Strategic Directions in Human-Computer Interaction

BRAD MYERS

Carnegie Mellon University, Pittsburgh, PA ⟨bam@cs.cmu.edu⟩

JIM HOLLAN

University of New Mexico, Albuquerque, NM ⟨hollan@cs.unm.edu⟩

ISABEL CRUZ ET AL.[1]

Tufts University, Medford, MA ⟨ifc@cs.brown.edu⟩

1. RETROSPECTIVE

Human-computer interaction (HCI) is the study of how people design, implement, and use interactive computer systems and how computers affect individuals, organizations, and society. This encompasses not only ease of use but also new interaction techniques for supporting user tasks, providing better access to information, and creating more powerful forms of communication. It involves input and output devices and the interaction techniques that use them; how information is presented and requested; how the computer's actions are controlled and monitored; all forms of help, documentation, and training; the tools used to design, build, test, and evaluate user interfaces; and the processes that developers follow when creating interfaces.

This report describes the historical and intellectual foundations of HCI and then summarizes selected strategic directions in human-computer interaction research. Previous important reports on HCI directions include the results of the 1991 [Sibert and Marchionini 1993] and 1994 [Strong 1994] NSF studies on HCI in general, and the 1994 NSF study on the World-Wide Web [Foley and Pitkow 1994].

1.1 Importance of HCI

Users expect highly effective and easy-to-learn interfaces and developers now realize the crucial role the interface plays. Surveys show that over 50% of the design and programming effort on projects is devoted to the user interface portion [Myers and Rosson 1992]. The human-computer interface is critical to the success of products in the marketplace, as well as the safety, usefulness, and pleasure of using computer-based systems.

There is substantial empirical evi-

[1] This article is based on the results of the Human-Computer Interaction Working Group of the ACM Workshop on Strategic Directions in Computing Research, and was written by: Steve Bryson (NASA Ames Research Center), Dick Bulterman (CWI), Tiziana Catarci (University of Rome), Wayne Citrin (University of Colorado Boulder), Isabel Cruz (Tufts University, Co-Chair and Editor), Ephraim Glinert (RPI), Jonathan Grudin (University of California Irvine), Jim Hollan (University of New Mexico, Editor, Yannis Ioannidis (University of Wisconsin-Madison), Rob Jacob (Tufts University), Bonnie John (Carnegie Mellon University), David Kurlander (Microsoft Research), Brad Myers (Carnegie Mellon University, Co-Chair and Editor), Dan Olsen (Carnegie Mellon University), Randy Pausch (University of Virginia), Stuart Shieber (Harvard University), Ben Shneiderman (University of Maryland College Park), John Stasko (Georgia Tech), Gary Strong (NSF), and Kent Wittenburg (Bellcore).

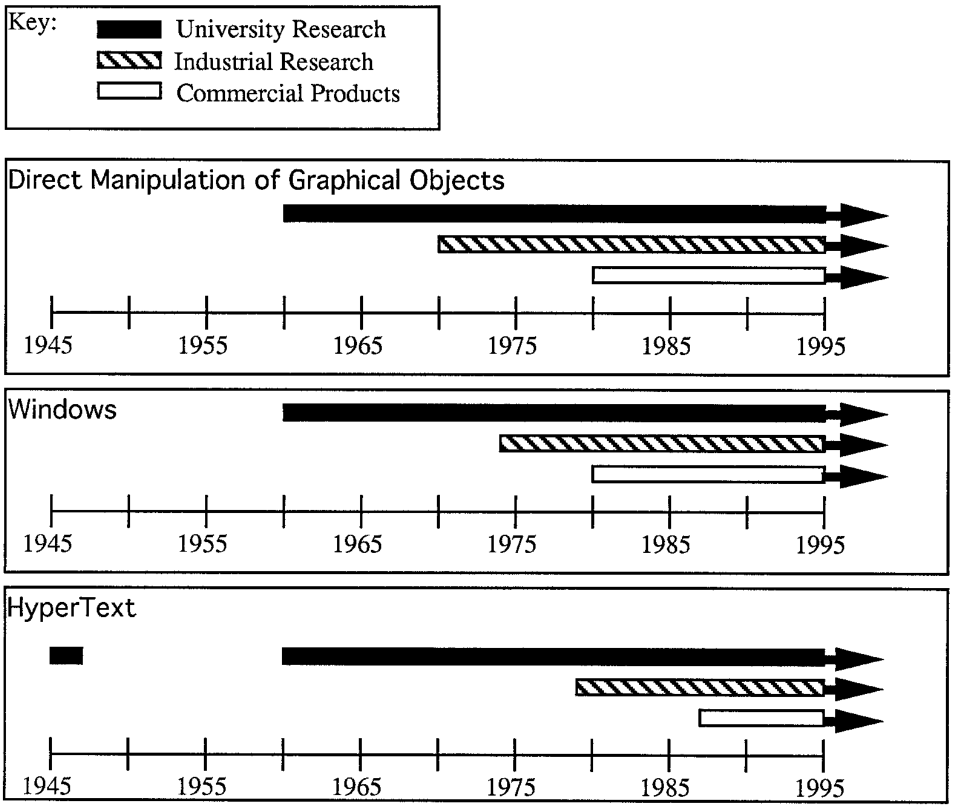

Figure 1. Summary time lines for various technologies.

dence that employing the processes, techniques, and tools developed by the HCI community can dramatically decrease costs and increase productivity. For example, one study [Karat 1990] reported savings due to the use of *usability engineering* [Nielsen 1993] of $41,700 in a small application used by 23,000 marketing personnel, and $6,800,000 for a large business application used by 240,000 employees. Savings were attributed to decreased task time, fewer errors, greatly reduced user disruption, reduced burden on support staff, elimination of training, and avoidance of changes in software after release. Another analysis estimates the mean benefit for finding *each* usability problem at $19,300 [Mantei and Teorey 1988]. A usability analysis of a proposed workstation saved a telephone company $2 million per year in operating costs [Gray et al. 1993]. A mathematical model based on 11 studies suggests that using software that has undergone thorough usability engineering will save a small project $39,000, a medium project $613,000, and a large project $8,200,000 [Nielsen and Landauer 1993]. By estimating all the costs associated with usability engineering, another study found that the benefits can be up to 5000 times the cost [Nielsen and Landauer 1993].

