The Evolution and Future of the Graphical User Interface in Personal Computing

Jennifer Brown

Hardware and Operating Systems

Dr. McDonald Davis

16 April 2003

Outline

I. The graphical user interface (GUI) has grown from an experiment in ease-of-use to a bona-fide part of operating system and application design. As the use of personal computers becomes more ubiquitous in society, the GUI will continue to develop until it is the primary means with which humans and computers interact.

II. History of the GUI
   a. Pre-Consumer Development & Input Devices
      i. Early conceptual discussion of the GUI
      ii. Ivan Sutherland & Sketchpad
      iii. GRAIL and the RAND Tablet
      iv. SmallTalk & other graphical development languages
   b. Early Consumer Development
      i. Xerox Parc & the Alto
      ii. Apple Lisa and Macintosh Development
      iii. Microsoft Windows Development
      iv. Other failed GUI operating systems
   c. Windows, MacOS and X Windows – the modern GUIs
      i. Evolution of Windows
      ii. Evolution of MacOS
      iii. Evolution of X Windows Systems

III. Current GUI interfaces and strategies
   a. Basic Components
      i. Pointing & positioning
      ii. Windowing System
      iii. Menus
      iv. Icons
      v. Metaphoric and Symbolic relationships
   b. Best practices for GUI Development
      i. Immediate feedback
      ii. Direct Manipulation
      iii. WYSWYG
      iv. Universal Access considerations

IV. The future of the GUI
   a. New input devices
      i. Chording glove
      ii. Biofeedback pointer
      iii. Handwriting recognition
      iv. Gesture recognition
      v. Eye tracking
   b. Attentive User Interfaces
   c. Intelligent Metaphoric relationships

The Evolution and Future of the Graphical User Interface in Personal Computing

Nothing illustrates the importance of graphical interfaces on ease-of-use than a visit to the canned goods aisle of a grocery store. Every can contains a label with a graphical picture of the can's contents—the canned tomatoes are wrapped with a paper label featuring a full-color depiction of a perfect tomato. Choosing the right product is as easy as looking for the picture of what you want. If each can were labeled the same—a white paper label with the contents printed in black letters—it would be considerably harder to distinguish and choose the canned tomatoes over another canned product like canned peaches.

From the earliest caveman drawings, to the canned foods aisle of the grocery store one thing remains the same—the human brain is wired to quickly understand graphical images. This concept is the basis behind the widespread use of the Graphical User Interface (GUI) in computing. The GUI offers users a visual way of interacting with a computer system, and have become ubiquitous in nearly all levels of personal computing. Initially nothing more than an experiment in Human Computer Interaction (HCI), the GUI has grown into a bona-fide part of operating system and application design. Modern PC operating systems rely heavily on graphical interfaces, and the study and development of these interfaces is a growing area of computer science. Early studies of human computer interaction provided the foundation for the concepts that would become the GUI, then in the 1960s and 1970s an explosion in laboratory expiraments in computer science led to the creation of the input devices commonly associated with the GUI and the graphical architecture of modern GUIs. In the 1980s and 1990s the first consumer graphical interfaces came to market—these interfaces would introduce a new way of computing that is still in use today. The GUI will continue to develop until it is the primary means with which humans and computers interact.

Early Theoretical Foundations for HCI

In hindsight, the development of graphical interfaces seems a natural and inevitable evolution in computing, but the path to the modern GUI operating systems is one of the most dramatic stories in the personal computer industry. While the GUI is still considered a fairly modern development in computer science, its origins can be tracked back to the birth of modern computers. In 1945 Vannevar Bush touted a concept called Memex—a computer that would pull information for users based on what they selected on screen. (Tuck 2001).

While theoretical, Bush's idea is the foundation for solving on of the largest issues in HCI—the difference between how humans and computers "think." Computers operate in the realm of syntax—or the structure and order of language, while humans tend to think in terms of semantics—or meaning. Designing an interface for a computer system requires the ability to bridge the sequential and literal order in which a computer processes data, and the non-sequential and symbolic methods that most humans use to problem solve. "Computers work in the domain of syntax, but actions are about semantics. A useful system embeds correct actions in a syntax that is easy to grasp, precisely because it carries the semantics" (Hodgson 2002).

