ELSEVIER

The Journal of Systems and Software 43 (1998) 103–118

The Journal of Systems and Software

Human–computer interaction: Interdisciplinary roots and trends

H. Rex Hartson [1]

Department of Computer Science, Virginia Tech, Blacksburg, VA 24061-0106, USA

Received 1 November 1996; received in revised form 22 January 1997; accepted 23 April 1997

Abstract

Methodology, theory, and practice in the field of Human–Computer Interaction (HCI) all share the goal of producing interactive software that can be used efficiently, effectively, safely, and with satisfaction. HCI is cross-disciplinary in its conduct and multi-disciplinary in its roots. The central concept of HCI is usability, ease of use plus usefulness. Achieving good usability requires attention to both product and development process, particularly for the user interaction design, which should serve as requirements for the user interface software component. This paper reviews some of the theory and modeling supporting the practice of HCI, development life cycles and activities, and much of the practice that constitutes "usability engineering". Future application areas of interest in HCI include new interaction styles, virtual environments, the World Wide Web, information visualization, and wearable computing.

1. Introduction

Human–Computer Interaction (HCI) is a field of research and development, methodology, theory, and practice, with the objective of designing, constructing, and evaluating computer-based interactive systems – including hardware, software, input/output devices, displays, training and documentation – so that people can use them efficiently, effectively, safely, and with satisfaction. HCI is cross-disciplinary in its conduct and multi-disciplinary in its roots, drawing on – synthesizing and adapting from – several other fields, including human factors (e.g., the roots for task analysis and designing for human error in HCI), ergonomics (e.g., the roots for design of devices, workstations, and work environments), cognitive psychology (e.g., the roots for user modeling), behavioral psychology and psychometrics (e.g., the roots of user performance metrics), systems engineering (e.g., the roots for much pre-design analysis), and computer science (e.g., the roots for graphical interfaces, software tools, and issues of software architecture).

1.1. Importance of usability

The entire field of HCI shares the single goal of achieving high *usability* for users of computer-based systems. Not fuzzy and vague as it is sometimes perceived, usability is tangible and can be quantified. Usability can be broadly defined as "ease of use", including such measurable attributes as learnability, speed of user task performance, user error rates, and subjective user satisfaction (Hix and Hartson, 1993a; Shneiderman, 1992). However, an easy-to-use system that does not support its users' needs, in terms of functionality, is of little value. Thus, usability has evolved toward the concept of "usability in the large" – ease of use, *plus* usefulness.

For expert users, usefulness is perhaps the primary usability factor, the factor that provides immediate access and affordance to the functionality without getting in the way. This does not match the usual concept of usability, more typically one that embraces learnability and understandability for novice and casual users. This kind of usability is, however, still defined by the measures of productivity, performance, and satisfaction; it is just that this class of "power users" has different criteria for these metrics. This maxim, originally intended for people, is also a good model for user interface designs: "Lead, follow, or get out of the way". A good interface design leads the novice user through task performance, follows intermediate user actions with informative feedback, and gets out of the way of expert users.

Despite many research advances in interactive computer systems, usability barriers still obstruct access to, and blunt effectiveness of, the power of computing, disenfranchising and disenchanting users across our

[1] Tel.: +1 540 231 4857; fax: +1 540 231 6075; e-mail: hartson@vt.edu.

0164-1212/98/$ – see front matter
PII: S0164-1212(98)10026-2

whole society. As a result, our nation fails to accrue the potentially enormous returns of our collective investment in computing technology. These barriers impede human productivity and have a profound impact on computer users in business, government, industry, education, and indeed, the whole nation.

In the not-too-distant past, computer usage was esoteric, conducted mostly by a core of technically-oriented users who were not only willing to accept the challenge of overcoming poor usability, but who sometimes welcomed it as a barrier to protect the craft from uninitiated "outsiders". Poor usability was good for the mystique, not to mention job security. Now, unprecedented numbers of people use computers and the user interface is often the first thing people ask about when discussing software. *To most users, the interface is the system*; communication with the system has become at least as important as computation by the system.

The goals of most organizations include increased employee and organization productivity, decreased employee training costs, decreased employee work errors, and increased employee satisfaction. These are also exactly the benefits of achieving high usability in user interfaces. Too often, especially in government and large businesses, training is used as a costly substitute for usability, and almost as often it fails to meet these goals. Attention to usability by developers no longer requires justification in most quarters: "Usability has become a competitive necessity for the commercial success of software" (Butler, 1996).

1.2. Product and process

Achieving good usability requires attention to both product and process. The *product*, in this case, is the content of the user interaction design and its embodiment in software. An effective *process* for developing interaction design is also important, and a poor understanding of this process is often responsible for lack of usability in the product. While state-of-the-*art* user interaction development processes are based on formative usability evaluation within an iterative cycle, much of the state-of-the-*practice* is fundamentally flawed in that remarkably little formal usability evaluation is performed on most interactive systems. This is generally changing now, in many industrial settings. However, ensuring usability remains difficult when evaluation, because of real or perceived costs, is not standard practice within interactive software development projects.

1.3. User interaction vs. user interface software

Developers attempting to incorporate usability methods into their development environments often refer to their usability efforts in terms of "evaluating software" or "evaluating user interface software". There are many reasons for evaluating *software*, but usability is not one of them. In the terminology of the simplistic diagram below, usability is seated within the design of the *user interaction component* of an interactive system and not within the *user interface software component*.

Development of the user interface	
Development of user interaction component	Development of user interface software component

Development of the interaction component, toward which most HCI effort is directed, is substantially different from development of the user interface software. The view of the user interaction component is *the user's perspective* of user interaction: how it works, how tasks are performed using it, its look and feel and behavior in response to what a user sees and hears and does while interacting with the computer.

In contrast, the user interface software component is the programming code by which the interaction component is implemented. *The user interaction component design should serve as requirements for the user interface software component*. Design of the user interaction component must be given attention at least equal to that given the user interface software component during the development process, if we are to ensure usability in interactive systems.

The overview of HCI topics, issues, and activities that follows is, loosely divided into theory, interaction techniques, development methods, and future trends. Reflecting its diverse roots, HCI is host to activity in many topical areas, some of which are reviewed here. An attempt has been made to capture a broad, inclusive cross-section of a very dynamic field, but this paper is not intended to be an exhaustive survey, and no claims are made for completeness.

2. Theory

HCI theory has its avid proponents. If the proportion of literature devoted to theory is to be taken as an indication, theory plays a strong role in HCI, but in fact theory has not seen broad, direct application in the practice of HCI. Indeed, some theory does offer a basis for user interface design guidelines, but the bulk of the guidelines, in fact, arose mostly out of practice rather than theory. Although cognitive principles underlie many of the guidelines, they originated more in the experience and educated opinions of experts and practitioners. Also, theory of work activity and of task analysis are embodied in techniques such as contextual inquiry, and cognitive theory is explicitly the basis for claims analysis (Carroll and Rosson, 1992, for example).

However, it is still the case that the bulk of real-world practice could benefit a great deal more from theory.

Much of the theory that is extant in HCI comes to it from cognitive psychology (Barnard, 1993; Hammond et al., 1987). Norman's theory of action expresses, from a cognitive engineering perspective (Norman, 1986), human task performance – the path from goals to intentions to actions (inputs to the computer) back to perception and interpretation of feedback to evaluation of whether the intentions and goals were approached or met. The study of learning in HCI (Carroll, 1984; Draper and Barton, 1993), as well as Fitts Law (relating cursor travel time to distance and size of target) (MacKenzie, 1992), also have their roots in cognitive theory.