There are also well-known catastrophes that have resulted from not paying enough attention to the human-computer interface. For example, the complicated user interface of the Aegis tracking system was a contributing cause to the erroneous downing of an Iranian passenger plane, and the US

Stark's inability to cope with Iraqi Exocet missiles was partly attributed to the human-computer interface [Neumann 1991]. Problems with the interfaces of military and commercial airplane cockpits have been named as a likely cause for several crashes, including the Cali crash of December, 1995 [Ladkin 1996]. Sometimes the implementation of the user interface can be at fault. A number of people died from radiation overdoses partially as a result of faulty cursor-handling code in the Therac-25 [Leveson and Turner 1993].

Effective user interfaces to complex applications are indispensable. The recognition of their importance in other disciplines is increasing and with it the necessary interdisciplinary collaboration needed to fully address many challenging research problems. For example, for artificial intelligence technologies such as agents, speech, and learning and adaptive systems, effective interfaces are fundamental to general acceptance. HCI subdisciplines such as information visualization and algorithm animation are used in computational geometry, databases, information retrieval, parallel and distributed computation, electronic commerce and digital libraries, and education. HCI requirements resulting from multimedia, distributed computing, real-time graphics, multimodal input and output, ubiquitous computing, and other new interface technologies shape the research problems currently being investigated in disciplines such as operating systems, databases, and networking. New programming languages such as Java result from the need to program new types of distributed interfaces on multiple platforms. As more and more of software designers' time and code are devoted to the user interface, software engineering must increase its focus on HCI.

1.2 History

HCI research has been spectacularly successful and has fundamentally changed computing. Just one example is the ubiquitous graphical interface. Another example is that virtually all software written today employs user interface toolkits and interface builders. Even the spectacular growth of the World-Wide Web is a direct result of HCI technology: applying hypertext technology to browsers allows one to traverse a link across the world with a click of the mouse. It is interface improvements more than anything else that triggered this explosive growth.

In this section we give a brief summary of the research that underlies a few selected HCI advances. By "research," we mean exploratory work at universities and government and industrial research labs (such as Xerox PARC) that is not directly related to products. Figure 1 shows a summary time line. Of course, deeper analysis would reveal much interaction between these three activity streams. For a more complete history, see Myers [1996]. It is important to appreciate that years of research, typically government-funded, are involved in creating and making these technologies ready for widespread use. The same will be true for the HCI technologies that will provide the interfaces of tomorrow.

Direct manipulation of graphical objects. The now ubiquitous direct manipulation interface was first demonstrated by Ivan Sutherland in Sketchpad [Sutherland 1963]. This system was the basis of his 1963 MIT Ph.D. thesis. SketchPad supported manipulation of objects using a light-pen, including grabbing objects, moving them, changing size, and using constraints. It contained the seeds of myriad important interface ideas. The system was built at Lincoln Labs with support from the Air Force and NSF. William Newman's Reaction Handler [Newman 1968], created at Imperial College, London (1966–67), provided direct manipulation of graphics and introduced "Light Handles," a form of graphical potentiometer, that was probably the first "widget." Another early system was AMBIT/G (implemented at

MIT's Lincoln Labs, 1968, ARPA-funded). It employed, among other interface techniques, iconic representations, gesture recognition, dynamic menus, selection of icons by pointing, and moded and mode-free styles of interaction. Smith coined the term "icons" in his 1975 Stanford Ph.D. thesis on Pygmalion [Smith 1977] (funded by ARPA and NIMH) and Smith later popularized icons as one of the chief designers of the Xerox Star [Smith et al. 1982]. Many of the interaction techniques popular in direct manipulation interfaces, such as how objects and text are selected, opened, and manipulated, resulted from research at Xerox PARC in the 1970s. The concept of direct manipulation interfaces for everyone was envisioned by Alan Kay of Xerox PARC in a 1977 article about the Dynabook [Kay 1977]. The first commercial systems to make extensive use of direct manipulation were the Xerox Star (1981) [Smith et al. 1982], the Apple Lisa (1982) [Williams 1983], and Macintosh (1984) [Williams 1984]. Ben Shneiderman at the University of Maryland coined the term "direct manipulation" in 1982, identified the components, and gave psychological foundations [Shneiderman 1983]. The concept was elaborated by other researchers (e.g., Hutchins et al. [1985]).

Windows. Multiple tiled windows were demonstrated in Engelbart's NLS in 1968. Early research at Stanford on systems such as COPILOT (1974) [Swinehart 1974] and at MIT with the EMACS text editor (1974) also demonstrated tiled windows. Alan Kay proposed the idea of overlapping windows in his 1969 University of Utah Ph.D. thesis [Kay 1969] and they first appeared in his 1974 Smalltalk system [Goldberg and Robson 1979] at Xerox PARC, and soon after in the InterLisp system [Teitelman 1979]. One of the first commercial uses of windows was on LMI and Symbolics Lisp Machines (1979), which grew out of MIT AI Lab projects. The main commercial systems popularizing windows were the Xerox Star (1981), the Apple Lisa (1982), and most importantly the Apple Macintosh (1984). Microsoft's original window managers were tiled, but eventually were overlapping. The X Window System, a current international standard, was developed at MIT in 1984 [Scheifler and Gettys 1986]. For a survey of window managers, see Myers [1988].

Hypertext. The idea for hypertext is credited to Vannevar Bush's famous MEMEX idea from 1945 [Bush 1945], but his idea of implementing this using microfilm was never tried. Engelbart's NLS system [Engelbart and English 1968] at the Stanford Research Laboratories in 1965 made extensive use of linking (funding from ARPA, NASA, and Rome ADC). The "NLS Journal," one of the first online journals, included full linking of articles. Ted Nelson coined the term "hypertext" in 1965 [Nelson 1965]. The Hypertext Editing System, jointly designed by Andy van Dam, Ted Nelson, and two students at Brown University (funding from IBM), was distributed extensively [van Dam et al. 1969]. The ZOG project (1977) from CMU was another early hypertext system, and was funded by ONR and DARPA [Robertson et al. 1977]. Ben Shneiderman's Hyperties was the first system where highlighted items in the text could be clicked on to go to other pages (1983, University of Maryland) [Koved and Schneiderman 1986]. HyperCard from Apple (1988) significantly helped to bring the idea to a wide audience. There have been many other hypertext systems through the years. The spectacular growth of the World-Wide Web is a direct result of Tim Berners-Lee's application of Hypertext as the interface to mostly existing capabilities of the Internet. This work was done in 1990 while he was at the government-funded European Particle Physics Laboratory (CERN).