From the user's perspective there are a five key elements that contribute to bridging the gap between syntax and semantics and create what is known as a user-friendly operating environment. These five elements are:

1. Learnability – The ability for a user to adjust to the computer's operating environment quickly.

2. Efficiency—The ability for a user to see a significant productivity boost when using the system.
3. Memorability—The ability for one to leave the system for a period of time and return to the operating environment without having to learn to use it over again.
4. Errors—The operating environment should be free from bugs and errors that would hinder productivity or frustrate users.
5. Satisfaction—The individual opinion of the user that the environment benefits them and is easy-to-use. (Hilbert and Redmiles 2000, 388)

In the 1940's when Bush was theorizing on Memex, his concepts met the criteria for a balance between syntax and semantics, but technology had not advanced to the level that would allow him to create a material version of his Memex concept. It wouldn't be until the 1960s that Bush's ideas and concepts would begin to take physical form, and the concept of the GUI would begin to develop.

Early Studies and Implementations of the Graphical Interface

In the 1960's technology had advanced to the point where the graphical user interface began to emerge from Bush's conceptual ideas into full-fledged physical reality. New developments in hardware and software technology allowed for the birth of the first graphical interfaces to appear. This period also gave birth to many of the basic design concepts in graphical interfaces.

In 1962 a visionary named Ivan Sutherland, working in MIT's Lincoln Labratories, introduced a program called Sketchpad. The program used a light pen, and allowed Sutherland to create and edit engineering drawings on a 9-inch CRT screen in a What You See, Is What You Get (WYSWYG) interface (Cringley 2003).

> Sketchpad pioneered the concepts of graphic computing, including memory structures to store objects, rubber banding of lines, the ability to zoon in and out on the display, and the ability to make perfect lines, corners and joints. This was the first GUI […] long before the term was coined. (Sun Microsystems 1999)

Sketchpad introduced such concepts as WYSWYG, the use of a pointing device for input and direct manipulation of the objects on the screen. Sutherland's work coincided with some significant work by another father of the GUI—Douglas Engelbart. Also in 1962, working out of Stanford University Englebart developed a device that he called the "X-Y Position Indicator." This device was a wooden box with mechanical wheels that could be used to manipulate the position of a pointer on a screen. Englebart's device was the first mouse. Englebart believed that his device could be used to to drive what he called a "graphical windowed interface." In 1968. Englebart developed NLS (oNLine System) which he referred to as "a windowed GUI." NLS featured a windowed interface that was manipulated by the mouse, hyperlinked media, and even video teleconferencing (Tuck 2001).

Englebart and Sutherland's work in the 1960s set the foundations for the way in which the GUI would be used in modern computers. Their work introduced three basic concepts that still drive GUIs today—those are the need for some kind of input device for pointing, and direct manipulation of the graphical objects on the display.

In order for a human to interact with any computer, whether via a GUI or some other interface the human needs some form of input device. The first input device most commonly associated with human computer interaction is the keyboard. Most modern keyboards have the basic letters and number keys as well as a number pad, arrow keys and function and control keys or key combinations. In a addition to inputting text, a keyboard can be useful for operating in a GUI environment—such things as arrow keys and control-key combinations can manipulate the objects on the screen. Despite issues with layout and ergonomics with the QWERTY keyboard, "the momentum built up from over a century of use has guaranteed its dominance despite the problems with both layout and basic keyboard shape" (Rosenberg 1998, 22).

