in "patching up" books and other paper documents. Instead, he wanted to create something distinctively new. But was not hypertext as proposed by Nelson simply an extension of older textual practices such as exegesis (extensive interpretations of holy scriptures such as the Bible, Talmud, Qur'ān), annotations, or footnotes? While such historical precedents for hypertext are often proposed, they mistakenly equate Nelson's proposal with a very limited form in which hypertext is experienced by most people today—i.e., the World Wide Web. As Noah Wardrip-Fruin pointed out, "The Web implemented only

[20] Douglas C. Engelbart, *Augmenting Human Intellect: A Conceptual Framework* (Stanford Research Institute, 1962), http://www.dougengelbart.org/pubs/augment-3906.html. Although the implementation of hypertext in Engelbart's NLS was much more limited than Nelson's concept of hypertext, looking at Engelbart's discussion in *Augmenting Human Intellect* shows that his ideas for new systems for organizing information were at least as rich as Nelson's.

[21] Theodor H. Nelson, "A File Structure for the Complex, the Changing, and the Indeterminate" (1965), in *New Media Reader*, p. 144.

one of many types of structures proposed by Nelson already in 1965—'chunk style' hypertext—static links that allow the user to jump from page to page."[22]

Following the Web implementation, most people today think of hypertext as a body of text connected through one-directional links. However, the terms "links" does not even appear in Nelson's original definition of hypertext. Instead, Nelson talks about new complex interconnectivity without specifying any particular mechanisms that can be employed to achieve it. A particular system proposed in Nelson's 1965 article is one way to implement such a vision, but as his definition implicitly suggests, many others are also possible.

"What kind of structures are possible in hypertext?" asks Nelson in a research note from 1967. He answers his own question in a short but very suggestive manner: "Any."[23] Nelson goes on to explain: "Ordinary text may be regarded as a special case—the simple and familiar case—of hypertext, just as three-dimensional space and the ordinary cube are the simple and familiar special cases of hyperspace and hypercube."[24] (In 2007 Nelson re-stated this idea in the following way: " 'Hypertext'—a word I coined long ago—is not technology but potentially the fullest generalization of documents and literature."[25])

If "hypertext" does not simply mean "links," it also does not only mean "text." Although in its later popular use the word "hypertext" came to refer to linked text, as one can see from the quote above, Nelson included "pictures" in his definition of hypertext.[26] And in the following paragraph, he introduces the terms *hyperfilm* and *hypermedia*:

[22] Noah Wardrip-Fruin, introduction to Theodor H. Nelson, "A File Structure for the Complex, the Changing, and the Indeterminate" (1965), in *New Media Reader*, p. 133.

[23] Ted Nelson, "Brief Words on the Hypertext" (Hypertext Note 1), 1967, http://xanadu.com/XUarchive/htn1.tif

[24] *Ibid.*

[25] Ted Nelson, http://transliterature.org/ (version TransHum-D23, 07.06.17).

[26] In his presentation at the 2004 Digital Retroaction symposium Noah Wardrip-Fruin stressed that Nelson's vision included hypermedia and not only hypertext. Noah Wardrip-Fruin, presentation at Digital Retroaction: a Research Symposium, UC Santa Barbara, September 17–19, 2005, http://dc-mrg.english.ucsb.edu/conference/D_Retro/conference.html

> Films, sound recordings, and video recordings are also linear strings, basically for mechanical reasons. But these, too, can now be arranged as non-linear systems – for instance, lattices – for educational purposes, or for display with different emphasis... The hyperfilm – a browsable or vari-sequenced movie – is only one of the possible hypermedia that require our attention."[27]

Where is hyperfilm today, almost 50 years after Nelson articulated this concept? If we understand hyperfilm in the same limited sense as hypertext is understood today—shots connected through links which a user can click on—it would seems that hyperfilm never fully took off. A number of early pioneering projects—*Aspen Movie Map* (Architecture Machine Group, 1978–9), *Earl King* and *Sonata* (Grahame Weinbren, 1983–5; 1991–3), CD-ROMs by Bob Stein's Voyager Company, and *Wax: Or the Discovery of Television Among the Bees* (David Blair, 1993)—have not been followed up. Similarly, interactive movies and FMV-games created by the video game industry in the first half of the 1990s soon fell out of favor, replaced by 3D games (which offered more interactivity). But if instead we think of hyperfilm in a broader sense, as it was conceived by Nelson—any interactive structure for connecting video or film elements, with a traditional film being a special case—we realize that hyperfilm is much more common today than it may appear. Numerous interactive Flash and HTML5 sites which use video, video clips with markers which allow a user jump to a particular point in a video (for instance, see the videos on TED.com[28]), and database cinema[29] are just some of the examples of hyperfilm today.

Decades before hypertext and hypermedia became the common ways for interacting with information, Nelson understood well what these ideas meant for our well-established cultural practices and concepts. The announcement for his January 5, 1965 lecture at Vassar College talks about this in terms that are even more relevant today than they were then: "The philosophical consequences of all this are very grave. Our concepts of 'reading', 'writing', and 'book' fall apart, and we are challenged to design 'hyperfiles' and write

[27] Nelson, *A File Structure*, p. 144.
[28] www.ted.com (March 8, 2008).
[29] See http://softcinema.net/form.htm

'hypertext' that may have more teaching power than anything that could ever be printed on paper."[30]

These statements align Nelson's thinking and work with artists and theorists who similarly wanted to destabilize the conventions of cultural communication. Digital media scholars extensively discussed parallels between Nelson and French theorists writing during the 1960s—Roland Barthes, Michel Foucault and Jacque Derrida.[31] Others have pointed out close parallels between the thinking of Nelson and literary experiments taking place around the same time, such as works by Oulipo.[32] (We can also note the connection between Nelson's hypertext and the non-linear structure of the films of French filmmakers who set out to question the classical narrative style: *Hiroshima Mon Amour, Last Year at Marienbad*, *Breathless* and others).

How far shall we take these parallels? In 1987 Jay Bolter and Michael Joyce wrote that hypertext could be seen as "a continuation of the modern 'tradition' of experimental literature in print" which includes "modernism, futurism, Dada surrealism, lettrism, the nouveau roman, concrete poetry."[33] Refuting their claim, Espen J. Aarseth has argued that hypertext is not a modernist structure *per se*, although it can support modernist poetics if the author desires this.[34] Who is right? Since this book argues that cultural software turned media into metamedia—a fundamentally new semiotic and technological system which includes most previous media techniques and aesthetics as its elements—I also think that hypertext is actually quite different from modernist literary tradition. I agree with Aarseth that hypertext is indeed much more general than any particular poetics such as modernist ones.