2.1. Task analysis

In order to design a user interface (or any system) to meet the needs of its users, developers must understand what tasks users will use the system for and how those tasks will be performed (Diaper, 1989). Because tasks, at all but the highest levels of abstraction, involve manipulation of user interface objects (e.g., icons, menus, buttons, dialogue boxes), tasks and objects must be considered together in design (Carroll et al., 1991). A complete description of tasks in the context of their objects is a rather complete representation of an interaction design. The process of describing tasks (how users do things) and their relationships (usually in a hierarchical structure of tasks and subtasks) is called *task analysis* and comes to HCI primarily from human factors (Meister, 1985). There are various task analysis methods to address various purposes. In HCI the primary uses are to drive design and to build predictive models of user task performance. Because designing for usability means understanding user tasks, task analysis is essential for good design; unfortunately, it is often ignored or given only minimal effort.

2.2. Models of human information processing

A significant legacy from cognitive psychology is the model of a human as cognitive information processor (Card et al., 1983). The Command Language Grammar (Moran, 1981) and the keystroke model (Card and Moran, 1980), which attempt to explain the nature and structure of human–computer interaction, led directly to the Goals, Operators, Methods, and Selection (GOMS) (Card et al., 1983) model. GOMS-related models, quantitative models combining task analysis and the human user as an information processor, are concerned with predicting various measures of user performance – most commonly task completion time based on physical and cognitive actions of users, with place holders and estimated times for highly complex cognitive actions and tasks. Direct derivatives of GOMS include NGOMSL (Kieras, 1988), and Cognitive Complexity Theory (CCT) (Kieras and Polson, 1985; Lewis et al., 1990), the latter of which is intended to represent the complexity of user interaction from the user's perspective. This technique represents an interface as the mapping between the user's job-task environment and the interaction device behavior.

GOMS-related techniques have been shown to be useful in discovering certain kinds of usability problems early in the life cycle, even before a prototype has been constructed. Studies (e.g., Gray et al., 1990) have demonstrated a payoff in some kinds of applications where the savings of a number of user actions (e.g., keystrokes or mouse movements) can improve user performance enough to have an economic impact, often due to the repetitiveness of a task.

Nonetheless, these models have not achieved widespread application within the tight constraints of industrial schedules and budgets, because of the labor-intensiveness of producing and maintaining these relatively formal and structured task representations, the need for specialized skills, and because of the difficulty in competing with the effectiveness of user-based usability evaluation using an interface prototype. Further, these techniques generally do not take individual differences in user classes into account and are often limited to expert, error-free behaviors (not representative of the typical user). In any case, it is generally agreed that this kind of analytic approach to usability evaluation cannot be considered a substitute for empirical formative evaluation – usability testing of a prototype with users in a lab or field setting (see Section 4.2.3).

2.3. Human work activity

Another area feeding HCI theory and practice is sometimes called work activity theory (Bødker, 1991; Ehn, 1990). Originating in Russia and Germany and now flourishing in Scandinavia (where it is, interestingly, related to the labor movement), this view of design based on work practice situated in a worker's own complete environment has been synthesized into several related mainstream HCI topics. For example, "participatory design" is a democratic process based on the argument that users should be involved in designs they will be using, and that all stakeholders, including and especially users, have equal inputs into interaction design. Muller and others have operationalized participatory design in an approach called PICTIVE (Muller, 1991), which supports rapid group prototype design using Post-It™ notes, marking pens, paper, and other "low technology" materials on a large table top.

This interest in design driven by work practice in context has led to the eclectic inclusion in some HCI practice of ethnography, an investigative field rooted in

anthropology (LeCompte and Preissle, 1993), and other hermeneutic (concerned with ways to explain, translate, and interpret perceived reality) approaches as qualitative research tools for extracting design requirements. Contextual inquiry/design (Wixon et al., 1990) is an example of an adaptation of this kind of approach, where design and evaluation are conducted collaboratively by users and developers, while users perform normal work tasks in their natural work environment. Much of this collaboration is based on interviews that seek to make implicit work practice more explicit and to draw out structure, language, and culture affecting the work.

The task artifact framework of Carroll and Rosson (1992) and, to some extent, scenario-based design follow an ethnographic focus on task performance in a work context. Scenarios are concrete, narrative descriptions of user and system activity for task performance (Carroll, 1995). They describe particular interactions happening over time, being deliberately informal, open-ended, and fragmentary. Scenarios often focus on interaction objects, or artifacts, and how they are manipulated by users in the course of task performance.

2.4. Formal methods

While not theory per se, formal methods have been the object of some interest and attention in HCI (Harrison and Thimbleby, 1990). The objectives of formal methods – precise, well-defined notations and mathematical models – in HCI are similar to those in software engineering. Formal design specifications can be reasoned about and analyzed for various properties such as correctness and consistency. Formal specifications also have the potential to be translated automatically into prototypes or software implementation. Thus, in principle, formal methods can be used to support both theory and practice; however, they have not yet had an impact in real-world system development and their potential is difficult to predict.

3. Devices, interaction techniques, and graphics

In contrast to theory, the influence of interaction devices and their associated interaction techniques represent a practical arena of real-world constraints, as well as hardware design challenges. ''An *interaction technique* is a way of using a physical input/output device to perform a generic task in a human–computer dialogue'' (Foley et al., 1990). A very similar term, *interaction style*, has evolved to denote the behavior of a user and an interaction object (e.g., a push button or pull-down menu) within the context of task performance. In practice, the notion of an interaction technique includes the concept of interaction style plus full consideration of internal machine behavior and software aspects. In the context of an interaction technique, an interaction object (and its supporting software) is often referred to as a ''widget.'' Libraries of widgets, software that supports programming of graphical user interfaces (GUIs), are an outgrowth of operating system device handler routines used to process user input–output in the now ancient and impoverished interaction style of line-oriented, character-cell, text-only, ''glass teletype'' terminal interaction. At first, graphics packages took interaction beyond text to direct manipulation of graphical objects, eventually leading to new concepts in displays and cursor tracking. Of course, invention of the mouse and advent of the Xerox Star and the Lisa and Macintosh by Apple accelerated the evolution of the now familiar point and click interaction styles. It is not surprising that many of the computer scientists who developed early graphics packages were also the ones who introduced GUI interaction techniques as part of their contribution to the HCI field (Foley et al., 1990; Foley and Wallace, 1974). Standardization, to some extent, of interactive graphical interaction techniques led to the widgets of today's GUI platforms and corresponding style guides intended for ensuring compliance to a style, but sometimes mistakenly thought of as usability guides.

This growth of graphics and devices made possible one of the major breakthroughs in interaction styles – direct manipulation (Hutchins et al., 1986; Shneiderman, 1983; Weller and Hartson, 1992) – changing the basic paradigm of interaction with computers. Unlike previous command-line-oriented interaction in which users plan tasks in terms of hierarchies of goals and subgoals, entering a command line for each, direct manipulation allows opportunistic and incremental task planning. Users can try something and see what happens, exploring many avenues for interactive problem solving. This kind of opportunistic interaction is also called display-based interaction (Payne, 1991).

4. Development methods and the relation to software engineering

The difference between user interaction and user interface software, mentioned in the introduction, results in a need for separate and fundamentally different development processes for the two components of a user interface.

4.1. Development life cycles

Studies deriving principles for user interaction development (e.g., (Gould et al., 1991) vary, but all agree that interaction development must involve usability evaluation. Just adding some kind of ''user testing'' to an existing software process is not enough, however.