The second most common input device for graphical interfaces is the mouse. Mechanical mice usually consist of a plastic case that houses a small rubber ball, which turns two wheels within the ball's housing, these wheels correspond to an X or Y axis on the computer screen. After an incremental rotation of one of the wheels the mouse would send this information to registers available to the computer or directly to the computer's memory. Each rotation would move the mouse's representational pointer on-screen an amount equal to the increment that the wheel has turned. The mouse is one of the best pointing devices created for interaction with graphical computer output:

> In addition to its simplicity and low cost, the mouse has the advantage that the user need not pick it up in order to use it. The house simply sits on the table surface until they need it. This makes the mouse an efficient device for pointing. (Fischer and Lee 1999)

While the mouse and keyboard are the most common types of input devices for graphical interfaces, there are several others that have been developed for use with GUIs. The decendants of Sutherland's light pen are still in use. In the mid-1960s a company called RAN developed the RAND tablet, which is the predecessor to the modern drawing tablet—which uses a tablet and pen combination for input. There are also touch pads and touch screens, roller balls—which act like a mechanical mouse turned upside down, and now even gyroscopic pointers that allows users to hold a mouse-like device in the air and manipulate an onscreen pointer on the same X-Y axis as a mouse. The input devices associated with a GUI interface all serve one purpose, and that is to point to items on the screen or position an item on the screen into a new area. This is the first basic component of any GUI (Fischer and Lee 1999).

Another major concept that developed during the early days of the GUI is the idea of direct manipulation, or a what you see is what you get environment. According to Johnathan Hodgson, “the central ideas of direct manipulation are visibility of objects and actions of interest; rapid, reversible, incremental actions; and replacement of command-language syntax by direct manipulation of the object of interest” (2002). The key to developing a good GUI is to think in visual terms rather than sequential terms. Nearly every modern GUI operating system follows the rules of direct manipulation, for instance, when copying a file from a removable disk to a hard drive in a GUI environment typically involves clicking on file’s picture and moving that file’s picture over the destination for the file. A dialog box usually appears that indicates that the copy process is occurring and shows the progress of the process, when the process is complete, the file’s picture appears in the desired location. By contrast, in a command-line interface the feedback is not instant or visual, to copy the same file requires typing in a copy command with the source and destination, once the user presses enter there’s no indication that the copy process has taken place until the user types in another command to see if the file is in the correct directory (Fischer and Lee 1999).

The use of a pointing device and the concept of direct manipulation set the foundation for a user-friendly computing environment, but in the late 1960s these concepts remained the sole domain of academics and research and development labs. It wouldn’t be until the 1970s, when the personal computer brought computer technology into the homes, that the GUI truly developed and evolved into the environment we use today.

Bringing the GUI to Market: The Foundation of Look and Feel

The race to bring a GUI-based operating system to market is one of the most engaging dramas in the history of personal computing—so engaging in fact that the cable channel TNT produced a prime-time movie called the "Pirates of Silicon Valley" in 2001, which told the story of Steve Jobs and Bill Gates and the development of MacOS and Windows. While the story has become the stuff of legend, the effort that went in to bringing the GUIs we use today to market is full of effort, hard work and sound HCI principles, and probably a little less drama than a prime time movie.

The story begins at the Xerox Palo Alto Research Center (PARC), where during the 1970s, some of the best minds in computer science created a think-tank that not only gave birth to the basics for the graphical user interface that is in use on nearly every personal computer today, but also developed everything from Ethernet networking, the laser printer, and even the optical mouse. In 1974 some of the best minds at PARC developed the Xerox Alto computer, which was intended to be a sort of personal computer. The Alto featured a bit-mapped interface, graphical applications including a WYSIWYG text editor and a painting program, and true WYSIWYG printing, but unfortunately "was about the size of a small Volkswagen" (Tuck 2000).

Undaunted the minds at PARC started developing the Xerox Star. The Star featured many of the elements that would become the basic components of every GUI today (Tuck 2000). The Star introduced the idea of using metaphors as a language for a graphical interface. The Star created a graphical environment that used how a person works in an office as the basis for the pictures, icons and task organization of the system. While in computer syntax files might be located in a directory, on the Star files were stored in folders—just like a file cabinet. The Star called the working space on the screen the "computer desktop" (Tuck 2000). The Star's concepts of Windows, Icons, Menus and Pointers (WIMP) would become the standard by which nearly every GUI, from personal computer interfaces to user interfaces for POS terminals in grocery stores are based (Tuck 2000).