UIMSs and toolkits. The first user interface management system (UIMS) was William Newman's Reaction Handler [Newman 1968], created at Imperial College, London (1966–67 with SRC funding). Most of the early work took place at universities (University of Toronto with Canadian government funding; George Washington University with NASA, NSF, DOE, and NBS funding; Brigham Young University with industrial funding). The term UIMS was coined by David Kasik at Boeing (1982) [Kasik 1982]. Early window managers such as Smalltalk (1974) and InterLisp, both from Xerox PARC, came with a few widgets, such as popup menus and scrollbars. The Xerox Star (1981) was the first commercial system to have a large collection of widgets and to use dialogue boxes. The Apple Macintosh (1984) was the first to actively promote its toolkit for use by other developers to enforce a consistent interface. An early C++ toolkit was InterViews [Linton and Vlissides 1989], developed at Stanford (1988, industrial funding). Much of current research is now being performed at universities, including Garnet [Myers et al. 1990] and Amulet [Myers et al. 1996] at CMU (ARPA funded), MasterMind [Neches et al. 1993] at Georgia Tech (ARPA-funded), and Artkit [Hudson and Smith 1996] at Georgia Tech (funding from NSF and Intel).

There are, of course, many other examples of HCI research that should be included in a complete history, including work that led to drawing programs, paint programs, animation systems, text editing, spreadsheets, multimedia, 3D, virtual reality, interface builders, event-driven architectures, usability engineering, and a very long list of other significant developments [Myers 1996]. Although our brief history here has had to be selective, what we hope is clear is that there are many years of productive HCI research behind our current interfaces and that it has been research results that have led to the successful interfaces of today.

For the future, HCI researchers are developing interfaces that will greatly facilitate interaction and make computers useful to a wider population. These technologies include: handwriting and gesture recognition, speech and natural language understanding, multiscale zoomable interfaces, "intelligent agents" to help users understand systems and find information, end-user programming systems so people can create and tailor their own applications, and much, much more. New methods and tools promise to make the process of developing user interfaces significantly easier but the challenges are many as we expand the modalities that interface designers employ and as computing systems become an increasingly central part of virtually every aspect of our lives.

As HCI has matured as a discipline, a set of principles emerged that are generally agreed upon and taught in courses on HCI at the undergraduate and graduate level (e.g., see Greenberg [1996]). These principles should be taught to every CS undergraduate, since virtually all programmers will be involved in designing and implementing user interfaces during their careers. These principles are described in other publications, such as Hewett et al. [1992], and include task analysis, user-centered design, and evaluation methods.

1.3 Foundations of the Field

The intellectual foundations of HCI derive from a variety of fields: computer science, cognitive psychology, social psychology, perceptual psychology, linguistics, artificial intelligence, and anthropology. Decades of research in perceptual and cognitive psychology were distilled by pioneers in HCI, beginning in the 1960s (e.g., Shackel [1969]), and several workers have explored the relationship between these sciences and the demands of design (e.g., Barnard [1991] and Landauer [1991]).

One influential early effort was directed at producing an "engineering model of human performance" able to make quantitative predictions that can contribute to design (the Model Human Processor [Card et al. 1983]). Drawing also on research into human problem solving [Ernst and Newell 1969; Newell and Simon 1972], this led to the GOMS family of analysis techniques that make quantitative predictions of skilled performance. Extensions and refinements of these models have continued to draw on basic psychological theories [Olson and Olson 1990]. In addition, the needs of HCI have given rise to new psychological theories, for example, Polson and Lewis's theory of learning through exploration that predicts behavior in walk-up-and-use interfaces and other applications where exploration is the norm [Polson and Lewis 1990].

Donald Norman and his colleagues applied knowledge from the psychology of perception, attention, memory, and motor control to human-computer interaction and design in a series of influential papers and books (e.g., Norman [1986, 1990]). The think-aloud protocol technique [Ericsson and Simon 1984], developed in cognitive psychology to assist human problem-solving research [Newell and Simon 1972], influenced early HCI work and has become a valuable usability engineering method [Nielsen 1993, p. 195]. Requirements-setting for HCI uses techniques from anthropology (e.g., ethnographic techniques [Blomberg et al. 1993]). Evaluation uses experimental techniques long established in experimental psychology. Social psychology contributes methods for discourse analysis (e.g., Clark [1985]), interviewing, and questionnaires. Using methods researched and validated in other scientific fields allows HCI to move quickly to robust valid results that are applicable to the more applied area of design.

The intellectual foundations of HCI also include the development of object-oriented programming. This style of programming comes from early work on Simula but was further developed and refined in Smalltalk as a natural way to implement user interfaces [Kay 1977]. Early HCI software work drew on compiler theories such as the conceptual/semantic/syntactic/lexical model and parser technologies. Constraint systems and solvers were developed to ease UI implementations ranging from SketchPad [Sutherland 1963] and ThingLab [Borning 1981] to Amulet [Myers et al. 1996] and Artkit [Hudson and Smith 1996].

Current widely used interaction techniques, such as how menus and scroll bars work, have been refined through years of research and experimentation. We now know how to provide effective control using the mouse and keyboard for 2D interfaces.

2. PROSPECTIVE

Although we are encouraged by past research success in HCI and excited by the potential of current research, we want to emphasize how central a strong research effort is to future practical use of computational and network technologies. For example, popular discussion of the National Information Infrastructure (NII) envisions the development of an information marketplace that can enrich people's economic, social, cultural, and political lives. For such an information marketplace or, in fact, many other applications, to be successful requires solutions to a series of significant research issues that all revolve around better understanding how to build effective human-centered systems. The following sections discuss selected strategic themes, technology trends, and opportunities to be addressed by HCI research.

2.1 Strategic Themes

If one steps back from the details of current HCI research a number of themes are visible. Although we cannot hope to do justice here to elaborating these or a number of other themes that arose in our workshop discussions, it is clear that HCI research has now started to crystallize as a critical discipline, intimately involved in virtually all uses of computer technologies, and decisive to successful applications. Here we expand on just a few themes.

2.1.1. *Universal Access to Large and Complex Distributed Information.* As the "global information infrastructure" expands at unprecedented rates, there are dramatic changes taking place in the kind of people who access the available information and the types of information involved. Virtually all entities (from large corporations to individuals) are engaged in activities that increasingly involve accessing databases, and their livelihood and/or competitiveness depend heavily on the effectiveness and efficiency of that access. As a result, the potential user community of database and other information systems is becoming startlingly large and rather nontechnical, with most users bound to remain permanent novices with respect to many of the diverse information sources they can access. It is therefore urgently necessary and strategically critical to develop user interfaces that require minimal technical sophistication and expertise by the users and support a wide variety of information-intensive tasks [Silberschatz and Stonebraker et al 1996].