[30] Announcement of Ted Nelson's lecture at Vassar College, January 5, 1965, http://xanadu.com/XUarchive/ccnwwt65.tif
[31] George Landow, ed., *Hypertext: The Convergence of Contemporary Critical Theory and Technology* (The Johns Hopkins University Press, 1991); Jay Bolter, *The writing space: the computer, hypertext, and the history of writing* (Hillsdale, NJ: L. Erlbaum Associates, 1991).
[32] Randall Packer and Ken Jordan, *Multimedia: From Wagner to Virtual Reality* (W. W. Norton & Company, 2001); Noah Wardrip-Fruin and Nick Monford, *New Media Reader* (The MIT Press, 2003).
[33] Quoted in Espen J. Aarseth, *Cybertext: Perspectives on Ergodic Literature* (The Johns Hopkins University Press, 1997), p. 89.
[34] Espen J. Aarseth, *Cybertext*, 89–90.

Indeed, already in 1967 Nelson said that hypertext could support any structure of information including that of traditional texts—and presumably, this also includes different modernist poetics. (Importantly, this statement is echoed in Kay and Goldberg's definition of the computer as a "metamedium" whose content is "a wide range of already-existing and not-yet-invented media.")

What about the scholars who see the strong connections between the thinking of Nelson and modernism? Although Nelson says that hypertext can support any information structure and that this information does not need to be limited to text, his examples and his style of writing show an unmistakable aesthetic sensibility—that of literary modernism. He clearly dislikes "ordinary text." The emphasis on *complexity and interconnectivity* and on *breaking up conventional units for organizing information* such as a page clearly aligns Nelson's proposal for hypertext with the early twentieth-century experimental literature—the inventions of Virginia Woolf, James Joyce, the Surrealists, etc. This connection to literature is not accidental since Nelson's original motivation for his research that led to hypertext was to create a system for handling both the notes for literary manuscripts and those manuscripts themselves. Nelson also already knew about the writings of William Burroughs. The very title of the article—*A File Structure for the Complex, the Changing, and the Indeterminate*—would make the perfect title for an early twentieth-century avant-garde manifesto, as long as we substitute "file structure" with some "ism."

Nelson's modernist sensibility also shows itself in his thinking about new mediums that can be established with the help of a computer. However, his work should not be seen as a simple continuation of modernist tradition. Rather, both his and Kay's research represent the next stage of the avant-garde project. The early twentieth-century avant-garde artists were primarily interested in questioning conventions of established media such as photography, print, graphic design, cinema, and architecture. Thus, no matter how unconventional the paintings that came out from Futurism, Orphism, Suprematism or De Stijl were, their manifestos were still talking about them as paintings—rather than as a new media. In contrast, Nelson and Kay explicitly write about creating new media, not only changing the existing ones. Nelson: "With the computer-driven display and mass memory, it

has become possible to create a new, readable medium." Kay and Goldberg: "It [computer text] need not be treated as a simulated paper book since this is a new medium with new properties."

Another key difference between how modernist artists and pioneers of cultural software approached the job of inventing new media and extending existing ones is captured by the title of Nelson's article I have been already quoting above: "*A File Structure for the Complex, the Changing, and the Indeterminate.*" Instead of a particular modernist "ism," we get a file structure. Cubism, Expressionism, Futurism, Orphism, Suprematism, and Surrealism proposed new distinct systems for organizing information, with each system fighting all others for the dominance in the cultural memesphere. In contrast, Bush, Licklider, Nelson, Engelbart, Kay, Negroponte, and their colleagues created meta-systems that can support many kinds of information structures. Kay called such a system "a first metamedium," Nelson referred to it as hypertext and hypermedia, Engelbart wrote about "automated external symbol manipulation" and "bootstrapping,"—but behind the differences in their visions lay the similar understanding of the radically new potential offered by computers for information manipulation. The prefixes "meta-" and "hyper-" used by Kay and Nelson were the appropriate characterizations for a system which was more than another new medium that could remediate other media in its particular ways. Instead, the new system would be capable of simulating all these media with all their remediation strategies—as well as supporting development of what Kay and Goldberg referred to as new "not-yet-invented media." And of course, this was not all. Equally important was the role of interactivity. The new meta-systems proposed by Nelson, Kay and others were to be used interactively to support the processes of thinking, discovery, decision making, and creative expression. In contrast, the aesthetics created by modernist movements could be understood as "information formatting" systems—to be used for selecting and organizing information into fixed presentations that are then distributed to the users, not unlike PowerPoint slides. Finally, at least in Kay's and Nelson's vision, the task of defining new information structures and media manipulation techniques—and, in fact, new media as a whole—was given to the user, rather than being the sole province of the designers. This decision had far-reaching consequences for shaping contemporary culture. Once

computers and programming were democratized enough, many creative people started to focus on creating these new structures and techniques rather than using the existing ones to make "content." Since the end of 2000, extending the computer metamedium by writing new software, plugins, programming libraries and other tools became the new cutting-edge type of cultural activity – giving a new meaning to McLuhan's famous formula "the medium is the message."

Today a typical article in computer science or information science will not be talking about inventing a "new medium" as a justification for research. Instead, it is likely to refer to previous work in some field or sub-field of computer science such as "knowledge discovery," "data mining," "semantic web," etc. It can also refer to existing social and cultural practices and industries—for instance, "e-learning," "video game development," "collaborative tagging," or "massively distributed collaboration." In either case, the need for new research is justified by a reference to already established or popular practices—academic paradigms which have been funded, large-scale industries, and mainstream social routines which do not threaten or question the existing social order. This means that practically all of computer science research which deals with media—web technologies, media computing, hypermedia, human-computer interfaces, computer graphics, and so on—is oriented towards "mainstream" media usage.

In other words, either computer scientists are trying to make more efficient the technologies already used in media industries (video games, web search engines, film production, etc.) or they are inventing new technologies that are likely to be used by these industries in the future. The invention of new mediums for its own sake is not something which anybody is likely to pursue, or get funded. From this perspective, the software industry and business in general is often more innovative than academic computer science. For instance, social media applications (Wikipedia, Flickr, YouTube, Facebook, del.icio.us, Digg, etc.) were not invented in the academy; nor were HyperCard, QuickTime, HTML, Photoshop, After Effects, Flash, or Google Earth. This was no different in previous decades. It is, therefore, not accidental that the careers of both Ted Nelson and Alan Kay were spent in the industry and not the academy: Kay worked for and was a fellow at Xerox PARC, Atari, Apple and Hewlett-Packard; Nelson was a consultant and

a fellow at Bell Laboratories, Datapoint Corporation, Autodesk; both were also associated with Disney.