Usability comes from a complete process, a process that ensures usability and one that attests to when usability has been achieved (Hix and Hartson, 1993a). Most researchers and practitioners also agree that an interaction development process must be iterative, unlike the phase-oriented, "waterfall" method (for example) for software development. Although software can be correctness-driven, user interaction design, because of infinite design possibilities and unpredictable, dynamic, and psychological aspects of the human user, must be self-correcting. Thus, interaction development is an essentially iterative process of design and evaluation, one that must, in the end, be integrated with other system and software life cycles. Within this cycle, the interaction design is an iteratively evolving design specification for the user interface software. The star life cycle (Hartson and Hix, 1989) for interaction development explicitly acknowledges these differences from software development, being unequivocally iterative, allowing the process to start with essentially any development activity and to proceed to any other activity before the previous activity is completed, with each activity informing the others.

4.2. Development activities

4.2.1. Design and design representation

Design is closely coupled to, and driven by, early systems analysis activities such as needs, task, and functional analyses. Good interaction design involves early and continual involvement of representative users and is guided by well-established design guidelines and principles, built upon the concept of user-centered design (Norman and Draper, 1986). Design guidelines address issues such as consistency, use of real-world metaphors, human memory limits, screen layout, and designing for user errors. Additionally, designers are expected to follow style guides (less oriented toward usability than toward compliance with some "standard" style) in their use of widgets.

Although more recently some guidelines enjoy the support of empirical studies, guidelines have typically been scattered throughout the literature, based mostly on experience and educated opinion. In a classic work, Smith and Mosier (1986) compiled guidelines for character-cell, textual interface design. Others (Mayhew, 1992; Shneiderman, 1992) have followed to help cover graphical interfaces.

Many practitioners believe it is enough to know and use interface design guidelines, possibly in addition to an interface style guide (e.g., for Windows™). Experience, however, has shown that guidelines and style guides do not eliminate the need for usability evaluation. Experience has also demonstrated that, although guidelines are not difficult to learn as factual knowledge, their effective application in real design situations is a skill acquired only through long experience.

The creative act of design must also be accompanied by the physical act of capturing and documenting that design. Although many *constructional* techniques exist for representing software aspects of interface objects, *behavioral* representation techniques are also necessary for communicating, among developers, the interaction design from a behavioral task and user perspective. The User Action Notation (UAN) (Hartson and Gray, 1992; Hartson et al., 1990) is one such technique. The UAN is a user- and task-oriented notation that describes the behavior of a user and an interface during their cooperative performance of a task. The primary abstraction of the UAN is a user task – a user action or group of temporally-related user actions performed to achieve a work goal. A user interaction design is represented as a quasi-hierarchical structure of asynchronous tasks. User actions, interface feedback, and internal state information are represented at various levels of abstraction in the UAN. In addition to design representation, design rationale (MacLean et al., 1991) is captured to record and communicate the history and basis for design decisions, to reason about designs, and to explore alternatives.

4.2.2. Prototyping

Rapid prototypes of interaction design are early and inexpensive vehicles for evaluation that can be used to identify usability problems in an interaction design before resources are committed to implementing that design in software. Much interest has been focused on low-fidelity prototypes (e.g., paper and pencil). Counter to intuition, low-fidelity prototypes have allowed developers to discover as many usability problems as found using interactive computer-based prototypes (Virzi et al., 1996). Paper prototypes are most useful early in the life cycle, because they are more flexible in exploring variations of interaction behavior, at a cost of less fidelity in appearance. Later in the life cycle, changes made to the behavior of a coded prototype are more expensive than changes in appearance. Almost all projects eventually move to computer-based rapid prototypes for formal usability evaluation.

4.2.3. User-based evaluation

Summative evaluation is used to make judgments about a finished product, to gauge the level of usability achieved and possibly to compare one system against another. In contrast, *formative evaluation* – the heart of the star life cycle – is used to detect and fix usability problems before the interaction design is coded in software (Hix and Hartson, 1993a; Hix and Hartson, 1993b; Nielsen, 1993), aiding in the improvement of an interaction design while a product is still being developed. For formative evaluation, unlike summative evaluation, statistical significance is not an issue. Formative evaluation relies on both quantitative and qualitative data.

The quantitative data are used as a gauge for the process – as an indication that usability is, indeed, improving with each design iteration and as a way to know when to stop iterating. Borrowing an adage from software engineering (and probably other places before that), "If you can't measure it, you can't manage it." The instruments used to quantify usability include benchmark tasks and user questionnaires. Benchmark tasks, drawn from representative and mission-critical tasks, yield objective user performance data, such as time on task and error rates (Whiteside et al., 1988). Questionnaires yield subjective data such as user satisfaction (Chin et al., 1988). In analysis of these quantitative data, results are compared against pre-established *usability specifications* (Whiteside et al., 1988) – operationally defined and measurable goals, used as criteria for success in the interaction design.

Even more valuable than these quantitative data are the qualitative data gathered in usability evaluation. Identification of *critical incidents* – occurrences in task performance that indicate a usability problem – are essential in pin-pointing problems in a design. Verbal protocol (capturing users' thinking aloud) helps designers understand what was going through a user's mind when a usability problem occurred, in order to ascertain its causes and to offer useful solutions.

These quantitative and qualitative data come typically from lab-based evaluation involving users as "subjects". While very effective, this process can be expensive. The need for faster, less costly usability methods has led to approaches, such as *discount usability engineering* (Nielsen, 1989), that trade off less than perfect and complete results for a lower cost. *Inspection methods* (Nielsen and Mack, 1994) use systematic examinations of design representations, prototypes, or software products. *Cognitive walkthroughs* (Lewis et al., 1990; Wharton et al., 1992) and *claims analysis* (Carroll and Rosson, 1992) are effective inspection methods, especially early in development, but can still be labor-intensive and require special training, intimidating to developers in search of cost-effective methods. *Heuristic evaluation* (Nielsen, 1992; Nielsen and Molich, 1990), reviewing compliance of an interaction design to a checklist of selected and generalized guidelines, is an even less expensive inspection method, but is limited by the scope of guidelines used.

Inspection methods are effective at finding some kinds of usability problems, but do not reliably pinpoint all types of problems that can be observed in lab-based testing. In fact, lab-based usability evaluation remains as the yardstick against which most new methods are compared in formal studies. Most real-world development organizations continue to be willing to pay the price for extensive lab-based usability evaluation because of its effectiveness in helping them identify and understand usability problems, their causes, and solutions.

4.3. Usability engineering

Many HCI practices, such as the employment of usability specifications and various kinds of evaluation, have been gathered under the banner of *usability engineering* (Nielsen, 1993). This is a good appellation because it includes a concern for cost in the notion of *discount usability methods* (Nielsen, 1989), the practical goal of achieving specifications and not perfection, and techniques for managing the process. This latter is important because iterative processes are sometimes perceived by management as "going around in circles", not attractive to a manager with a limited budget and dwindling production schedule.

Usability specifications provide this essential management control for the iterative process. The quantitative usability data are analyzed in each iteration and results are compared with the usability specifications, allowing management to decide if iteration can stop. If the specifications are not met, data are assessed to weigh cost and severity or importance of each usability problem, assigning a priority ranking for designing and implementing solutions to those problems which, when fixed, will give the largest improvement in usability for the least cost.

4.4. Development tools

Almost any software package that provides support for the interface development process can be called an *interface development tool*, a generic term referring to anything from a complete interface development environment to a single library routine (Myers, 1989, 1993). New software tools for user interface development appear with increasing frequency.

Interface development tools can be divided into at least four types (Hix and Hartson, 1993a). *Toolkits* are libraries of callable routines for low-level interface features and are often integrated with window managers (e.g., X, Windows™). *Interface style support tools* are interactive systems that enforce a particular interface style and/or standards (e.g., OSF Motif, Common User Access). *User interface management systems (UIMS)* are development environments that can include both prototyping and run-time support, with the goal of allowing developers to produce an interface implementation without using traditional programming languages. Of these groups, the UIMSs perhaps are the most interesting, have the most potential, and suffer the most difficult technical problems (Myers, 1995).