Besides their genius, the researchers at PARC had two other things that worked to their advantage—timing and location. At the time that PARC was developing these technologies the personal computer was coming into the market and landing in homes across the country. The easy-to-understand and use interfaces that Xerox developed were the perfect fit for the computer leaving the halls of science, research and industry and landing in the living rooms. The other advantage that Xerox PARC had was its location in Silicon Valley, the heart of the personal computer revolution. In the 1970s the Silicon Valley area contained a close-knit community of computer developers and hobbiests known as the Homebrew Computer Club, and word of new developments and discoveries moved quickly across the valley. It wasn't long until Steve Jobs and Steve Wozniak, the inventors of the Apple I and II computer heard about the GUI that Xerox had developed (Cringely 2002).

The legend that has grown around this states that Jobs arranged a visit to PARC where he walked in, looked around and walked out and set Apple to work on developing the GUI he saw at Xerox. But as Mike Tuck states in his online article *The Real History of the GUI,* "the idea of Jobs coming in like a kid touring Epcot [sic] with a tape recorder hidden under his shirt is mistaken." Instead as Steve Wozniak, the co-founder of Apple Computer explains on his website:

> Steve Jobs made the case to Xerox PARC execs directly that they had great technology but that Apple knew how to make it affordable enough to change the world. This was very open. In the end, Xerox got a large block of Apple stock for sharing the technology. (Tuck 2000).

Regardless, in December 1979 Apple toured the Xerox facility, and walked away and began development on the first successful consumer-level GUI operating system.[1] In January 1983 Apple released the Lisa, and the GUI-based operating system was finally available to any one in the masses that had a spare $10,000 to spend on a personal computer. The Lisa was an extremely powerful computer for its time, featuring a 5 Mhz Motorola MC68000 processor and 512K RAM. The Lisa featured a full-GUI interface using the WIMP components, a suite of business software, communication and networking abilities, and was considered a major breakthrough in personal computing. The price however, deterred buyers, and the Lisa was an efficient workhorse of a failure (Tuck 2000).

Undaunted, Apple began development on a more price-friendly system called the Macintosh. The Macintosh debuted in January 1984 and featured the first version of MacOS. The development effort for the MacOS operating system went well beyond the effort that was put on the development of the Lisa. Apple worked with psychologists, graphic artists, teachers and even elementary school students on the development for the MacOS GUI. The final release was a stronger candidate than the Lisa, featuring graphical interface features like pull-down menus and click-and-drag functionality (Tuck 2000). The Macintosh also had a better price point than the

[1] It should be noted that VisiCorp released the first consumer GUI operating system, VisiOn was released prior to the Lisa. But VisiOn faded out quickly without any applications to run on it.

Lisa, the original Macintosh computer featuring an 8Mhz 32-bit Motorola 68000 processor, and 128K RAM sold for $2500 (Cringley 2002).

Apple, having experience with the success of its Apple II line of computers because of the spreadsheet application VisiCalc, understood the importance of the killer application to the success of any computer platform. To ensure that the Macintosh would survive, Apple looked to a little company called Microsoft to create that killer application. The agreement benefited both companies—Apple got the killer apps Microsoft Word and Microsoft Excel for the Macintosh, and Microsoft, by developing these products became intimately familiar with the workings of a GUI operating system (Tuck 2000).

Microsoft's first attempts at developing and releasing a GUI-based operating system failed miserably. Windows 1.0—which was really nothing other than a shell that ran over DOS—was released in November 1985, and did not stir much interest. Despite the fact that Microsoft was developing and programming several of the major killer apps that made the Macintosh GUI a success, Windows 2.0—released in December 1987 flopped on the market because of a lack of programs that ran on it. (Tuck 2000).