Why did Nelson and Kay find more support in industry than in academia for their quest to invent new computer media? And why is the industry (by which I simply mean any entity which creates the products which can be sold in large quantities, or monetized in other ways, regardless of whether this entity is a large multinational company or a small start-up)—more interested in innovative media technologies, applications, and content than computer science? The systematic answer to this question will require its own investigation. Also, what kinds of innovations each modern institution can support changes over time. But here is one brief answer: modern business thrives on creating new markets, new products, and new product categories. Although the actual development of such new markets and products is always risky, it is also very profitable. This was already the case in the previous decades when Nelson and Kay were supported by Xerox, Atari, Apple, Bell Labs, Disney, etc. In the 2000s, following the globalization of the 1990s, all areas of business embraced innovation to an unprecedented degree; this pace quickened around 2005 as companies fully focused on competing for new consumers in China, India, and other "emerging" economies. Around the same time, we saw a similar increase in the number of innovative products in the IT industry: open APIs of leading Web 2.0 sites, daily announcements of new web services, locative media applications, new innovative products such as iPhone, new paradigms in imaging such as HDR and non-destructive editing, the beginnings of a "long tail" for software, open source hardware, and so on.

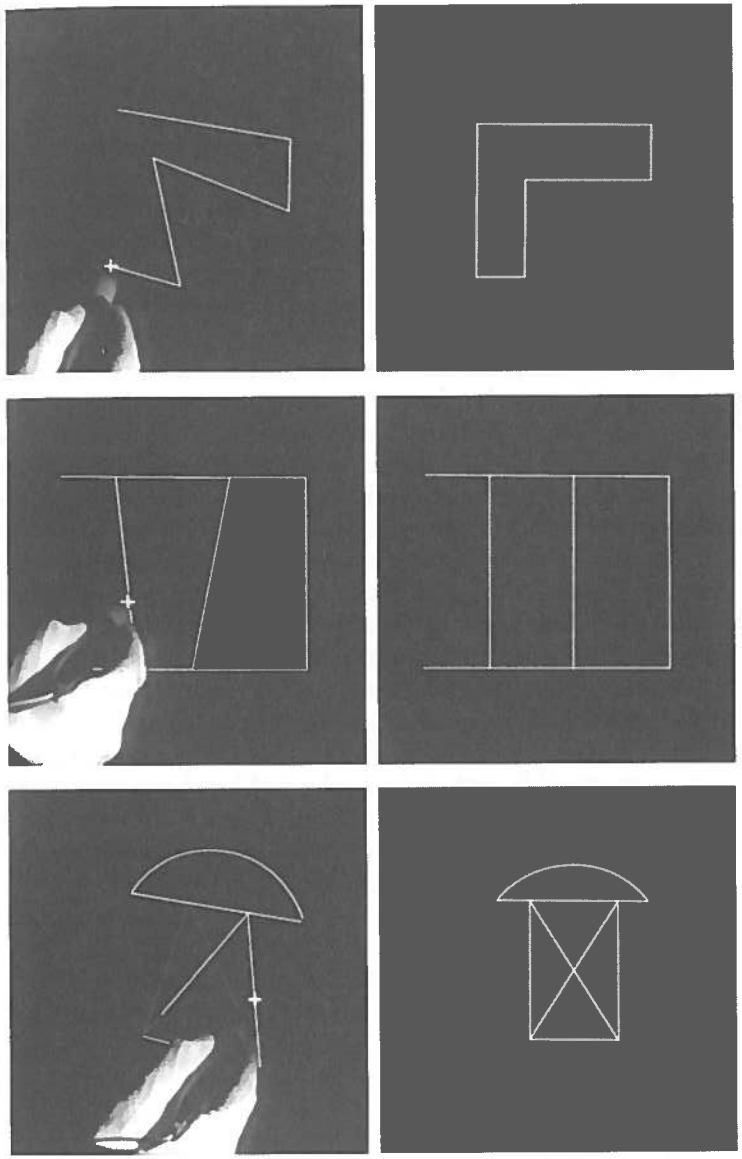
As we can see from the examples we have analyzed, the aim of the inventors of computational media—Engelbart, Nelson, Kay and the people who worked with them—was not simply to create accurate simulations of physical media. Instead, in every case the goal was to create "a new medium with new properties" which would allow people to communicate, learn, and create in new ways. So while today the content of these new media may often look the same as that of its predecessors, we should not be fooled by this similarity. The newness lies not in the content but in the software tools used to create, edit, view, distribute, and share this content. Therefore, rather than only looking at the "output" of software-based cultural practices, we need to consider

software itself—since it allows people to work with media in a number of historically unprecedented ways. So while on the level of appearance computational media indeed often remediate (i.e. represent) previous media, the software environment in which this media "lives" is very different.

Let me add two more examples. One is Ivan Sutherland's *Sketchpad* (1962). Created by Sutherland as a part of his PhD thesis at MIT, Sketchpad deeply influenced all subsequent work in computational media (including that of Kay) not only because it was the first interactive media authoring program but also because it made it clear that computer simulations of physical media can add many exciting new properties to the media being simulated. Sketchpad was the first software that allowed its users to interactively create and modify line drawings. As Noah Wardrip-Fruin pointed out, it "moved beyond paper by allowing the user to work at any of 2000 levels of magnification—enabling the creation of projects that, in physical media, would either be unwieldy large or require detail work at an impractically small size."[35] Sketchpad similarly redefined graphical elements of a design as objects which "can be manipulated, constrained, instantiated, represented ironically, copied, and recursively operated upon, even recursively merged.'[36] For instance, if the designer defined new graphical elements as instances of a master element and later made a change to the master, all these instances would also change automatically.

Another new property, which perhaps demonstrated most dramatically how computer-aided drafting and drawing were different from their physical counterparts, was Sketchpad's use of constraints. In Sutherland's own words, "The major feature which distinguishes a Sketchpad drawing from a paper and pencil drawing is the user's ability to specify to Sketchpad mathematical conditions on already drawn parts of his drawing which will be automatically satisfied by the computer to make the drawing take the exact shape desired."[37] For instance, if a user drew a few lines, and then gave the appropriate command, Sketchpad automatically

[35] Noah Wardrip-Fruin, introduction to "Sketchpad. A Man-Machine Graphical Communication System," in *New Media Reader*, 1963, p. 109.
[36] *Ibid.*
[37] Ivan Sutherland, "Sketchpad. A Man-Machine Graphical Communication System," *Proceedings of the AFIPS Spring Joint Computer Conference,* Detroit,



Frames from Sketchpad demo video illustrating the program's use of constraints. Left column: a user selects parts of a drawing. Right column: Sketchpad automatically adjusts the drawing. (The captured frames were edited in Photoshop to show the Sketchpad screen more clearly.)

moved these lines until they were parallel to each other. If a user gave a different command and selected a particular line, Sketchpad moved the lines in such a way so they would parallel to each other and perpendicular to the selected line.