These first three categories primarily address user interface software. A fourth category, *interaction development tools*, provides interactive support for user interaction development. Of all the interaction development activities, the one most commonly supported by

tools in this group is formative evaluation (Hix and Hartson, 1994; Macleod and Bevan, 1993).

Although tools now exist on many programming platforms to lay out objects of a user interface quickly and easily, usability problems are not necessarily addressed by adding this kind of technology to the process; many interface development tools are potentially a faster way to produce poor interfaces.

5. Cost justification

Economic justification for usability effort in interactive system development is now beginning to be established (Bias and Mayhew, 1994). Broad acceptance in business and industry requires further demonstration of a return on investment; documented cases and success stories are essential. The bottom line answer is that usability engineering does not add *overall* cost, for two reasons: Usability does not add as much cost to the development process as many people think, and good usability saves many other costs.

Considering cost added to the process, one must realize that any added cost is confined. Interaction development is a small part of total system development. It occurs early in the process, when the cost to make changes is still relatively low, and mainly impacts only a prototype, not the final system software.

Considering cost savings due to good usability, it is easy to establish that poor usability is costly, and good usability is all about saving cost. Usability is simply good business. The most expensive operational item in an interactive system is the user. People who develop software are concerned with the cost of development, but the people who buy and use a software application are concerned with the costs of usage. Development costs are mostly one-time, while operational costs – such as training, productivity losses, help desks and field support, recovery from user errors, dissatisfied employees, and system maintenance costs (the cost of trying to fix problems *after* release) – accrue for years.

Unless the net of analysis is cast broadly enough, the problem with cost–benefit analysis is that one group pays development costs and another group gets the benefits. It is necessary to include the people who purchase computing systems, the ones who think about which costs more: user-based tasks that are quick, efficient, and accurate, or error-prone tasks that take more time? confused users, or confident and competent users?

Beyond this kind of argumentation, used in software engineering for years, substantial measurable economic advantage can be accrued from usability. Case studies have demonstrated that large sums of real money can be saved by increasing user (employee) productivity alone (Bias and Mayhew, 1994). In the end, these are the cases that will make the difference.

Mantei and Teorey (1988) estimate that early changes can cost less than one-fourth of the cost of changes made after installation. More recently a "powers of 10" rule for the cost of changes has become common in the lore. By this rule a given change made in a low fidelity paper prototype at a cost of $1 is estimated to cost $10 in a computer-based prototype, $100 in an early coded implementation, and $1000 after the system is deployed.

Mayhew and Mantei (1994) offer a framework with examples for estimating the costs of adding a usability engineering program to a software development project, including the costs of a usability lab setup, user interviews, questionnaires, prototype testing, style guide preparation, purchase of software tools, and prototype construction and revision. Examples of costs savings for increased productivity, decreased training, decreased errors, and decreased late design changes are also given.

A simple example of cost savings from productivity increases is taken (and sanitized) from our own work with a large real-world government agency. Consider the economic impact of making just one design change affecting just one kind of transaction in just one system, a large distributed system with 75,000 users. The average number of times this transaction is made by each user in a day is 20. That is a total average of 1,500,000 occurrences of this transaction per day. The average user time per transaction is documented at 5–20 min, and the average time saved per transaction, due to improved usability, has been measured at 30 s (modest compared to typical cases which can save minutes). Given an average fully loaded hourly rate of $25.00 for the employees who use this system, the yearly savings are enormous, being equal to

$$75,000 \text{ users} \times 20 \text{ transactions/user-day} \times 0.5 \text{ min/transaction} \times 230 \text{ days/year} \times \$25/\text{h} \times 1 \text{ h}/60 \text{ min}$$

$$= \$71,875,000.00!$$

Karat (1994) goes beyond these simple cost–benefit analyses in presenting a business case approach to usability cost justification using estimates of increased sales and revenues and calculations based on more sophisticated measures such as net present value of investment and internal rate of return.

6. The future

HCI is a relatively young and broadly diverse field with a rapidly growing impact on the world of computing. Usability is now recognized as the crucial driving force for user interface design and evaluation. The future of HCI in this context can be viewed from a perspective of product and process.

6.1. The future of HCI in products/applications

A rich part of the future of HCI is in its application areas, which are growing more rapidly than the HCI methods needed for their development. As an example, it is unlikely that usability methods developed for desktop applications will apply directly to virtual environments, one of the most exciting areas of application development. Despite intense and wide-spread research in virtual environments, very little work has been applied toward developing the usability methods that will be required to evaluate this new technology – a necessary coupling if virtual environments are to reach their full potential. Similarly, groupware and computer-supported cooperative work (CSCW) (Baecker, 1993; Grudin, 1994), multimedia (Blattner and Dannenberg, 1992), hypermedia, and interface access for the disabled and impaired (Williges and Williges, 1995) will require development of new methods for design and usability evaluation. Educational technology for the classroom, the Web, and the home is emerging as a giant application area. Perhaps nowhere is usability more important than in the discipline of education where understanding and communication of concepts and ideas is the stock and trade.

6.1.1. Beyond WIMP interfaces

We can expect interaction styles and techniques used in products and applications of the future to expand beyond the currently ubiquitous WIMP – windows, icons, menus, pointers – or desktop-style interface. While WIMP interfaces have provided a great step forward for interfaces in static situations (e.g., word processing, spreadsheets), innovative interaction techniques that go beyond the WIMP paradigm are necessary to meet user interface needs of demanding, real-time, high-performance applications such as those found in military applications, medical systems, "smart road" applications, and so on. Researchers are promoting a greatly expanded vision of interaction beyond the limited interaction styles now available via just keyboard and mouse, including extensions to current work in graphic and visual displays (Mullet and Sano, 1995), use of hands and feet (Buxton, 1986), eye movement (Jacob, 1993), haptic (touch) feel and force feedback (Baecker et al., 1995), audio and sound (Brewster et al., 1993; Gaver and Smith, 1995), voice (McCauley, 1984), and stylus and gesture (Goldberg and Goodisman, 1995).

6.1.2. The Web

6.1.2.1. Problems and some attempts at solutions. The Internet, the World Wide Web, and cyberspace are technological and sociological frontiers with many analogies to the frontier that was our West over a century ago – lawlessness and disorganization, with exploration, expansion, and exploitation in every direction. These incredibly fast-growing application domains bear new kinds of usability challenges. Users having trouble with an interactive system often cannot find solutions in user manuals or on-line help. They are more likely to ask a friend, colleague, or co-worker for help. This strategy can work in a local setting where there are other users. However, users of the Internet and the Web will often be remote and distributed, the network being their workplace. For these isolated users, who are less able to tolerate poor user interfaces and who will abandon applications that they find too difficult to learn and use, there is often no one to ask when things go wrong, and usability will have a large impact on productivity and user satisfaction. For this large-scale environment with its diversity of user types and characteristics, its variety of application types, and potential user isolation, usability takes on special importance.

Fortunately, some help in designing for usability on the Web is on the way. Nonetheless, it appears to be arriving in the same profusion of material, with the same disorganization and the same varying quality as found on the Web itself. Ironically, most of this help resides on the Web, and it mostly comes in the form of style guides and ad hoc prescripts.