Microsoft finally got the formula right with the release of Windows 3.0 in May 1990. This time Microsoft ensured that it released several applications specifically tailored for Windows along with the release of the GUI. Subsequent upgrades to Windows 3.0, including "Windows for Workgroups"—a GUI specifically tailored for business users—propelled Windows in the marketplace and ensured its viability. While Windows 3.0 was a GUI, it was not an operating system, instead, like its predecessors it was only a shell that ran on top of MS-DOS. It wasn't until Microsoft released WindowsNT in 1993 that Windows truly became a GUI-based operating system (Tuck 2000).

Today, the GUI-based operating system is ubiquitous in personal computing. Nearly everyone who uses a personal computer uses either Windows, MacOS or X Windows. The GUI has matured to the point where windows, icons and office desktop metaphors are standard across nearly all GUI-based interfaces, and thouroughly entreched in the way that people use computers. While interfaces are maturing and and developing, the WIMP model of Windows, Icons, Menus and Pointers that Xerox pioneered has come to dominate the graphical interface.

Nearly every modern GUI relies on all the elements of WIMP. The first element is Windows. The windowing system "is for sharing a computer's graphical display presentation resources among multiple applications, at the same time." The windowed interface displays each running application in a discrete area, often referred to as a window. In some windowing systems, such as Microsoft Windows each instance of an application has its own window, whereas in others, such as Mac OS X, each application has its own window, regardless of the number of instances running on the machine. A windowing system uses a window manager to track the location and status of each open window on the system, this windowing system may also monitor other components of the GUI, such as menus and icons (Fischer and Lee 2003l).

The next component of WIMP is the icon. The icon owes its roots to Xerox PARC engineer David Canfield Smith. "According to Smith[,] he adopted the term from the Russian Orthodox Church where an icon is more than an image because it embodies the properties of what if represents." (Fischer and Lee 2003). In the modern GUI systems an icon is typically a bitmapped image that it placed on the screen—the image typically either represents a real object, such as picture of a small CD to represent the contents of a CD-ROM or a symbolic image, such as a globe to represent a network browser. In computing terms, icons are very simplistic, GUIs often offer command words, dialog boxes or other cues when an icon is selected byu the user (Fischer, Lee, GUIGI5.html).

The third basic component of the WIMP is the menu. Menus allows a user to select a command from a list of several available commands for the window he or she is using. Menus in modern GUIs are composed of several different parts including the menubar, which is an area that houses the menu items available for use with that particular window, and menu items, which are the unqiue commands available to the user. There are also several different types of menus including heriarchal menus that smaller menus branching off of the main menu items located in the menu bar (Fischer and Lee 2003). Heriarcal menus are still fairly new, but are well-entrneched in modern operating systems. Despite this there is still some debate over their usefulness:

> Hierarchal Menus are a hot topic among UI people. Selecting a menu item is esy. You only have to be accurate in one direction (vertically)—and you can be sloppy in the horizontal direction, without 'falling off' the menu. (Fischer and Lee 2003)

Other new developments in menu design include Pop-Up menus, which are not contained in a manu bar, and typically do not offer commands like typical menu items. Usually Pop-Up menu items are selections that a user can make. In good GUI design, Pop-Up menus are usually well marked with visual feedback cues such as down-arrows. Another new menu feature are hidden or contextual menus, which are menus that offer no visual cue to their use or existence (Fichser and Lee 2003). Right-clicking on the desktop in Mac OS X or Windows XP will bring up one of these hidden menus.

The final element of WIMP is the pointer. The pointer moves on screen in response to one of the available input devices attached to the computer. This is one element of GUI that hasn't changed since Sutherland's Sketchpad.

The Future of the GUI

In the nearly twenty years since since the MacOS was introduced the GUI-based operating system has grown into a mature area of computer science. The future of the GUI is ensured as long as humans need to interact with a computer that outputs information on a screen, but the development of the GUI is far from over. Researchers and scientists are continuing to try to refine the way in which humans and computers interact, and the metaphors and methods that are in use today may be surpassed by future improvements to input devices, functionality metaphors and even the concept of the interface itself.

One of the recent trends in personal computing is a movement towards more portability—computer users now have personal digital assistants, digital telephones, laptop computers, tablet PCs, and an array of countless other computing devices in use every day. This trend towards users carrying several unique computer products has produced some new challenges in graphical interface design.