Although we have not exhausted the list of new properties that Sutherland built into Sketchpad, it should be clear that this first interactive graphical editor was not only simulating existing media. Appropriately, Sutherland's 1963 paper on Sketchpad repeatedly emphasizes the new graphical capacities of his system, marveling how it opens new fields of "graphical manipulation that has never been available before."[38] The very title given by Sutherland to his PhD thesis foregrounds the novelty of his work: *Sketchpad: A man-machine graphical communication system*. Rather than conceiving of Sketchpad as simply another medium, Sutherland presents it as something else—a communication system between two entities: a human and an intelligent machine. Kay and Goldberg later also foregrounded this communication dimension, referring to it as "a two-way conversation" and calling the new "metamedium" "active."[39] (We can also think of Sketchpad as a practical demonstration of the idea of "man-machine symbiosis" by J. C. R. Licklider applied to image making and design.[40])

My last example comes from the software development that at first sight may appear to contradict my argument: paint software. Surely, the applications which simulate in detail the range of effects made possible with various physical brushes, paint knives, canvases, and papers are driven by the desire to recreate the experience of working within an existing medium rather than the desire to create a new one? Wrong. In 1997 an important computer graphics pioneer Alvy Ray Smith wrote a memo titled *Digital Paint Systems: Historical Overview*.[41] In this text Smith (who himself had a background in art) makes an important distinction between

Michigan, May 21–3, 1963, pp. 329–46; in *New Media Reader*, Noah Wardrip-Fruin and Nick Montfort (eds).

[38] *Ibid.*, p. 123.

[39] Kay and Goldberg, "Personal Dynamic Media," 394.

[40] J. C. R. Licklider, "Man-Machine Symbiosis," *IRE Transactions on Human Factors in Electronics*, vol. HFE-1, March 1960, pp. 4–11, in *New Media Reader*, eds. Noah Wardrip-Fruin and Nick Montfort.

[41] Alvy Ray Smith, *Digital Paint Systems: Historical Overview* (Microsoft Technical Memo 14, May 30, 1997). http://alvyray.com/

digital paint programs and *digital paint systems*. In his definition, "A digital paint *program* does essentially no more than implement a digital simulation of classic painting with a brush on a canvas. A digital paint *system* will take the notion much farther, using the "simulation of painting" as a familiar metaphor to seduce the artist into the new digital, and perhaps forbidding, domain." (Emphasis in the original). According to Smith's history, most commercial painting applications, including Photoshop, fall into the paint system category. His genealogy of paint systems begins with Richard Shoup's SuperPaint, developed at Xerox PARC in 1972–3.[42] While SuperPaint allowed the user to paint with a variety of brushes in different colors, it also included many techniques not possible with traditional painting or drawing tools. For instance, as described by Shoup in one of his articles on SuperPaint, "Objects or areas in the picture may be scaled up or down in size, moved, copied, overlaid, combined or changed in color, and saved on disk for future use or erased."[43]

Most important, however, was the ability to grab frames from video. Once loaded into the system, such a frame could be treated as any other image—that is, an artist could use all of SuperPaint's drawing and manipulation tools, add text, combine it with other images, etc. The system could also translate what appeared on its screen back into a video signal. Accordingly, Shoup is clear that his system was much more than a way to draw and paint with a computer. In a 1979 article, he refers to SuperPaint as a new "videographic medium."[44] In another article published a year later, he refines this claim: "From a larger perspective, we realized that the development of SuperPaint signaled the beginning of the synergy of two of the most powerful and pervasive technologies ever invented: digital computing and video or television."[45]

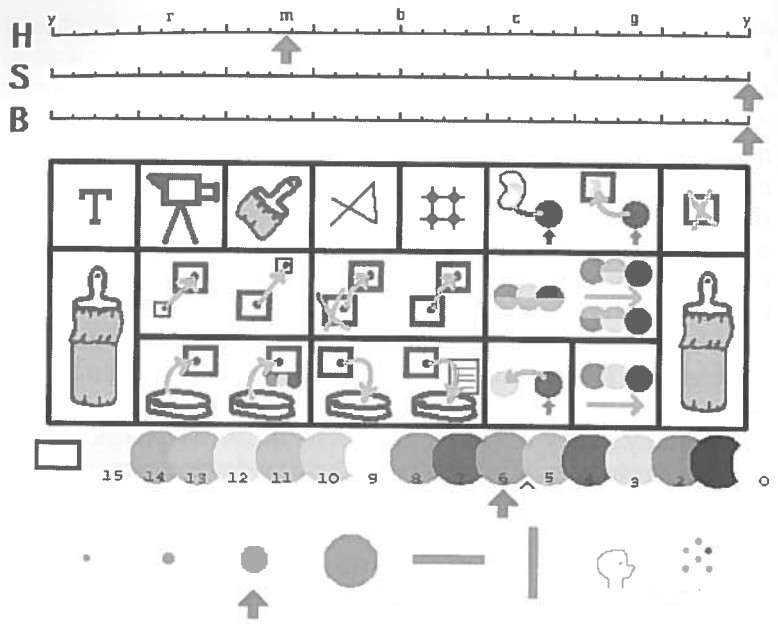
This statement is amazingly perceptive. When Shoup was writing this in 1980, computer graphics were used in television

[42] Richard Shoup, "SuperPaint: An Early Frame Buffer Graphics Systems," IEEE Annals of the History of Computing 23, issue 2 (April–June 2001), p. 32–7, http://www.rgshoup.com/prof/SuperPaint/Annals_final.pdf; Richard Shoup, "SuperPaint...The Digital Animator," *Datamation* (1979), http://www.rgshoup.com/prof/SuperPaint/Datamation.pdf.

[43] Shoup, "SuperPaint...The Digital Animator," p. 152.

[44] *Ibid.*, p. 156.

[45] Shoup, "SuperPaint: An Early Frame Buffer Graphics System," p. 32.



SuperPaint menu, 1975.

broadcasts just a handful of times. And while in the next decade their use became more common, only in the middle of the 1990s did the synergy Shoup predicted truly became visible. As we will see in the chapter on After Effects below, the result was a dramatic reconfiguration not just of the visual languages of television but of all visual techniques invented by humans up to that point. In other words, what began as a new "videographic medium" in 1973 had eventually changed all visual media.