An example of one of the better sources of Web design information (as of this writing) is the Web site of the Yale C/AIM WWW Style Manual at:

http : //info.med.yale.edu/caim/manual

This manual contains tips for designing Web pages, improving site structure, aiding navigation in hyperspace, and helping to guide users to important information. More general discussion (and conjecture) can be found at the site called "WebHCI: A guide to HCI issues of the Web," which uses resources provided by the ACM SIGCHI group, and is located at:

http : //www.acm.org/sigchi/webhci/index.html

Jakob Nielsen's "Alert Boxes" (moved to his personal Web site at useit.com from the SunSoft Alertbox pages) are linked to from the WebHCI pages and offer examples of some of the best of this kind of topical discussion. A good example is the page on WebTV usability, at:

http : //www.useit.com/alertbox/9702a.html

which reviews remote control design, wireless keyboard options, learnability issues, screen size, and making the Web more accessible to users with disabilities. Other Alertboxes address topics such as the "Top Ten Mistakes in Web Design" and guidelines for multimedia Web design.

The WebHCI pages contain reviews of good and bad Web interface designs (to learn by example) and refer to a number of design awards, of which the weekly Design Excellence Award from the Internet Professional Publishers Association (IPPA) is typical. The research link

on the WebHCI pages connects to a list of conferences and workshops on Web design and usability. Microsoft also maintains a Web site called the Microsoft Internet Workshop at:

http : //www.microsoft.com/workshop/

that offers help for authoring, design, and programming of Web sites and Internet applications.

While some information on design and usability is quite good, as one might expect with an emerging area as rapidly growing and changing as the Web, much of the information addressing usability on the Web suffers from the same problems as other Web-based material – lack of controls and reviews, instability of content and location, and prodigious variation in quality. We do not yet know what tools or techniques will be required for the future to help Web users looking for more authoritative or proven methods to sort out the more organized and more carefully reviewed information that has been tempered with experience and empiricism.

This kind of control, however, is just what Microsoft was trying to accomplish with its October, 1996 conference entitled, "Designing for the Web: Empirical Studies." More information, including the conference proceedings, on the Microsoft conference can be viewed at:

http : //microsoft.com/usability/webconf.htm

As testimony to its growth and importance, Web conferences and journals are springing up. In June 1997, the US West-sponsored Conference on Human Factors and the Web will be held in Denver, already its third annual meeting. The most recent ACM Conference on Human Factors in Computing Systems (CHI'97, in Atlanta) featured a special interest group meeting on Measuring Website Usability. An entire new journal, *The World Wide Web*, is devoted to research, case studies, surveys, and tutorials related to the Web

(http : //manta.cs.vt.edu/www/).

Compared to other application areas, very little found on the Web has had the benefit of usability (or other) evaluation. Probably more than in any other area of computing, the paradigm is to "put it out there and see what happens". For those who do seek to evaluate their Web designs, present methods for usability evaluation are not completely adequate. Better methods are needed, especially methods tailored to the specific nature of the Web as a user interface (e.g., much more attention to navigation and the phenomenon of getting lost in hyperspace).

6.1.2.2. Information searching: The major task. By far, the most significant kind of Web usage involves searching for information. Shneiderman, Byrd, and Croft propose a framework for common structure and terminology for information searching with the aim of faster learning for users, with increased comprehension and better control at:

http : //www.dlib.org/dlib/january97/retrieval/01shneiderman.html

(in D-Lib Magazine, an on-line magazine devoted to research on digital libraries). Shneiderman (1997) identifies four types of Web searching: specific fact-finding for a known item, extended fact-finding, open-ended browsing, and exploration of availability (e.g., "What genealogy information is at the Library of Congress?"). His Object/Actions Interface Model is adapted to the Web by focusing on four design components: under the heading of task – information objects/structure and information actions for navigation, and under the heading of interface – metaphors for information objects and affordances for actions.

6.1.2.3. Amateur and professional Web pages. At the outset, design for presentation of information content on the Web has mostly been an amateur undertaking. The "amateur side" of the Web, still the bulk of the Web today, includes individuals or groups who are often professionals in the information industry, but who are not trained specifically in design and/or usability. In contrast, a more "professional side" is emerging. Organizations have begun to invest in stylish, attractive, attention-catching, and interesting professionally-designed Web pages. Corporations and other institutions are willing to expend resources to employ professional designers, especially in graphics and communications – the same skills that have made the commercial advertising world what it is (for better or worse). The result will be increasingly high production quality, rivaling that of existing TV commercials.

The Web today is, by and large, information-oriented and not entertainment-oriented, aside from the entertainment value that accompanies searching for and finding interesting information. While still information-oriented on the surface, the professional side of the Web will also be moving toward more entertainment, or at least "edutainment", through multimedia (e.g., audio and video clips, extensive graphics and even animation) where available bandwidth can support it.

The professional side of the Web is, in the main, a commercial side. Expected commercial gain is usually the justification for resources invested in production quality. But commercial objectives are not limited to the professional side. Many small-scale business interests are represented on the amateur side of the Web. The difference is mainly one of tone – the amateur side is usually chatty, informal, and often personal, the style of the roots of the Web and the Internet. On the amateur side, everyone is an author, everyone is a designer, and everyone has an opinion. The potential to propagate poor usability is enormous. On the professional side we

will at least see more slickness and more formality, more obviously expensive production. Whether that translates into better usability is yet to be seen.

6.1.2.4. Enter Web-TV. It is surely the professional side of the Web on which Sony and Philips (and others) are banking in their development of Web-TV, a Web-browsing device using a phone modem connection to the Internet, displaying on a TV screen, and using a remote control for selecting and clicking. Typed input can be made via an optional keyboard with infra-red connections or a "soft" on-screen keyboard. Although most usability issues of Web-TV are yet to be discovered, we do know that usage conditions are likely to be quite different from PC-based Web usage. Much of this difference is due to the relatively low bandwidth via phone modem (until high band rate access to the Internet becomes publicly available, for example, via TV cable) and the low resolution display viewed at a distance in a living room. One is compelled to ask whether a user sitting ten feet or more from the screen, in a living room full of people, is an environment that (for example) supports the task of composing a letter for email?

Finally, since TV users are almost exclusively entertainment oriented, Web-TV users may, quite likely, be looking for entertainment, too. It is not difficult to envision Web-TV applications connecting the commercial world of TV and the information world of the Web. Sponsors of TV shows are already giving their Web addresses in TV commercials. Now viewers are only a click away from details of a TV show sponsor's story and information on products that catch their interest.

6.1.2.5. Web URLs as references? As an aside, this field finds itself increasingly confronted with a dilemma stemming from the need to cite Web page URLs in published work. Unfortunately, URLs are not archival or permanent, and few, if any, publishers recognize a Web URL as an acceptable reference type, with none providing standard formats for their citation or reference. Indeed, some of the URLs cited in this paper did change between the reviewed version and the final version. Some authors include a "most recently accessed" date as a "freshness indicator".

6.1.3. Information visualization

Information visualization is a well established area of research and application with many significant issues that cannot be surveyed in this space. The essence of visualization is graphical presentation of often large volumes of numeric (and other) data in order to make sense of its meaning. It is said, for example, that data indicating a hole in the ozone layer existed for almost ten years before an animated visualization revealed the problem in a way that could be interpreted.

Because design for visualization requires consideration of user cognition, researchers are attempting to move these design issues away from opinion and personal taste into psychological underpinnings. Psychophysical issues (e.g., perception and cognition) are strong determiners of usability, but they are rarely included in design analysis for user interfaces that feature information visualization. And that might be because, despite the research attention this area has received, we still lack an in-depth comprehension of the underlying human perceptual and cognitive mechanisms that govern how displays of information are visualized and understood.

One type of task that is particularly well suited for support by information visualization is search and identification, as commonly occurring in process control tasks, where the user must monitor, and react to, displays of processes and events. The visual search task involves scanning for and locating target objects or events. Identification involves semantic interpretation of the display objects and events. A good example of this kind of application is seen in air traffic control displays – a large-scale, complex, real-time process in which user performance can affect the safety of human lives.