Information-access interfaces must offer great flexibility on how queries are expressed and how data are visualized; they must be able to deal with several new kinds of data, for example, multimedia, free text, documents, the Web itself; and they must permit several new styles of interaction beyond the typical, two-step query-specification/result-visualization loop, for example, data browsing, filtering, and dynamic and incremental querying. Fundamental research is required on visual query languages, user-defined and constraint-based visualizations, visual metaphors, and generic and customizable interfaces. Advances seem most likely to come from collaborations between the HCI and database research communities.

Information-discovery interfaces must support a collaboration between humans and computers, for example, for data mining. Because of our limited memory and cognitive abilities, the growing volume of available information has increasingly forced us to delegate the discovery process to computers, greatly underemphasizing the key role played by humans. Discovery should be viewed as an interactive process in which the system gives users the necessary support to analyze terabytes of data and users give the system the feedback necessary to better focus its search. Fundamental issues for the future include how best to array tasks between people and computers, create systems that adapt to different kinds of users, and support the changing context of tasks. Also, the system could suggest appropriate discovery techniques depending on data characteristics as well as data visualizations, and help integrate what are currently different tools into a homogeneous environment (see Brachman and Anand [1996] and Keim [1995]).

2.1.2 *Education and Lifelong Learning.* Computationally assisted access to information has important implications for education and learning, as evidenced in current discussions of "collaboratories" and "virtual universities." Education is a domain that is fundamentally intertwined with human-computer interaction. HCI research includes both the development and evaluation of new educational technologies such as multimedia systems, interactive simulations, and computer-assisted instructional materials. For example, consider distance-learning situations involving individuals far away from schools. What types of learning environments, tools, and media effectively deliver the knowledge and understanding that these individuals seek? Furthermore, what constitutes an effective educational technology? Do particular media or types of simulations foster different types of learning? These questions apply not only to K-12 and college students, but also to adults through lifelong learning. Virtually every current occupation involves workers who encounter new technologies and require additional training. How can computer-assisted instructional systems engage individuals and help them to learn new ideas? HCI research is crucial to answering these important questions.

2.1.3 *Electronic Commerce.* Another important theme revolves around the increasing role of computation in our economic life and highlights central HCI issues that go beyond usability to concerns with privacy, security, and trust. Although currently there is much hyperbole, as with most Internet technologies, over the next decade commercialization of the Internet may mean that digital commerce replaces much traditional commerce [Margolis 1996]. The Internet makes possible services that could potentially be quite adaptive and responsive to consumer wishes. Digital commerce may require dramatic changes to internal processes as well as the invention of new processes [Lynch et al. 1996]. For digital commerce to be successful, the technology surrounding it will have to be affordable, widely available, simple to use, and secure. Interface issues are, of course, key.

2.1.4 *End-User Programming.* An important reason that the WWW has been so successful is that everyone can create his or her own page. With the advent of WYSIWYG HTML page-editing tools, it will be even easier. However, for "active" pages that use forms, animations, or computation, a professional programmer must write the required code in a programming language like PERL or Java. The situation is the same for the desktop where applications are becoming increasingly programmable (e.g., by writing Visual Basic scripts for Microsoft Word), but only to those with training in programming. Applying the principles and methods of HCI to the design of programming languages and programming systems for end-users should bring to everyone the ability to program Web pages and desktop applications.

End-user programming will be increasingly important in the future. No matter how successful interface designers are, systems will still need to be customized to the needs of particular users. Although there will likely be generic structures, for example, in an email filtering system, that can be shared, such systems and agents will always need to be tailored to meet personal requirements. The use of various scripting languages to meet such needs is widespread, but better interfaces and understandings of end-user programming are needed.

2.1.5 *Information Visualization.* This area focuses on graphical mechanisms designed to show the structure of information and improve the cost structure of access. Previous approaches have studied novel visualizations for information, such as the Information Visualizer [Card et al. 1991; Robertson et al. 1993], history-enriched digital objects for displaying graphical abstractions of interaction history [Hill and Hollan 1994], and dotplots for visualizing self-similarity in millions of lines of text and code [Church and Helfman 1993]. Other approaches provide novel techniques for displaying data, for example, dynamic queries [Shneiderman et al. 1994], visual query languages [Cruz 1994], zoomable interfaces for supporting multiscale interfaces [Bederson et al. 1996], and lenses to provide alternative views of information [Bier et al. 1993]. Another branch of research is studying automatic selection of visualizations based on properties of the data and the user's tasks (e.g., Mackinlay [1986] and Roth et al. [1994]).

The importance of information visualization will increase as people have access to larger and more diverse sources of information (e.g., digital libraries, large databases), which are becoming universally available with the WWW. Visualizing the WWW itself and other communication networks is also an important aim of information visualization systems (see, e.g., Catarci and Cruz [1996]). The rich variety of information may be handled by giving the users the ability to tailor the visualization to a particular application, to the size of the data set, or to the device (e.g., 2D vs. 3D capabilities, large vs. small screens). Research challenges include making the specification, exploration, and evolution of visualizations interactive and accessible to a variety of users. Tools should be designed that support a range of tailoring capabilities, from specifying visualizations from scratch to minor adaptations of existing visualizations. Incorporating automatic generation of information visualization with user-defined approaches is another interesting open problem, for example, when the user-defined visualization is underconstrained.

One fundamental issue for information visualization is how to characterize the expressiveness of a visualization and judge its adequacy to represent a data set. For example, the "readability" of a visualization of a graph may depend on (often conflicting) aesthetic criteria, such as the minimization of edge crossings and of the area of the graph, and the maximization of symmetries [DiBattista et al. 1994]. For other types of visualization, the criteria are quite ad hoc. Therefore, more foundation work is needed for establishing general principles (see, for example, European Working Group [1996]).

2.1.6 *Computer-Mediated Communication.* Examples of computer-mediated communication range from work that led to extraordinarily successful applications such as email to that involved in newer forms of communication via computers, such as real-time video and audio interactions. Research in computer-supported cooperative work (CSCW) confronts complex issues associated with integration of several technologies (e.g., telephone, video, 3D graphics, cable, modem, fax, email), support for multiperson activities (which have particularly difficult interface development challenges), and issues of security, privacy, and trust.