But even if we forget about SuperPaint's revolutionary ability to combine graphics and video, and discount its new tools such resizing, moving, copying, etc., we are still dealing with *a new creative medium* (Smith's term). As Smith pointed out, this medium is the *digital frame buffer*,[46] a special kind of computer memory

[46] Alvy Ray Smith, "Digital Paint Systems: An Anecdotal and Historical Overview," *IEEE Annals of the History of Computing*. 2011, http://accad.osu.edu/~waynec/history/PDFs/paint.pdf

designed to hold images represented as an array of pixels (today a more common name is *graphics card*). An artist using a paint system is modifying pixel values in a frame buffer—regardless of what particular operation or tool s/he is employing at the moment. This opens up a door to all kinds of new image creation and modification operations, which follow different logic than physical painting. The telling examples of this can be found in a paint system called Paint developed by Smith in 1975–6. In Smith's own words, "Instead of just simulating painting a stroke of constant color, I extended the notion to mean 'perform any image manipulation you want under the pixels of the paintbrush."[47] Beginning with this conceptual generalization, Smith added a number of effects which still used a paintbrush tool but actually no longer referred to painting in a physical world. For instance, in Paint "any image of any shape could be used as a brush." In another example, Smith added "'not paint' that reversed the color of every pixel under the paintbrush to its color complement." He also defined 'smear paint' that averaged the colors in the neighborhood of each pixel under the brush and wrote the result back into the pixel." And so on. Thus, the instances where the paintbrush tool behaved more like a real physical paintbrush were just particular cases of a much larger universe of new behaviors made possible in a new medium.

The permanent extendibility

As we saw, Sutherland, Nelson, Engelbart, Kay, and other pioneers of computational media have added many previously non existent properties to media that they have simulated in a computer. The subsequent generations of computer scientists, hackers, and designers added many more properties—but this process is far from finished. And there is no logical or material reason why it will ever be finished. It is the "nature" of computational media that it is open-ended and that new techniques are continuously being invented.

To add new properties to physical media requires modifying its physical substance. But since computational media exists as

[47] *Ibid.*, p. 18.

software, we can add new properties or even invent new types of media by simply changing existing or writing new software. Or by adding plug-ins and extensions, as programmers have been doing it with Photoshop and Firefox, respectively. Or by putting existing software together. (For instance, starting in 2006, thousands of people extended the capacities of mapping media by creating software mashups which combine the services and data provided by Goggle Maps, Flickr, Amazon, other sites, and media uploaded by users.)

In short, *"new media" is "new" because new properties (i.e., new software techniques) can always be easily added to it.* Put differently, in industrial (i.e. mass-produced) media technologies, "hardware" and "software" were one and the same thing. For example, the book pages were bound in a particular way that fixed the order of pages. The reader could not change this order nor the level of detail being displayed *à la* Engelbart's "view control." Similarly, the film projector combined hardware and what we now call a "media player" software into a single machine. In the same way, the controls built into a twentieth-century mass-produced camera could not be modified at the user's will. And although today the users of a digital camera similarly cannot easily modify the hardware of their camera, as soon as they transfer the pictures into a computer they have access to endless number of controls and options for modifying their pictures via software.

In the nineteenth and twentieth centuries the normally rigid industrial media was fluid in two situations. First, when a new media was being first developed: for instance, the invention of photography in the 1820s–1840s. Second, when artists would systematically experiment with and "open up" already industrialized media—such as the experiments with film and video during the 1960s that came to be called "Expanded Cinema."

What used to be separate moments of experimentations with media during the industrial era became the norm in a software society. In other words, the computer legitimizes experimentation with media. Why is this so? What differentiates a modern digital computer from any other machine—including industrial media machines for capturing and playing media—is separation of hardware and software. It is because an endless number of different programs performing different tasks can be written to run on the same type of machine, that that machine—i.e. a digital

computer—is used so widely today. Consequently, the constant invention of new (and modification of existing) media software, is simply one example of this general principle. In its very structure computational media is "avant-garde" since it is constantly being extended and thus redefined.

If in modern culture "experimental" and "avant-garde" were opposed to normalized and stable, this opposition largely disappears in software culture. And the role of the media avant-garde is performed no longer by individual artists in their studios but by a variety of players, from very big to very small—from companies such as Microsoft, Adobe, and Apple to independent programmers, hackers, and designers.

But this process of continual invention of new algorithms does not just move in any direction. If we look at contemporary media software—CAD, computer drawing and painting, image editing, word processors—we will see that most of their fundamental principles were already developed by the generation of Sutherland and Kay. In fact the very first interactive graphical editor—Sketchpad—already contains most of the genes, so to speak, of contemporary graphics applications. As new techniques continue to be invented they are layered over the foundations that were gradually put in place by Sutherland, Engelbart, Kay, and others in the 1960s and 1970s.

Of course we are not dealing here only with the history of ideas. Various social and economic factors—such as the dominance of the media software market by a handful of companies or the wide adoption of particular file formats — also constrain possible directions of software evolution. Put differently, today software development is an industry and as such it is constantly balancing between stability and innovation, standardization and exploration of new possibilities. But it is not just any industry. New programs can be written and existing programs can be extended and modified (if the source code is available) by anybody who has programming skills and access to a computer, a programming language and a compiler. In other words, today software is fundamentally malleable in a way that twentieth-century industrially produced objects were not. (The emergence of consumer 3D printing and the "open hardware" movement promise to bring such flexibility to physical objects as well, but it will be a while before you can print a whole ready-to-drive-car on your home 3D printer.)

Although Turing and Von Neumann formulated this fundamental extendibility of software in theory, its contemporary practice—hundreds of thousands of people daily involved in extending the capabilities of computational media—is a result of a long historical development. This development took us from the few early room-sized computers, which were not easy to reprogram to a wide availability of cheap computers and programming tools decades later. This democratization of software development was at the core of Kay's vision. Kay was particularly concerned with how to structure programming tools in such a way that would make development of media software possible for ordinary users. For instance, at the end of the 1977 article I have already extensively quoted, he and Goldberg write, "We must also provide enough already-written general tools so that a user need not start from scratch for most things she or he may wish to do."

Comparing the process of continuous media innovation via new software to the history of earlier, pre-computational media reveals a new logic at work. According to a commonplace idea, when a new medium is invented it first closely imitates already existing media, before discovering its own language and aesthetics. Indeed, the first Gutenberg Bible closely imitated the look of the handwritten manuscripts; early films produced in the 1890s and 1900s mimicked the presentational format of theatre by positioning the actors on the invisible shallow stage and having them face the audience. Slowly, printed books developed a different way of presenting information; similarly cinema also developed its own original concept of narrative space. Through repetitive shifts in points of view presented in subsequent shots, the viewers were placed inside this space—thus literally finding themselves inside the story.

Can this logic apply to the history of computer media? As theorized by Turing and Von Neumann, the computer is a general-purpose simulation machine. This is its uniqueness and its difference from all other machines and previous media. This means that the idea that a new medium gradually finds its own language cannot apply to computer media. If this were true it would go against the very definition of a modern digital computer. This theoretical argument is supported by practice. The history of computer media so far has been not about arriving at some standardized language—as, for instance, happened with cinema—but rather about the gradual expansion of uses, techniques, and possibilities. Rather than arriving at a particular language, we are gradually discovering that the computer can speak more and more languages.