Visualization involves encoding information abstractions using graphical "devices", visual display elements that include color hue, color saturation, flash rate, object shape, object size, position, and alphanumeric identifiers. Choosing the right graphical device to encode a particular information attribute turns out to be a surprisingly difficult task (Nowell et al., 1996). The challenge in interface design is to select graphical devices that will support the expected range of user tasks, while simultaneously supporting perceptual and individual differences of the user population.

Different kinds of data (e.g., nominal, ordinal, or quantitative) require different kinds of graphical devices. Christ (1975) and Wickens and Andre (1990) have compared graphical devices for visual search tasks, and Cleveland and McGill (1985) have ranked graphical devices for their ability to represent quantitative data. Mackinlay (1986) has extended rankings to other kinds of data.

Despite the fact that color is perhaps the most studied graphical device, relatively little is understood about how humans perceive and cognitively process color. Yet, it is known that color is a very effective graphical device for supporting search tasks, especially within high density displays (Christ, 1975, 1984; Smith and Thomas, 1964). Shape and size, as well as the display of letters and digits, are also useful for searching.

One of the most studied questions about color has to do with the number of colors that can be used in a display and still be discriminable. The answer seems to be about six (Christ, 1975; Shneiderman, 1992), especially in high density displays. More colors tend to re-

duce, rather than enhance, user performance. One of the problems with using color is that it has such cognitive dominance that color coding tends to reduce the effectiveness of most other graphical devices, especially object size, used in the same display (Christ, 1975). Conversely, object size affects perception of color. Further, color seems to be the device with the most variation in perception due to individual differences.

In sum, visualization of data is an enormously important area of application for usability in design, but many significant problems remain, leaving it as an area where design skills and intuition contribute at least as much to results as theory and principles. Readers interested in the scientific, communication-related, and aesthetic aspects of this topic should see the well-known books on graphical design for information display by Tufte (1983, 1990, 1997).

6.1.4. Virtual environments

6.1.4.1. Lack of connections to usability. Despite broad research in both fields separately, usability has not yet generally been brought to bear on the exciting new technology of virtual environments. Usability guidelines and heuristics for virtual environments are rare, tasks in virtual environments are not well understood, and methods for usability evaluation based on user observation in a lab setting often cannot be applied in the innovative physically interactive settings of virtual environments.

Virtual environment applications have interaction styles so radically different from ordinary user interfaces that traditional user-task-based evaluation methods may be neither appropriate nor effective. For example, three-dimensional navigation, wayfinding (Darken and Sibert, 1996a), distance estimation (Eggleston et al., 1996), object manipulation (Zhai and Milgram, 1993), and visual search tasks present challenges unseen in traditional GUI evaluations. Moreover, it is precisely the characteristics that are unique to virtual environment applications that are key to their usability. For example, perceived presence and perceived real world fidelity are critical within some virtual environments (Snow, 1996), but are not addressed at all in existing evaluation methods.

Nonetheless, the virtual environment community is becoming aware of the need for usability evaluation. Some guidelines have emerged for spatial input devices (Hinckley et al., 1994), as have hints for three-dimensional interface design and usability issues in feedback hardware (Hannaford and Venema, 1995). However, these approaches fail to address and integrate the complex inter-dependencies present in virtual environments among users, tasks, input devices, and output devices. The immediate goal is to increase the awareness of the necessity for virtual environment usability evaluation. The near-future goal of the usability community is to develop methods for virtual environment usability evaluation.

6.1.4.2. Virtual environments as an extension of direct manipulation. Inherently, virtual environments do not contain the same kind of interface primitives standard to GUIs (e.g., mouse-driven point-and-click manipulation of icons, buttons, menus). In much the same way that GUIs are direct manipulation extensions of textual command-line interfaces, virtual environments are a direct manipulation extension of the GUIs. More illusion (Laurel, 1986) and more engagement (Hutchins et al., 1986) are possible; there is more sense of task and less sense of the computer as intermediary. Yet, designers have often failed to take advantage of this opportunity, opting rather for users to interact with two-dimensional GUI objects such as pull-down menus hanging in three-dimensional space and operated with a hand or finger, for example via a "dataglove" (once again proving that a Fortran program can be written in C++). Expression of commands with finger and hand gestures can be very effective in virtual environments, if appropriate and natural to the task as in the case of a user holding a painter's palette in one hand while selecting a color with the other. However, arbitrary encodings of finger and hand gestures to computer commands frequently used in virtual environments is a throwback beyond GUIs, all the way back to command languages.

6.1.4.3. Presence and engagement. A major distinguishing feature of virtual environments is inclusion of an explicit concept of the presence of oneself. Studies have shown that the perception of presence in virtual environments can have a distinct effect on task performance (Snow, 1996). Tied to the presence of self is the problem of locomotion, moving about in virtual environments. Given the restricted real space usually provided for the physical actions of users, mapping user actions to virtual locomotion is an enduring problem (Darken and Sibert, 1996b). A limited variety and range of real motions must map to everything from the microscopic movements of gene splicing to crossing vast distances in space. Even in non-immersive (desktop) virtual environments movement and navigation can be slow and unnatural.

Perhaps more profoundly tied to the concept of self presence is the matter of avatars, graphical surrogates for the user, the user's body, movement, activity, and viewpoint (Benford, 1995) within the virtual environment. These animated on-screen stand-ins, controlled by the user in real time are especially important when multiple users are interacting as they share, explore, and work in the same virtual environment. They are seen by other users in the same part of the virtual environment and the interaction of two or more users is accomplished via the interaction of their avatars. This kind of inter-

action has potential for application in areas such as training, public education, and collaborative problem solving.

On the other side of the coin, a possible danger of avatars is confusion with so-called "intelligent" agents, anthropomorphic devices intended to help users by doing things for them. The role, value, and risks of anthropomorphism in user interfaces are controversial. Shneiderman (1993) has been a forceful voice against the use of anthropomorphism in computer interfaces. Shneiderman argues, and we agree, that most users want a comprehensible and predictable interface that gives them sense of direct and immediate control over the computer as an inanimate tool, a style that is fundamentally different from how most users interact with people.

In the context of presence and engagement, additional questions also arise about the degree of realism (e.g., photorealism of graphics, detailed observance of real-world physical laws) needed for a given task. For example, what types of applications really need stereopsis (true 3d imagery) (Hsu et al., 1994)? Realistic haptic feedback (feedback to the sense of touch) is a difficult problem, mostly due to the physics involved. We do not know yet what kinds of feedback (e.g., tactile, vibratory, force-feedback) are necessary for various types of tasks. Do environmental effects (e.g., wind, rain, heat, humidity) have a place in virtual environments? Possibly, for example, training soldiers to repair a helicopter in simulated blazing heat and blowing sand might enhance training transfer to the counterpart real-world situation. On the other hand, designers are still grappling with cases where presence is too intense for some users, resulting in motion sickness, dizziness, and disorientation.

Finally, shared virtual environments are certainly opening the door to social computing by providing richer ways for interpersonal interaction via the Internet and the Web. However, we must continue to question the long term social and psychological effects, if virtual environments become used as a substitute for normal social interaction. In any case, we have not even scratched the surface of usability methods for evaluating these complex interactions, unprecedented in the previous generation of GUIs.