The unpredicted shift of focus to the Internet, intranets, and the World-Wide Web has ended a period in which the focus was on the interaction between an individual and a computer system, with relatively little attention to group and organizational contexts. Computer-mediated human communication raises a host of new interface issues. Additional challenges arise in coordinating the activities of computer-supported group members, either by providing shared access to common online resources and letting people structure their work around them, or by formally representing work processes to enable a system to guide the work. The CSCW subcommunity of human-computer interaction has grown rapidly, drawing from diverse disciplines. Social theory and social science, management studies, communication studies, and education are among the relevant areas of knowledge and expertise. Techniques drawn from these areas, including ethnographic approaches to understanding group activity, have become important adjuncts to more familiar usability methods.

Mounting demands for more function, greater availability, and interoperability affect requirements in all areas. For example, the great increase in accessible information shifts the research agenda toward more sophisticated information retrieval techniques. Approaches to dealing with the new requirements through formal or de facto standards can determine where research is pointless as well as where it is useful. As traditional applications are integrated into the Web, social aspects of computing are extended.

2.2 Technological Trends

Again, the number and variety of trends identified in our discussions outstrip the space we have here for reporting. One can see large general trends that are moving the field, from concerns about connectivity, as the networked world becomes a reality, to compatibility, as applications increasingly need to run across different platforms and code begins to move over networks as easily as data, to issues of coordination, as we understand the need to support multi-person and organization activities. We limit our discussion here to a few instances of these general trends.

2.2.1 *Computational Devices and Ubiquitous Computing.* One of the most notable trends in computing is the increase in the variety of computational devices with which users interact. In addition to workstations and desktop personal computers, users are faced with (to mention only a few) laptops, PDAs, and LiveBoards [Elrod et al. 1994]. In the near future, Internet telephony will be universally available, and the much-heralded Internet appliance may allow interactions through the user's television and local cable connection. In the more distant future, wearable devices may become more widely available. All these technologies have been considered under the heading "ubiquitous computing" [Weiser 1993] because they involve using computers everywhere, not just on desks.

The introduction of such devices presents a number of challenges to the discipline of HCI. First, there is the tension between the design of interfaces appropriate to the device in question and the need to offer a uniform interface for an application across a range of devices. The computational devices differ greatly, most notably in the sizes and resolutions of displays but also in the available input devices, the stance of the user (is the user standing, sitting at a desk, or on a couch?), the physical support of the device (is the device sitting on a desk, mounted on a wall, or held by the user, and is the device immediately in front of the user or across the room?), and the social context of the device's use (is the device meant to be used in a private office, a meeting room, a busy street, or a living room?). On the other hand, applications offered across a number of devices need to offer uniform interfaces, both so that users can quickly learn to use a familiar application on new devices and so that a given application can retain its identity and recognizability, regardless of the device on which it is operating.

Development of systems meeting these requirements will involve user testing and research into design of displays and input devices, as well as into design of effective interfaces, but some systems have already begun to address these problems. Some browsers for the World-Wide Web attempt to offer interfaces that are appropriate to the devices on which they run and yet offer some uniformity. At times this can be difficult. For example, the frames feature of HTML causes a browser to attempt to divide up a user's display without any knowledge of the characteristics of that display. Although building applications that adapt their interfaces to the characteristics of the device on which they are running is one potential direction of research in this area, perhaps a more promising one is to separate the interface from the application and give the responsibility of maintaining the interface to the device itself. A standard set of protocols would allow the application to negotiate the setup of an interface and later to interact with that interface and, indirectly, with the user. Such multimodal architectures could address the problems of generating an appropriate interface as well as providing better support for users with specific disabilities. The architectures could also be distributed, and the building blocks of forthcoming distributed applications could become accessible from assorted computational devices.

2.2.2 *Speed, Size, and Bandwidth.* The rate of increase of processor speed and storage (transistor density of semiconductor chips doubles roughly every 18 months, according to Moore's law) suggests a bright future for interactive technologies. An important constraint on utilizing the full power afforded by these technological advances, however, may be network bandwidth. Given the overwhelming trends towards global networked computing, and even the network as computer, the implications of limited bandwidth deserves careful scrutiny. The bottleneck is the "last mile" connecting the Internet to individual homes and small offices [Bell and Gemmell 1996]. Individuals who do not get access through large employers may be stuck at roughly the present bandwidth rate (28,800 kilobits per second) at least until the turn of the century. The rate needed for delivery of television-quality video, one of the promises of the National Information Infrastructure, is 4–6 megabits, many times that amount. What are the implications for strategic HCI research of potentially massive local processing power together with limited bandwidth?

Increases in processor speed and memory suggest that if the information can be collected and cached from the network and/or local sources, local interactive techniques based on signal processing and work context could be utilized to the fullest. With advances in speech and video processing, interfaces that actively watch, listen, catalogue, and assist become possible. With increased CPU speed we might design interactive techniques based on work context rather than isolated event handling. Fast event dispatch becomes less important than helpful action. Tools might pursue multiple redundant paths, leaving the user to choose and approve rather than manually specify. We can afford to "waste" time and space on indexing information and tasks that may never be used, solely for the purpose of optimizing user effort. With increased storage capacity it becomes potentially possible to store every piece of interactive information that a user or even a virtual community ever sees. The processes of sifting, sorting, finding, and arranging increase in importance relative to the editing and browsing that characterizes today's interfaces. When it is physically possible to store every paper, email, voice-mail, and phone conversation in a user's working life, the question arises of how to provide effective access.

2.2.3 *Speech, Handwriting, Natural Language, and Other Modalities.* The use of speech will increase the need to allow user-centered presentation of information. Where the form and mode of the output generated by computer-based systems are currently defined by the system designer, a new trend may be to increasingly allow the user to determine the way in which the computer will interact and to support multiple modalities at the same time. For instance, the user may determine that in a given situation, textual natural language output is preferred to speech, or that pictures may be more appropriate than words. These distinctions will be made dynamically, based on the abilities of the user or the limitations of the presentation environment. As the computing environment used to present data becomes distinct from the environment used to create or store information, interface systems will need to support information adaptation as a fundamental property of information delivery.