If we are to look more closely at the early history of computer media—for instance, the way we have been looking at Kay's ideas and work in this text—we will discover another reason why the idea of a new medium gradually discovering its own language does not apply to computer media. The systematic practical work on making a computer simulate and extend existing media (Sutherland's Sketchpad, the first interactive word processor developed by Engelbart's group, etc.) came after computers had already been put to multiple uses—performing different types of calculations, solving mathematical problems, controlling other machines in real time, running mathematical simulations, simulating some aspects of human intelligence, and so on. (We should also mention the work on SAGE by MIT Lincoln Laboratory which, by the middle of the 1950s, had already established the idea of interactive communication between a human and a computer via a screen with a graphical display and a pointing device. In fact, Sutherland developed Sketchpad on a TX-2, the new version of a larger computer MIT constructed for SAGE.) Therefore, when the generation of Sutherland, Nelson, and Kay started to create "new media," they built it on top, so to speak, of what computers were already known to be capable of. Consequently they added new properties into physical media they were simulating right away. This can be very clearly seen in the case of Sketchpad. Understanding that one of the roles a computer can play is that of a problem solver, Sutherland built in a powerful new feature that never before existed in a graphical medium—satisfaction of constraints. To rephrase this example in more general terms, we can say that rather than moving from an imitation of older media to finding its own language, computational media was from the very beginning speaking a new language.

In other words, the pioneers of computational media did not have the goal of making the computer into a 'remediation machine" which would simply represent older media in new ways. Instead, knowing well the new capabilities provided by digital computers, they set out to create fundamentally new kinds of media for expression and communication. These new media would use as their raw "content" the older media which already served humans well for hundreds and thousands of years—written

language, sound, line drawings and design plans, and continuous tone images (i.e. paintings and photographs). But this does not compromise the newness of new media. Computational media uses these traditional human media simply as building blocks to create previously unimaginable representational and information structures, creative and thinking tools, and communication options.

Although Sutherland, Engelbart, Nelson, Kay, and others developed computational media on top of already existing developments in computational theory, programming languages, and computer engineering, it would be incorrect to conceive the history of such influences as only going in one direction—from already existing and more general computing principles to particular techniques of computational media. The inventors of computational media had to question many, if not most, already established ideas about computing. They have defined many new fundamental concepts and techniques of how both software and hardware function, thus making important contributions to hardware and software engineering. A good example is Kay's development of Smalltalk, which for the first time systematically established a paradigm of object-oriented programming. Kay's rationale to develop this new programming language was to give a unified appearance to all applications and the interface of the PARC system and, even more importantly, to enable its users to quickly program their own media tools. (According to Kay, an object-oriented illustration program written in Smalltalk by a particularly talented 12-year-old girl was only a page long.[48]) Subsequently the object-oriented programming paradigm became very popular and object-oriented features have been added to most popular languages such as C.

Looking at the history of computer media and examining the thinking of its inventors makes it clear that we are dealing with the opposite of technological determinism. When Sutherland designed Sketchpad, Nelson conceived hypertext, Kay programmed a paint program, and so on, each new property of computer media had to be imagined, implemented, tested, and refined. In other words, these characteristics did not simply come as an inevitable result of a meeting between digital computers and modern media.

[48] Alan Kay, *Doing with Images Makes Symbols* (University Video Communications, 1987), videotaped lecture, http://archive.org/details/AlanKeyD1987/

Computational media had to be invented, step-by-step. And it was invented by people who were looking for inspiration in modern art, literature, cognitive and education psychology, and theory of media as much as technology. For example, Kay recalls that reading McLuhan's *Understanding Media* led him to a realization that a computer can be a medium rather than only a tool.[49] Accordingly, the opening section of Kay and Goldberg's article is called "Humans and Media," and it does read like media theory. But this is not a typical theory that only describes the word, as it currently exists. Similar to Marx's analysis of capitalism in his works, here the analysis is used to create a plan for action for building a new world—in this case, enabling people to create new media.

But the most important example of such non-deterministic development is the invention of the modern interactive graphical human-computer interface itself by Sutherland, Engelbart, Kay and others. None of the key theoretical concepts of modern computing as developed by Turing and Von Neumann called for an interactive interface. In the late 1940s and 1950s the MIT Lincoln Laboratory developed interactive graphical computers used in SAGE—the control centers created around the US to collect information from radar stations and coordinate a counter-attack. But the SAGE interface was designed for very particular tasks and it had no effect on the development of commercial computing. It did, however, lead to a new smaller machine: the TX-2, used by young students at MIT (including Sutherland) to explore what can be done with an "interactive computer"—i.e. a computer which had a visual display. Some students started to create interactive games including the famous Spacewar (1960). Sutherland was one of these students who were exploring the possibilities of visual interactive computing using the TX-2. He went to create Sketchpad (his Ph.D. thesis) which influenced other pioneers of cultural computing in the 1960s including Kay. But the theoretical road that led from SAGE to modern GUI through PARC was a very long one.

According to Kay, the key step for him and his group was to start thinking about computers as a medium for learning,

[49] Alan Kay, "User Interface: A Personal View," p. 192–3.

experimentation, and artistic expression which can be used not just by adults but also by "children of all ages."[50] Kay was strongly influenced by the theory of the cognitive psychologist Jerome Bruner. Bruner developed his theory by redefining the ideas of Jean Piaget who postulated that children go through a number of distinctive intellectual stages as they develop: a kinesthetic stage, a visual stage, and a symbolic stage. But while Piaget thought that each stage only exists for a particular period during a child's development only to be completely replaced by a new stage, Bruner suggested that separate mentalities that correspond to these stages continue to exist as the child grows. That is, the mentalities do not replace each other but are added. Bruner gave slightly different names to these different mentalities: enactive, iconic, and symbolic. While each mentality has developed at different stages of human evolution, they continue to co-exist in an adult.

Kay's interpretation of this theory was that a user interface should appeal to all these three mentalities. In contrast to a command-line interface, which is not accessible for children and forces the adult to use only symbolic mentality, the new interface should also make use of emotive and iconic mentalities. Kay also drew on a number of studies on creativity in math, science, music, art and other areas which suggested that initial creative work is done mostly in iconic mentality and also in enactive.[51] This provided additional motivation for the idea that if computers were to function as a dynamic medium for learning and creativity they should allow their users to think not only through symbols but also through actions and images.