6.1.4.4. Hardware devices. Much of the problem of providing realism falls to the designer of hardware devices. There are already far more physical interaction devices in virtual environments than in traditional GUI interfaces but, because physical interaction plays such an important role in virtual engagement, there is a need for even more and better interaction devices. We need to begin with research that yields a better understanding of which devices are best-suited (or ill-suited) to support the appropriate physical interaction for particular tasks. Certainly, the trend is toward involving the whole human body. Users soon reach a limit in virtual interaction, beyond which the mouse, joystick, and even the spaceball are indirect and unnatural. We will see more development of devices such as haptic gloves allowing users to grasp and point, eye tracking as a way for users to indicate objects of interest, and interaction devices that incorporate real-world props (tools with mass and "feel," such as a real scalpel for a surgery simulation) to support realism in interaction (Hinckley et al., 1994).

No matter what new hardware devices emerge, we can be confident that our appetite for more detail, more realism, and more effects will always outstrip our ability to provide higher performance real-time displays and the bandwidth to communicate them.

6.1.4.5. Application areas. Despite the enormous attention and resources directed toward virtual environments, there are only a few application areas that really seem to benefit from virtual environments. Examples include training and simulation where safety and accessibility are issues, architectural design and walkthrough, engineering design and data visualization, and, of course, entertainment. Clearly the potential exists for more. The test that will tell in time is how well we can separate glitz, glamour, and novelty from true utility.

6.1.5. Wearable computing

Finally, many technology forecasters have predicted that, in addition to desktop computers and network-based interaction we now know, the most significant area of future applications may be computing embedded within appliances, homes, offices, vehicles, and roads. Sometimes called *wearable computers*, these devices can be strapped on one's wrist or embedded in a shoe! A television news feature (CNN News, July 1996) described a project at MIT in which a person's shoes will, indeed, be instrumented so that as one gets the milk out for breakfast in the morning, sensors note the milk is getting low. Approaching the grocery store on the way home, the system speaks via a tiny earphone, reminding of the need to pick up some milk. (Perhaps a way can be found to deliver milk over the Internet?!)

The requirements for usability of desktop and other familiar systems will pale in comparison with the importance of usability in this new era of computing. That the average citizen will not tolerate training courses, user manuals, or on-line help to operate everyday objects such as refrigerators and automobiles compels designers to take seriously their responsibility for usability. Issues of social impact carry high risks if this kind of "every-citizen" interface is threatening, intimidating, or difficult to use. In successful designs, the computing component will be transparent, users not even thinking of themselves as users of computers. When human factors was first adapted to user interfaces (e.g., Williges et al., 1987)

ergonomics was largely filtered out. Interestingly, these new devices, combining hardware, software, and firmware as "appliances", will require a re-integration of ergonomics as a part of usability.

6.2. The future of HCI processes

Development of future HCI processes will struggle to keep pace with these new application areas and interaction styles. One area that is already changing among real-world system developers is the representation of roles and skills within interactive system development teams. Usability specialists, human factors engineers, and HCI practitioners are starting to take their long-overdue places alongside systems analysts and software engineers. These new roles imply the need for new kinds of training in HCI methods. In the future (it has already begun) these roles will also be joined by those with technical writing and documentation skills and especially by those with graphic and visual design skills (Mullet and Sano, 1995; Tufte, 1983); e.g., to use color effectively (Shubin et al., 1996) and to design icons, avatars, and rendered images.

It is also expected that a significant increase in future HCI activity will be applied to developing new methods. We are not only trying to develop more robust tools and techniques for current GUIs, but to understand what is needed for future interaction styles. There is especially an on-going need for new high-impact usability evaluation methods. High impact means cost-effective, applicable to a wide variety of application types (e.g., World Wide Web applications), applicable to many new interaction styles (e.g., virtual environments), and suitable for gathering usability data from remote and distributed user communities.

Among the approaches to remote evaluation just emerging, most either are limited to subjective user feedback (Abelow, 1993) or require expensive bandwidth to support video conferencing as an extension of the usability lab (Hammontree et al., 1994). A method based on user-assisted critical incident gathering (Hartson et al., 1996) has been proposed to bypass the bandwidth requirements required for full-time video transmission and to save in analysis costs.

Methods and software support tools are also in demand for boosting return-on-investment for resources committed to usability evaluation. Koenemann-Belliveau et al. (1994), p. 250, articulate this need: "We should also investigate the potential for more efficiently *leveraging* the work we do in empirical formative evaluation – ways to 'save' something from our efforts for application in subsequent evaluation work." Most of the time results from usability evaluation are applied only to specific usability problems in a single design. Database tools for information management of the results would accrue immediate gains in effective usability problem reporting (Jeffries, 1994; Pernice and Butler, 1995). More significantly, a usability database tool would afford some "memory" to the process, amortizing, through reuse of analysis, the cost of results across design iterations, and across multiple products and projects. Beyond organizational boundaries, a collective usability database could serve as a commonly accessible repository of a science base for the HCI community and a practical knowledge base of exemplar usability problems, solutions, and costs.

The future of HCI is both exciting and challenging. As we move beyond GUIs and as we develop new methods, the problems continue to increase. But the promise of these new products and processes will come to fruition through the efforts of a vital and prolific community of HCI researchers and practitioners.

Acknowledgements

Many thanks to Dr. Deborah Hix at Virginia Tech for her help in providing inputs and in reading of this paper. Thanks also to Joey Gabbard of Virginia Tech for help with the discussion of usability for virtual environments and Lucy Nowell of Virginia Tech for help with the discussion of information visualization. Appreciation is also expressed to the reviewers; I used some of their helpful words but cannot thank them by name since they are anonymous. This paper is adapted with permission from the author's contribution to TOWARD AN EVERY-CITIZEN INTERFACE FOR THE NATIONAL INFORMATION INFRASTRUCTURE. Copyright 1996 by the National Academy of Sciences. Courtesy of the National Academy Press, Washington, DC.

References

Abelow, D., 1993. Automating feedback on software product use. CASE trends, 15–17.

Baecker, R.M. (Ed.), 1993. Readings in Groupware and Computer-Supported Cooperative Work: Assisting Human–Human Collaboration. Morgan Kaufmann, San Francisco, CA.

Baecker, R.M., Grudin, J., Buxton, W.A.S., Greenberg, S., 1995. Touch, gesture, and marking. In: Baecker, Grudin, Buxton, Greenberg (Eds.), Readings in Human–Computer Interaction: Toward the Year 2000. Morgan Kaufmann, San Francisco, CA.

Barnard, P., 1993. The contributions of applied cognitive psychology to the study of human–computer interaction. In: Baecker, Grudin, Buxton, Greenberg (Eds.), Readings in Human–Computer Interaction: Toward the Year 2000. Morgan Kaufmann, San Francisco, CA.

Benford, S., 1995. User embodiment in collaborative virtual environments. In: Proceeedings of the CHI Conference on Human Factors in Computing Systems, 242–249.

Bias, R.G., Mayhew, D.J. (Eds.), Cost Justifying Usability. Academic Press, Boston, MA.

Blattner, M.M., Dannenberg, R.B. (Eds.), 1992. Multimedia Interface Design. ACM Press, New York.

Bødker, S., 1991. Through the Interface: A Human Activity Approach to User Interface Design. Lawrence Erlbaum, Hillsdale, NJ.

Brewster, S.A., Wright, P.C., Edwards, A.D.N., 1993. An evaluation of earcons for use in auditory human–computer interfaces. In: Proceedings of the INTERCHI Conference on Human Factors in Computing Systems. pp. 222–227.

Butler, K.A., 1996. Usability engineering turns 10. Interactions (January), 58–75.

Buxton, W., 1986. There's more to interaction than meets the eye: Some issues in manual input. In: Norman, Draper (Eds.), User Centered System Design Lawrence Erlbaum Associates, Hillsdale, NJ,.

Card, S.K., Moran, T.P., 1980. The keystroke-level model for user performance time with interactive systems. Commun. ACM 23, 396–410.