2.2.4 *3D and Virtual Reality*. Another trend is the migration from two-dimensional presentation space (or 2½-dimensional space, in the case of overlapping windows) to three-dimensional space. The beginnings of this in terms of a conventional presentation environment is the definition of the Virtual Reality Modeling Language (VRML). Other evidences are the use of integrated three-dimensional input and output control in virtual reality systems. The notions of selecting and interacting with information will need to be revised, and techniques for navigation through information spaces will need to be radically altered from the present page-based models. Three-dimensional technologies offer significant opportunities for human-computer interfaces. Application areas that may benefit from three-dimensional interfaces include training and simulation, as well as interactive exploration of complex data environments.

A central aspect of three-dimensional interfaces is "near-real-time" interactivity, the ability for the system to respond quickly enough that the effect of direct manipulation is achieved. Near-real-time interactivity implies strong performance demands that touch on all aspects of an application, from data management through computation to graphical rendering. Designing interfaces and applications to meet these demands in an application-independent manner presents a major challenge to the HCI community. Maintaining the required performance in the context of an unpredictable user-configured environment implies a "time-critical" capability, where the system automatically gracefully degrades quality in order to maintain performance. The design of general algorithms for time-critical applications is a new area and a significant challenge.

2.3 Design and Evaluation Methods

Design and evaluation methods have evolved rapidly as the focus of human-computer interaction has expanded. Contributing to this are the versatility of software and the downward price and upward performance spiral, which continually extend the applications of software. The challenges overshadow those faced by designers using previous media and assessment methods. Design and evaluation for a monochrome, ASCII, stand-alone PC was challenging, and still does not routinely use more than ad hoc methods and intuition. New methods are needed to address the complexities of multimedia design, of supporting networked group activities, and of responding to routine demands for ever-faster turnaround times.

More rapid evaluation methods will remain a focus, manifest in recent work on cognitive walkthrough [Wharton et al. 1994], heuristic evaluation [Nielsen 1994], and other modifications of earlier cognitive modeling (e.g., John and Kieras [1997]) and usability engineering approaches. Methods to deal with the greater complexity of assessing use in group settings are moving from research into the mainstream. Ethnographic observation, participatory design, and scenario-based design are being streamlined [Schuler and Namioka 1993]. Contextual inquiry and design is an example of a method intended to quickly obtain a rich understanding of an activity and transfer that understanding to all design team members [Holtzblatt and Jones 1993].

As well as developing and refining the procedures of design and evaluation methods, we need to understand the conditions under which they work. Are some better for individual tasks, some excellent for supporting groupware? Are some useful very early in the conceptual phase of design, others best when a specific interface design has already been detailed, and some restricted to when a prototype is in existence? In addition, for proven and promising techniques to become widespread, they need to be incorporated into the education of UI designers. Undergraduate curricula should require such courses for a subset of their students; continuing education courses need to be developed to address the needs of practicing designers.

2.4 Tools

All the forms of computer-human interaction discussed here will need to be supported by appropriate tools. The interfaces of the future will use multiple modalities for input and output (speech and other sounds, gestures, handwriting, animation, and video) and multiple screen sizes (from tiny to huge), and have an "intelligent" component ("wizards" or "agents" to adapt the interface to the different wishes and needs of the various users). The tools used to construct these interfaces will have to be substantially different from those of today. Whereas most of today's tools well support widgets such as menus and dialogue boxes, these will be a tiny fraction of the interfaces of the future. Instead, the tools will need to access and control in some standard way the main application data structures and internals, so the speech system and agents can know what the user is talking about and doing. If the user says "delete the red truck," the speech system needs access to the objects to see which one is to be deleted. Otherwise, each application will have to deal with its own speech interpretation, which is undesirable. Furthermore, an agent might notice that this is the third red truck that was deleted, and propose to delete the rest. If confirmed, the agent will need to be able to find the rest of the trucks that meet the criteria. Increasingly, future user interfaces will be built around standardized data structures or "knowledge bases" to make these facilities available without requiring each application to rebuild them.

In addition, tools of the future should incorporate the design and evaluation methods discussed in Section 2.3. These procedures should be supported by the system-building tools themselves. This would make the evaluation of ideas extremely easy for designers, allowing ubiquitous evaluation to become a routine aspect of system design.

3. CONCLUSIONS

Although some areas of computer science are maturing and perhaps no longer have the excitement they once did, the current generally felt concern with developing human-centered systems, that is, those that more effectively support people in accomplishing their tasks, is bringing HCI to the center of computer science. We have never had more interest, positive publicity, and recognition of the importance of the area. And it is warranted. We now have a solid foundation of principles and results to teach in courses and from which to base today's user interface design and tomorrow's research. As computing systems become increasingly central to our society, HCI research will continue to grow in importance. The field can expect a stream of rapidly changing technological developments, challenges associated with integrating research from multiple disciplines, and crucially important problems to address. We look forward to many exciting new HCI research results in the future as well as the benefits associated with their application.

REFERENCES

Barnard, P. 1991. The contributions of applied cognitive psychology to the study of human-computer interaction. In *Human Factors for Informatics Usability*, B. Shackel, and S. Richardson, Eds., Cambridge University Press, New York, 151–182.

Bederson, B., Hollan, J., Perlin, K., Meyer, J., Bacon, D., and Furnas, G. 1996. Pad++: A zoomable graphical sketchpad for exploring alternate interface physics. *J. Visual Lang. Comput. 7*, 3–31.

Bell, G. and Gemmell, J. 1996. On ramp prospects for the information superhighway dream. *Commun. ACM 39*, 7 (July), 55–61.

Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T. D. 1993. Toolglass and Magic lenses: The see-through interface. *SIGGRAPH Comput. Graph. Proc.*, 73–80.

Blomberg, J., Giacomi, J., Mosher, A., and Swenton-Wall, P. 1993. Ethnographic field methods and their relation to design. In *Participatory Design: Principles and Practices*, D. Schuler and A. Namioka Eds., Lawrence Erlbaum Associates, Hillsdale, NJ, 123–155.

Borning, A. 1981. The programming language aspects of ThingLab, a constraint-oriented simulation laboratory. *ACM Trans. Program. Lang. and Syst. 3*, 4 (Oct.), 353–387.

Brachman, R. J. and Anand, T. 1996. The process of knowledge discovery in databases. In *Advances in Knowledge Discovery and Data Mining*, U. M. Fayad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Eds., AAAI Press, 37–57.

Bush, V. 1945. As we may think. *The Atlantic Monthly 176* (July), 101–108.

Card, S. K., Moran, T. P., and Newell, A. 1983. *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Card, S. K., Robertson, G. G., and Mackinlay, J. D. 1991. The information visualizer, information workspace. In *Proceedings of the ACM Conference on Computer Human Interaction, CHI'91*, 181–188.