Following Kay's interpretation of Bruner's work, the group at PARC mapped Bruner's theory of multiple mentalities into the interface technologies in the following way. *Mouse* activates enactive mentality (know where you are, manipulate). *Icons and windows* activate iconic mentality (recognize, compare, configure.) Finally, *Smalltalk programming language* allows for the use

[50] Alan Kay, "A Personal Computer for Children of All Ages," *Proceedings of the ACM National Conference*, Boston, 1972, http://www.mprove.de/diplom/gui/kay72.html

[51] Alan Kay, "User Interface: A Personal View," p. 195.

of symbolic mentality (tie together long chains of reasoning, abstract.)[52]

In actual use, a contemporary GUI involves constant interplay between different mentalities. You use a mouse to move around the screen as though it is a physical space and point at screen objects. All objects are represented by visual icons. You double-click on an icon to activate it or, if it is a folder icon, to examine its contents. This can be interpreted as an equivalent of picking up and examining a physical object in a real world. After a folder window opens, you may switch between different views, looking at the data as icons and alternatively as a list, then sort the list in different ways to examine file names, creation dates and other symbolic information (i.e. text). If you did not find the files you were looking for, you may then use a search function to search the whole computer—possibly defining multiple options and carefully choosing the search terms (symbolic mentality). As these examples demonstrate, the user is constantly switching between different mentalities using whatever works best at a given moment.

But in addition to the general interface principles, other key techniques that were developed by Kay's group can also be understood as enabling the use of different mentalities in combination with each other. For instance, the user interface developed at PARC was the first to run on a bit-mapped display—which meant not only giving users the ability to move the pointer and open multiple windows but also to write simulation programs in Smalltalk which could display their results visually right on the screen. By making a change in the code a user would be able to see the visual result of this change in the image produced by the program. Today this ability is fundamental to computer use in all areas of science (in particular, the use of interactive visualization and data analysis software). And of course, we should not forget about all the media editors created at PARC: a paint program, an illustration program, a music editor, etc. These media editors gave the users the ability to switch between different mentalities in a way not available in the physical media. For instance, the objects in the animation program could be drawn by hand or by writing code in Smalltalk. As Kay and Goldberg point out, "The control of the animation

[52] *Ibid.*, p. 197.

could be easily done from a Smalltalk simulation. For example, an animation of objects bouncing in a room is most easily accomplished by a few lines of Smalltalk code that express the class of bouncing objects in physical terms."[53]

In defining this new type of user interface, Kay and his collaborators simultaneously created a radically new type of media. If we are to agree with Bruner's theory of multiple mentalities and Kay's interpretation of this theory, we should conclude that the new computational media that he helped to invent can do something no previous media can—activate our multiple mentalities which all play a role in learning and creativity, allowing a user to employ whatever works best at any given moment and to rapidly switch between them as necessary. This may explain the success and popularity of the GUI, which, forty years after its invention, continues to dominate our interaction with computers. People prefer it not because it is "easy" or "seamless" or "intuitive." It is successful because it was designed to help them think, discover, and create new concepts using not just one type of mentality but all of them together. In short, while many HCI experts and designers continue to believe that the ideal human-computer interface should be invisible and get out of the way to let users do their work, looking at the theories of Kay and Goldberg that were behind GUI design gives a very different way of understanding an interface's identity. Kay and his colleagues at PARC have conceived GUI as a *medium* designed in its every detail to facilitate learning, discovery, and creativity.

Given the overall emphasis of information society on constant innovation, continuous learning, and creativity, it is only appropriate that as this society was coming into existence, a new medium was being invented specifically to facilitate these needs. In 1973 Daniel Bell published his highly influential *The Coming of Post-Industrial Society*; right around that time at PARC Kay, Goldberg, Chuck Thacker, Dan Ingalls, Larry Tesler, and other members of the Learning Research Group created the paradigm of modern computing. Or rather, they reinvented the computer—from a fast calculator that can only work on tasks articulated

[53] Kay and Goldberg, "Personal Dynamic Media," p. 399.

beforehand to an interactive support system for thinking and discovery. In short: from a tool to a metamedium.

Unfortunately, when GUI became the commercially successful paradigm following the success of Apple's Mac computers, introduced in 1984, the intellectual origins of GUI were forgotten. Instead, GUI was justified using a simplistic idea that since computers are unfamiliar to people, we should help them by making interface intuitive by making it mimic something users are already well familiar with—the physical world outside of a computer (which in reality was an office environment with folders, desks, printers, etc.) Surprisingly, even in recent years– when "born digital" generations were already using computer devices even before they ever set foot in an office—this idea was still used to explain GUI. For example, Apple's iPhone Human Interface guidelines (March 2010) advise developers: "When possible, model your application's objects and actions on objects and actions in the real world. This technique especially helps novice users quickly grasp how your application works. Folders are a classic software metaphor. People file things in folders in the real world, so they immediately understand the idea of putting data into folders on a computer."[54] The irony of this statement is that these Interface guidelines are also aimed at the developers of iPad—which clearly represents yet another step in migration from the world of physical print to all-digital environment. It is as though we are asked to remember and cherish the older media—and erase it at the same time.

The computer as a metamedium

As we have established, the development of computational media runs contrary to previous media history. But in a certain sense, the idea of a new media gradually discovering its own language actually does apply to the history of computational media after all. And just as with printed books and cinema, this process took a

[54] http://developer.apple.com/iphone/library/documentation/UserExperience/Conceptual/MobileHIG/PrinciplesAndCharacteristics/PrinciplesAndCharacteristics.html#//apple_ref/doc/uid/TP40006556-CH7-SW1 (April 5, 2010).

few decades. When the first computers were built in the middle of the 1940s, they could not be used as media for cultural representation, expression, and communication. Slowly, through the work of Sutherland, Engelbart, Nelson, Papert, and others in the 1960s, the ideas and techniques were developed that made computers into a cultural machine. One could create and edit text, make drawings, move around a virtual object, etc. And finally, when Kay and his colleagues at PARC systematized and refined these techniques and put them under the umbrella of a GUI (making computers accessible to multitudes) a digital computer finally was given its own language—in cultural terms. In short, only when a computer became a cultural medium—rather than merely a versatile machine—could it be so used.

Or rather, it became something that no other media had been before. For what had emerged was not yet another media, but as Kay and Goldberg insist in their article, something qualitatively different and historically unprecedented. To mark this difference, they introduce a new term—"metamedium."