The first challenge deals with the need for portability with input devices. While the QWERTY keyboard and mouse work well for a desktop and laptop systems, they become inefficient, as a computing device gets smaller and more portable (Rosenberg 1998, 16). To accommodate these smaller devices the pointing device that's used to indicate a selection in a GUI needs to evolve past the mouse.

Several new input devices have been developed recently to address the need for some form of pointer in a portable GUI environment. With handwriting recognition a small pen-like device is either used on a screen similar to writing on a document, or is used on a specific area of the portable device where the user writes one letter over another in a set area (Rosenberg 1998, 30). Handwriting recognition technology, however, is still a developing area. Most devices—such as the Palm Pilot PDA—that currently employ the pen-input method do not actually recognize handwriting, but rather recognize a series of shorthand characters. One advantage to pen-based input devices are that the pen works as both the keyboard for text input, and a pointing device for interaction with the GUI. The drawback is that the device must allocate an area on the screen large enough for writing.

Another input device solution for portability is eye tracking, which uses a sensor to track the movement of the eyes in reaction to the content of a screen. The user selects the items he or she wants to manipulate simply by staring at that item. There are some drawbacks to eye-tracking:

> This can have problems in a windows-style or hypertext GUI. The user cannot look at a hot area such as an icon, button or menu for too long without activating it. […] Eye tracking also tends to have problems with location small targets because of involuntary eye motion. (Rosenberg 1998, 43).

Although still in development two other input items that may affect the future of the graphical interface are the chording glove and the biofeedback pointer. The chording glove is an input device that is worn like a glove, but allows the user to move their fingers to tap out all the items that would normally be available on a conventional keyboard (Rosenberg 1998, 28). The second device, the biofeedback pointer requires no traditional pointing device, but rather would involve connecting small sensors to the skin and using bioelectric feedback to move a pointer in screen (Rosenberg 2000, 43). In studies, after some training most users are able to use the biofeedback pointer with a 70 percent degree of accuracy (Rosenberg 1998, 48).

The second challenged posed by the movement towards more portable computing is the accuracy and usefulness of the desktop metaphor which seems to be synonymous with graphical interfaces. The personal computer has evolved well beyond its original intention, and now does everything from allowing someone to edit home movies to running all the accounting functions of a small business. Recently there has been an explosion in the use of computers for creative work. The desktop metaphor of files, folders and a working area called the desktop is entrenched in the computing world, but may not be the best metaphor for the graphical interfaces of the future (Marcus 1994).

The use of the desktop metaphor for interaction may be hampering the use of computers in the creative process. In the desktop-based GUI most tasks and activities are sequentially organized and the basis for interaction with a document is that it is being created in a series of sequential steps. Recent studies have shown that when performing creativity-based tasks, most people do not use the sequential model that these systems are based on. Instead "most users follow Schön's theory of Reflection-In-Action. In other words they frequently tried something, then backed out to evaluate the chance, and flip-flopped between the two options until they decided a direction" (Terry and Mynatt 2002).

Several interface improvements have been suggested to improve the user interface for the use of computers for creative tasks, and these concepts stretch the sequential concept of the desktop to a more exploratory environment. Some examples of ways to improve this environment would be to reduce the need for users to save multiple copies of a creative piece at different stages of the process, and instead save all these generations into one document so the user can go back to any previous state if needed. Live comparison of changes is also critical—perhaps by showing a side-by-side comparison of before and after. This will save the user the time to repeat the undo and redo commands. Interfaces should not use generic previews, but rather offer true previewing. For instance if a user highlights several words in a document and mouses over the icon to make that section italic, a tool-tip could appear that shows exactly what that part of the document would look like with that change (Terry and Mynatt 2002).