CATARCI, T. AND CRUZ, I. F., EDS. 1996. Special issue on information visualization. *ACM Sigmod Rec. 25*, 4.

CHURCH, K. W. AND HELFMAN, J. I. 1993. Dotplot: A program for exploring self-similarity in millions of lines of text and code. *J. Comput. Graph. Stat. 2*, 153–174.

CLARK, H. H. 1985. Language use and language users. In *The Handbook of Social Psychology*, (3rd ed.). G. Lindzey & E. Aronson (Eds.), Knopf, New York.

CRUZ, I. F. 1994. User-defined visual query languages. In *Proceedings of the IEEE Tenth International Symposium on Visual Languages* (VL '94) (St. Louis, Oct.), 224–231.

DIBATTISTA, G., EADES, P., TAMASSIA, R., AND TOLLIS, I. G. 1994. Algorithms for drawing graphs: An annotated bibliography. *Comput. Geom. Theory Appl. 4*, 5, 235–282.

ELROD, S., BRUCE, R., GOLD, R., GOLDBERG, D., HALASZ, F., JANSSEN, W., LEE, D., MCCALL, K., PEDERSEN, E., PIER, K., TANG, J., AND WELCH, B. 1992. Liveboard: A large interactive display supporting group meetings, presentations and remote collaboration. In *Proceedings of the Conference on Computer Human Interaction, CHI'94* (May), 599–607.

ENGELBART, D. AND ENGLISH, W. 1968. A research center for augmenting human intellect. Reprinted in *ACM SIGGRAPH Video Review*, 1994. 106.

ERICSSON, K. A. AND SIMON, H. A. 1984. *Protocol Analysis: Verbal Reports as Data*. MIT Press, Cambridge, MA.

ERNST, G. W. AND NEWELL, A. 1969. *GPS: A Case Study in Generality and Problem-Solving*. Academic Press, New York.

EUROPEAN (ESPRIT) WORKING GROUP 1996. FADIVA, foundations of advanced 3D information visualization. http://www-cui.darmstadt.gmd.de:80/visit/Activities/Fadiva/.

FOLEY, J. AND PITKOW, J., EDS. 1994. Research priorities for the World-Wide Web. October 31. http://www.cc.gatech.edu/gvu/nsf-ws/report/Report.html.

GOLDBERG, A. AND ROBSON, D. 1979. A metaphor for user interface design. In *Proceedings of the 12th Hawaii International Conference on System Sciences*, 148–157.

GRAY, W. D., JOHN, B. E., AND ATWOOD, M. E. 1993. Project Ernestine: Validating a GOMS analysis for predicting and explaining real-world task performance. *Hum. Comput. Interaction 8*, 237–309.

GREENBERG, S. 1996. Teaching human computer interaction to programmers. *ACM Interactions 3*, 4 (July), 62–76.

HEWETT, T. T. ET AL., EDS. 1992. *ACM SIGCHI Curricula for Human-Computer Interaction*. ACM Press, New York.

HILL, W. AND HOLLAN, J. D. 1994. History-enriched digital objects: Prototypes and policy. *Inf. Soc., 10*, 139–145.

HOLTZBLATT, K. AND JONES, S. 1993. Contextual inquiry: A participatory technique for system design. In *Participatory Design: Principles and Practices*, A. Namioka and D. Schuler Eds., Lawrence Erlbaum Associates. Hillsdale, NJ.

HUDSON, S. E. AND SMITH, I. 1996. Ultra-lightweight constraints. In *Proceedings of the ACM Symposium on User Interface Software and Technology, UIST'96*, (Seattle, WA, Nov. 6–8), 147–155.

HUTCHINS, E. L., HOLLAN, J. D., AND NORMAN, D. A. 1985. Direct manipulation interfaces. *Hum. Comput. Interaction 1*, 311–338.

JOHN, B. E. AND KIERAS, D. E. 1997. Using GOMS for user interface design and evaluation: Which technique? *ACM Trans. Comput. Hum. Interaction* (to appear).

KARAT, C. M. 1990. Cost-benefit analysis of usability engineering techniques. In *Proceedings of the Human Factors Society 34th Annual Meeting* (Orlando, FL, Oct.), Vol. 2, 839–843.

KASIK, D. J. 1982. A user interface management system. *Comput. Graph. 16*, 3.

KAY, A. 1969. The reactive engine. Ph.D. Thesis, Electrical Engineering and Computer Science, University of Utah.

KAY, A. 1977. Personal dynamic media. *IEEE Comput. 10*, 3, 31–42.

KEIM, D. 1995. Visual support for query specification and data mining. Ph.D. Thesis, Institute for Computer Science, Ludwig Maximilians University, Munich, Germany.

KOVED, L. AND SHNEIDERMAN, B. 1986. Embedded menus: Selecting items in context. *Commun. ACM 29*, 4 (April), 312–318.

LADKIN, P. 1996. Inside risks forum. http://csrc.ncsl.nist.gov//rskforum/risks18.010.

LANDAUER, T. 1991. Let's get real: A position paper on the role of cognitive psychology in the design of humanly useful and usable systems. In *Designing Interaction*, J. Carroll, Ed., Cambridge University Press, New York, 60–73.

LEVESON, N. G. AND TURNER, C. S. 1993. An investigation of the Therac-25 accidents. *IEEE Comput. 26*, 7 (July), 18–41.

LINTON, M. A., VLISSIDES, J. M., ET AL. 1989. Composing user interfaces with InterViews. *IEEE Comput. 22*, 2, 8–22.

LYNCH, D. C., DANIEL, C., AND LUNDQUIST, L. 1996. *Digital Money: The New Era of Internet Commerce*. John Wiley and Sons, New York.

MACKINLAY, J. 1986. Automating the design of graphical presentation of relational information. *ACM Trans. Graph. 5*, 2 (April), 110–141.

MANTEI, M. M. AND TEOREY, T. J. 1988. Cost/benefit analysis for incorporating human factors in the software lifecycle. *Commun. ACM 31*, 4 (April) 428–439.

MARGOLIS, B. 1996. Digital commerce: the future of retailing. *Direct Marketing 41*, 6 (Jan.).

MYERS, B. A. 1988. A taxonomy of user interfaces for window managers. *IEEE Comput. Graph. Appl. 8* 5, 65–84.

MYERS, B. A. 1996. A quick history of human computer interaction technology. Carnegie Mellon University School of Computer Science Tech. Rep. CMU-CS-96-163 and Human Computer Interaction Institute Tech. Rep. CMU-HCII-96-103, Nov.