This metamedium is unique in a number of different ways. One of them I have already discussed in detail—it can represent most other media while augmenting them with many new properties. Kay and Goldberg also name other properties that are equally crucial. The new metamedium is "active—it can respond to queries and experiments—so that the messages may involve the learner in a two-way conversation." For Kay who was strongly interested in children and learning, this property was particularly important since, as he puts it, it "has never been available before except through the medium of an individual teacher." [55] Further, the new metamedium can handle "virtually all of its owner's information-related needs." (I have already discussed the consequence of this property above.) It can also serve as "a programming and problem solving tool" and "an interactive memory for the storage and manipulation of data."[56] But the property that is the most important from the point of view of media history is that *the computer metamedium is simultaneously a set of different media and a system for generating new media tools and new types of media.* In other words, a computer

[55] Kay and Goldberg, "Personal Dynamic Media," p. 394.
[56] *Ibid.*, p. 393.

can be used to create *new tools for working with the media types it already provides as well as to develop new not-yet-invented media.*

In the opening to his book *Expressive Processing*, Noah Wardrip-Fruin perfectly articulates this "meta-generative" specificity of computers:

> A computer can simulate a typewriter—getting input from the keyboard and arranging pixels on the screen to shape the corresponding letters—but it can also go far beyond a typewriter, offering many fonts, automatic spelling correction, painless movement of manuscript sections (through simulations of "cut" and "paste"), programmable transformations (such as "find and replace"), and even collaborative authoring by large, dispersed groups (as with projects like Wikipedia). This is what modern computers (more lengthily called "stored-program electronic digital computers") are designed to make possible: the continual creation of new machines, opening new possibilities, through the definition of new sets of computational processes.[57]

Using the analogy with print literacy, Kay motivates this property in this way: "The ability to 'read' a medium means you can *access* materials and tools generated by others. The ability to write in a medium means you can *generate* materials and tools for others. You must have both to be literate."[58] Accordingly, Kay's key effort at PARC was the development of the Smalltalk programming language. All media editing applications and the GUI itself were written in Smalltalk. This made all the interfaces of all applications consistent, facilitating quick learning of new programs. Even more importantly, according to Kay's vision, Smalltalk would allow even novice users to write their own tools and define their own media. In other words, all media editing applications that would be provided with a computer, were to serve also as examples, inspiring users to modify them and to write their own applications.

[57] Noah Wardrip-Fruin, *Expressive Processing: Digital Fictions, Computer Games, and Software Studies* (The MIT Press, 2009).
[58] Alan Kay, "User Interface: A Personal View," in *The Art of Human-Computer Interface Design*, ed. Brenda Laurel (Reading, MA, Addison-Wesley, 1990), p. 193. The emphasis is in the original.

Accordingly, the large part of Kay and Goldberg's paper is devoted to description of software developed by the users of their system: "an animation system programmed by animators", "a drawing and painting system programmed by a child," "a hospital simulation programmed by a decision-theorist," "an audio animation system programmed by musicians", "a musical score capture system programmed by a musician", "electronic circuit design by a high school student." As can be seen from this list, (which corresponds to the sequence of examples in the article), Kay and Goldberg deliberately juxtaposed different types of users—professionals, high school students, and children—in order to show that everybody could develop new tools using the Smalltalk programming environment.

The sequence of examples also strategically juxtaposes media simulations with other kinds of simulations in order to emphasize that simulation of media is only a particular case of the computer's general ability to simulate all kinds of processes and systems. This juxtaposition of examples gives us an interesting way to think about computational media. Just as a scientist may use simulation to test different conditions and play different what/if scenarios, a designer, a writer, a musician, a filmmaker, or an architect working with computer media can quickly "test" different creative directions in which the project can be developed as well as see how modifications of various "parameters" affect the project. The latter is particularly easy today since the interfaces of most media editing software not only explicitly present these parameters but also simultaneously give the user the controls for their modification. For instance, when the Formatting Palette in Microsoft Word shows the font used by the currently selected text, it is displayed in a column next to all other fonts available. Trying different fonts is as easy as scrolling down and selecting the name of a new font.

Giving users the ability to write their own programs was a crucial part of Kay's vision for the new "metamedium" he was inventing at PARC. According to Noah Wardrip-Fruin, Engelbart's research program was focused on a similar goal: "Engelbart envisioned users creating tools, sharing tools, and altering the tools of others."[59]

[59] Noah Wardrip-Fruin, introduction to Douglas Engelbart and William English, "A Research Center for Augmenting Human Intellect" (1968), *New Media Reader*, p. 232.

Unfortunately, when in 1984 Apple shipped Macintosh, which was to become the first commercially successful personal computer modeled after the PARC system, it did not have an easy-to-use programming environment. HyperCard, written for Macintosh in 1987 by Bill Atkinson (who was one of PARC's alumni), gave users the ability to quickly create certain kinds of applications—but it did not have the versatility and breadth envisioned by Kay. Only more recently, as the general computer literacy has widened and many new high-level programming languages have become available—Perl, PHP, Python, JavaScript, etc.—have more people started to create their own tools by writing software. A good example of a contemporary programming environment, very popular among artists and designers and which, in my view, is close to Kay's vision, is Processing.[60] Built on top of the Java programming language, Processing features a simplified programming style and an extensive library of graphical and media functions. It can be used to develop complex programs and also to quickly test ideas. Appropriately, the official name for Processing projects is sketches.[61] In the words of Processing inventors and main developers Ben Fry and Casey Reas, the language's focus is "on the 'process' of creation rather than end results."[62] Another popular programming environment that similarly enables quick development of media projects is Max/MSP and its successor PD—both developed by Miller Puckette.

At the end of the 1977 article that served as the basis for our discussion in this chapter, Kay and Goldberg summarize their arguments in the phrase—which in my view is the best formulation we have had so far—of what computational media is artistically and culturally. They call the computer *"a metamedium"* whose content is *"a wide range of already-existing and not-yet-invented media."* In another article published in 1984 Kay unfolds this definition. As a way of concluding this chapter, I would like to quote this longer definition which is as accurate and inspiring today as it was when Kay wrote it:

> It [a computer] is a medium that can dynamically simulate the details of any other medium, including media that cannot exist

[60] www.processing.org
[61] http://www.processing.org/reference/environment/
[62] http://wiki.processing.org/w/FAQ

physically. It is not a tool, though it can act like many tools. It is the first *metamedium*, and as such it has degrees of freedom for representation and expression never before encountered and as yet barely investigated.[63]

[63] Alan Kay, "Computer Software," *Scientific American* (September 1984), p. 52. Quoted in Jean-Louis Gassée, "The Evolution of Thinking Tools," in *The Art of Human-Computer Interface Design*, p. 225.

CHAPTER TWO

Understanding metamedia

"It [the electronic book] need not be treated as a simulated paper book since this is a new medium with new properties."
Kay and Goldberg, "Personal Dynamic Media," 1977

Today *Popular Science*, published by Bonnier and the largest science+tech magazine in the world, is launching Popular Science+ — the first magazine on the Mag+ platform, and you can get it on the iPad tomorrow...What amazes me is that you don't feel like you're using a website, or even that you're using an e-reader on a new tablet device — which, technically, is what it is. *It feels like you're reading a magazine.*" (emphasis is in the original.)
"Popular Science+," posted on April 2, 2010.
http://berglondon.com/blog/2010/04/02/popularscienceplus/

The building blocks

I started putting this book together in 2007. Today is April 3, 2010, and I am editing this chapter. Today is also an important day in the history of media computing (which started exactly forty years ago with Ivan Sutherland's Sketchpad)—Apple's iPad tablet computer first went on sale in the US on this date. During the years I was writing and editing the book, many important developments made Alan Kay's vision of a computer as the "first metamedium" more real—and at the same time more distant.