Finally, there is a new area in interface design that threatens to turn the desktop GUI on its head. This new concept for interface design is called the Attentitive User Interface (AUI), and its based on the concept that most computer users now rely on several devices. Roel Vertegaal describes the impetus that is pushing the creating of the AUI:

> If there was a Moore's Law for user interfaces, it would state that the number of computers per user will double every two years. […] Researchers are becoming aware of the fact that uer attention is a limited resource that must be conserved. User interface designers and engineers are beginning to design computing devices that negotiate, rather than impose […on] the user. (Vertegaal 2003).

The AUI would not only require that the various computing devices that occupy our lives use similar interface elements, but would also require that the way in which we interact with them change significantly. For instance, the mouse and keyboard would not be enough to indicate to indicate what items currently have the user's attention and interest (Vertegaal 2003).

The AUI will be another evolution in HCI, but the GUI will remain the most important way that humans and computers interact for the foreseeable future. Even the AUI would not function without the ability to display its information in a way that a user can quickly digest and understand. Humans are well-tuned to deal with pictures and metaphoric relationships, and this is why the GUI has been such a powerful tool in aiding the growth of personal computers. Nearly sixty years ago Bush's Memex gave us the starting blocks from which to build the concepts of human computer interaction. Englebart's mouse and Sutherland's Sketchpad proved that direct manipulation of objects increased the ease-of-use for any computer system. With these tools, Xerox PARC brought the GUI to fruition, introducing the core concepts of Windows, Icons, Menus and Pointers—the components of nearly every GUI system in use today. It took Apple to bring the GUI to market and prove its viability, and Microsoft to turn it into a household item. Now, as the GUI reaches maturity systems are being constantly refined and explored. The GUI has met new challenges with more computing devices per user, the portable computing environment and the changing nature of general computer use. Despite this the GUI has survived, because as the adage goes: an icon is worth a thousand command-line instructions.

Bibliography

Cringely, Robert X. *Welcome to Triumph of the Nerds!* [resource website on-line]. Alexandria, Virginia: PBS Online, 2003, accessed 29 March 2003. Available from http://www.pbs.org/nerds/; Internet.

Fischer, Andy, and Kevin Lee. *Graphical User Interface* [website on-line]. .: George Mason University, 1999, accessed 12 April 2003. Available from http://cne.gmu.edu/itcore/userinterface/GUIHistory1.html; Internet.

Hilbert, David M., and David F. Redmiles. *Extracting usability information from user interface events* [journal on-line]. New York, New York: ACM Press, 2000, pp 384-421 accessed 29 March 2003. Available from http://doi.acm.org/10.1145/371578.371593; Internet.

Hodgson, Ph.D, Jonathan. *The Design of Graphic User Interfaces* [personal website on-line]. Philadephia, PA: Saint Joseph's University, 2002, accessed 29 March 2003. Available from http://www.sju.edu/~jhodgson/gui/guihome.html; Internet.

Marcus, Aaron. *Managing metaphors for advanced user interfaces* [journal on-line]. New York, New York: ACM Press, 1994, accessed 29 March 2003. Available from http://doi.acm.org/10.1145/192309.192317; Internet.

Rosenberg, Robert. "Computing without mice and keyboards: Text and graphic input devices for mobile computing." Ph.D. diss., University College, London, 1998.

Sun Microsystems. *Ahead of the Pack: Ivan E. Sutherland* [article on-line]. Palo Alto, CA: Sun Microsystems, 1999, accessed 8 April 2003. Available from http://www.sun.com/960710/feature3/ivan.html; Internet.

Terry, Michael, and Elizabeth D. Mynatt. *Recognizing creative needs in user interface design* [journal on-line]. New York, New York: ACM Press, 2002, accessed 29 March 2003. Available from http://doi.acm.org/10.1145/581710.581718; Internet.

Tuck, Mike. *The Real History of the GUI* [article on-line]. SitePoint Pty. Ltd., 2001, accessed 12 April 2003. Available from http://www.sitepoint.com/article/511; Internet.

Vertegaal, Roel. *Attentitive User Interfaces: Introduction* [journal on-line]. New York, New York: ACM Press, 2003, accessed 29 March 2003. Available from http://doi.acm.org/10.1145/636772.636794; Internet.