MYERS, B. A. AND ROSSON, M. B. 1992. Survey on user interface programming. In *Proceedings SIGCHI'92: Human Factors in Computing Systems* (Monterey, CA, May 3–7), 195–202.

MYERS, B. A., GIUSE, D. A., ET AL. 1990. Garnet: Comprehensive support for graphical, highly-interactive user interfaces. *IEEE Comput. 23*, 11, 71–85.

MYERS, B. A., MILLER, R. C., MCDANIEL, R., AND FERRENCY, A. 1996. Easily adding animations to interfaces using constraints. In *ACM Symposium on User Interface Software and Technology, UIST'96*, (Seattle, WA, Nov. 6–8), 119–128.

NECHES, R., FOLEY, J., SZEKELY, P., SUKAVIRIYA, P., LUO, P., KOVACEVIC, S., AND HUDSON, S. 1993. Knowledgeable development environments using shared design models. In *Proceedings of the ACM SIGCHI 1993 International Workshop on Intelligent User Interfaces* (Orlando, FL, Jan.), 63–70.

NELSON, T. 1965. A File Structure for the Complex, the Changing, and the Indeterminate. In *Proceedings ACM National Conference*, 84–100.

NEUMANN, P. G. 1991. Inside risks: Putting on your best interface. *Commun. ACM 34*, 3 (March), 138.

NEWELL, A. AND SIMON, H. A. 1972. *Human Problem Solving*. Prentice-Hall, Englewood Cliffs, NJ.

NEWMAN, W. M. 1968. A system for interactive graphical programming. In *Proceedings of the AFIPS Spring Joint Computer Conference*.

NIELSEN, J. 1993. *Usability Engineering*, Academic Press, Boston.

NIELSEN, J. 1994. Heuristic evaluation. In *Usability Inspection Methods*, J. Nielsen and R. L. Mack, Eds., John Wiley & Sons, New York, 25–62.

NIELSEN, J. AND LANDAUER, T. K. 1993. A mathematical model of the finding of usability problems. In *Proceedings INTERCHI'93: Human Factors in Computing Systems* (Amsterdam, The Netherlands, April), 206–213.

NORMAN, D. A. 1986. Cognitive engineering. In *User Centered System Design*, D. A. Norman and S. W. Draper, Eds., Lawrence Erlbaum Associates, Hillsdale, NJ.

NORMAN, D. A. 1990. *The Design of Everyday Things*. Doubleday Currency, New York.

OLSON, J. AND OLSON, G. 1990. The growth of cognitive modeling in human-computer interaction since GOMS. *Hum. Comput. Interaction 5*, 221–265.

POLSON, P. AND LEWIS, C. 1990. Theory-based design for easily learned interfaces. *Hum. Comput. Interaction 5*, 191–220.

ROBERTSON, G. G., CARD, S. K., AND MACKINLAY, J. D. 1993. Information visualization using 3D Interactive Visualization. *Commun. ACM, 26*, 4 (April), 56–71.

ROBERTSON, G., NEWELL, A., ET AL. 1977. *ZOG*: A man-machine communication philosophy. Carnegie Mellon University Tech. Rep.

ROTH, S. F., KOLOJEJCHICK, J., MATTIS, J., AND GOLDSTEIN, J. 1994. Interactive graphic design using automatic presentation knowledge. In *ACM Proceedings Human Factors in Computing Systems*, CHI'94, 112–117.

SCHEIFLER, R. W. AND GETTYS, J. 1986. The X Window System. *ACM Trans. Graph. 5*, 2, 79–109.

SCHULER, D. AND NAMIOKA, A., EDS. 1993. *Participatory Design: Principles and Practices*. Lawrence Erlbaum Associates, Hillsdale, NJ.

SHACKEL, B. 1969. Man-computer interaction—the contribution of the human sciences. *Ergonomics 12*, 4, 485–499.

SHNEIDERMAN, B. 1983. Direct manipulation: A step beyond programming languages. *IEEE Comput. 16*, 8, 57–69.

SHNEIDERMAN, B. 1994. Dynamic queries for visual information seeking. *IEEE Softw. 11*, 6, 70–77.

SIBERT, J. AND MARCHIONINI, G. 1993. Human-computer interaction research agendas. *Behav. Inf. Tech. 12*, 2 (March–April), 67–135.

SILBERSCHATZ, A., STONEBRAKER, M., AND ULLMAN, J., EDS. 1996. Database research: Achievements and opportunities into the 21st century. *ACM Sigmod Rec. 25*, 1, 52–63. http://bunny.cs.uiuc.edu/sigmod/sigmod_record/3-96/lagunita.ps.

SMITH, D. C. 1977. *Pygmalion: A Computer Program to Model and Stimulate Creative Thought*. Birkhauser Verlag, Basel.

SMITH, D. C., HARSLEM, E., ET AL. 1982. The Star user interface: An overview. In *Proceedings of the 1982 National Computer Conference*, AFIPS.

STRONG, G. W. 1994. *NSF Workshop on New*

Directions in Human-Computer Interaction Education, Research and Practice (Sept. 11). http://www.sei.cmu.edu/arpa/hci/directions/TitlePage.html.

SUTHERLAND, I. E. 1963. SketchPad: A man-machine graphical communication system. In *Proceedings of the AFIPS Spring Joint Computer Conference*.

SWINEHART, D. C. 1974. Copilot: A multiple process approach to interactive programming systems. Computer Science Department, Stanford University.

TEITELMAN, W. 1979. A display oriented programmer's assistant. *Int. J. Man-Mach. Stud. 11*, 157–187.

VAN DAM, A., CARMODY, S., GROSS, T., NELSON, T., AND RICE, D. 1969. A hypertext editing system for the /360. In *Proceedings Conference in Computer Graphics* (University of Illinois).

WEISER, M. 1993. Some computer science issues in ubiquitous computing. *Commun. ACM 36*, 7 (July), 74–83.

WHARTON, C., RIEMAN, J., LEWIS, C., AND POLSON, P. 1994. The cognitive walkthrough method: A practitioner's guide. In *Usability Inspection Methods*, J. Nielsen and R. L. Mack, Eds., John Wiley & Sons, New York, 105–140.

WILLIAMS, G. 1983. The Lisa computer system. *Byte 8*, 2, 33–50.

WILLIAMS, G. 1984. The Apple Macintosh computer. *Byte 9*, 2, 30–54.