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

Carroll, J.M., 1984. Minimalist design for active users. In: Proceedings of the Human–Computer Interaction-Interact '84. pp. 39–44.

Carroll, J.M. (Ed.), 1995. Scenario-Based Design: Envisioning Work and Technology in System Development, Wiley, New York.

Carroll, J.M., Kellogg, W.A., Rosson, M.B., 1991. The task-artifact cycle. In: Carroll (Ed.), Designing Interaction: Psychology at the Human–Computer Interface. Cambridge University Press, Cambridge, UK.

Carroll, J.M., Rosson, M.B., 1992. Getting around the task-artifact cycle: How to make claims and design by scenario. ACM TOIS 10, 181–212.

Chin, J.P., Diehl, V.A., Norman, K.L., Development of an instrument measuring user satisfaction of the human–computer interface. In: Proceedings of the CHI Conference on Human Factors in Computing Systems. pp. 213–218.

Christ, R.E., 1975. Review and analysis of color coding research for visual displays. Human Factors 17 (6), 542–570.

Christ, R.E., 1984. Research for evaluating visual display codes: An emphasis on colour coding. In: Easterby, Zwaga (Eds.), Information Design: The Design and Evaluation of Signs and Printed Material. Wiley, New York.

Cleveland, W., McGill, R., 1985. Graphical perception and graphical methods for analyzing scientific data. Science 229, 828–833.

Darken, R.P., Sibert, J.L., 1996a. Navigating large virtual spaces. Int. J. HCI 8 (1), 49–72.

Darken, R.P., Sibert, J.L., 1996b. Wayfinding strategies and behaviors in large virtual environments. In: Proceedings of the CHI Conference on Human Factors in Computing Systems. pp. 142–149.

Diaper, D. (Ed.), 1989. Task Analysis for Human–Computer Interaction. Ellis Horwood, Chichester, UK.

Draper, S.W., Barton, S.B., 1993. Learning by exploration, and affordance bugs. In: Proceedings of the INTERCHI Conference on Human Factors in Computing Systems (Adjunct). pp. 75–76.

Eggleston, R.G., Janson, W.P., Aldrich, K.A., 1996. Virtual reality system effects on size-distance judgments in a virtual environment. In: Proceedings of the IEEE Annual Virtual Reality International Symposium. pp. 139–146.

Ehn, P., 1990. Work Oriented Design of Computer Artifacts, Erlbaum, Hillsdale, NJ.

Foley, J.D., van Dam, A., Feiner, S.K., Hughes, J.F., 1990. Computer Graphics: Principles and Practice, Addison–Wesley, Reading, MA.

Foley, J.D., Wallace, V.L., 1974. The art of natural graphic man–machine Conversation. Proceedings of the IEEE 63 (4), 462–471.

Gaver, W.W., Smith, R.B., 1995. Auditory icons in large-scale collaborative environments. In: Baecker, Grudin, Buxton, Greenberg (Eds.), Readings in Human–Computer Interaction: Toward the Year 2000. Morgan Kaufmann, San Francisco, CA.

Goldberg, D., Goodisman, A., 1995. Stylus user interfaces for manipulating text. In: Baecker, Grudin, Buxton, Greenberg (Eds.), Readings in Human–computer Interaction: Toward the Year 2000. Morgan Kaufmann, San Francisco, CA.

Gould, J.D., Boies, S.J., Lewis, C., 1991. Making usable, useful, productivity-enhancing computer applications. Commun. ACM 34 (1), 74–85.

Gray, W.D., John, B.E., Stuart, R., Lawrence, D., Atwood, M., 1990. GOMS meets the phone company: Analytic modeling applied to real-world problems. In: Proceedings of the INTERACT '90-Third IFIP Conference on Human–Computer Interaction.

Grudin, J., 1994. Groupware and social dynamics: Eight challenges for developers. Communications of the ACM 37 (1), 92–105.

Hammond, N., Gardiner, M.M., Christie, B., Marshall, C., 1987. The role of cognitive psychology in user-interface design. In: Gardiner, Christie (Eds.), Applying Cognitive Psychology to User-Interface Design. Wiley, Chichester.

Hammontree, M., Weiler, P., Nayak, N., 1994. Remote usability testing. Interactions (July), 21–25.

Hannaford, B., Venema, S., 1995. Kinesthetic displays for remote and virtual environments. In: Barfield (Ed.), Virtual Environments and Advanced Interface Design, 3rd ed. Oxford University Press, New York.

Harrison, M., Thimbleby, H. (Eds.), 1990. Formal Methods in Human–Computer Interaction. Cambridge University Press, Cambridge.

Hartson, H.R., Castillo, J.C., Kelso, J., Kamler, J., Neale, W.C., 1996. Remote Evaluation: The Network as an Extension of the Usability Laboratory. In: Proceedings of the CHI Conference on Human Factors in Computing Systems. pp. 228–235.

Hartson, H.R., Gray, P.D., 1992. Temporal aspects of tasks in the user action notation. Human–Computer Interaction 7, 1–45.

Hartson, H.R., Hix, D., 1989. Toward empirically derived methodologies and tools for human–computer interface development. Int. J. Man–Machine Studies 31, 477–494.

Hartson, H.R., Siochi, A.C., Hix, D., 1990. The UAN: A user-oriented representation for direct manipulation interface designs. ACM Trans. Info. Sys. 8 (3), 181–203.

Hinckley, K., Pausch, R., Goble, J.C., Kassell, N.F., 1994. A survey of design issues in spatial input. Proc. User Interface Software Technology (UIST), 213–222.

Hix, D., Hartson, H.R., 1993a. Developing User Interfaces: Ensuring Usability Through Product and Process. Wiley, New York.

Hix, D., Hartson, H.R., 1993b. Formative evaluation: Ensuring usability in user interfaces. In: Bass, Dewan (Eds.), Trends in Software, vol. 1: User Interface Software. Wiley, New York.

Hix, D., Hartson, H.R., 1994. IDEAL: An environment for user-centered development of user interfaces. In: Proceedings of the EWHCI'94: Fourth East–West International Conference on Human–Computer Interaction. pp. 195–211.

Hsu, J., Pizlo, Z., Babbs, C.F., Chelberg, D., Delp, E.J., 1994. Design of studies to test the effectiveness of stereo imaging – truth or dare: Is stereo viewing really better? Proc. Internat. Soc. Optical Engrg., 211–220.

Hutchins, E.L., Hollan, J.D., Norman, D.A., 1986. Direct manipulation interfaces. In: Norman, Draper (Eds.), User Centered System Design. Erlbaum, Hillsdale, NJ.

Jacob, R.J.K., 1993. Eye-movement-based human–computer interaction techniques: Toward non-command interfaces. In: Hartson, Hix (Eds.), Advances in Human–Computer Interaction. Ablex, Norwood, NJ.

Jeffries, R., 1994. Usability problem reports: Helping evaluators communicate effectively with developers. In: Nielsen, Mack (Eds.), Usability Inspection Methods. Wiley, New York.

Karat, C.-M., 1994. A business case approach to usability cost justification. In: Bias, Mayhew (Eds.), Cost Justifying Usability. Academic Press, Boston, MA.

Kieras, D., Polson, P.G., 1985. An approach to the formal analysis of user complexity. Int. J. Man-Machine Studies 22, 365–394.

Kieras, D.E., 1988. Towards a practical GOMS model methodology for user interface design. In: Helander (Ed.), Handbook of Human–Computer Interaction. Elsevier, Amsterdam.

Koenemann-Belliveau, J., Carroll, J.M., Rosson, M.B., Singley, M.K., 1994. Comparative usability evaluation: Critical incidents and critical threads. In: Proceedings of the CHI Conference on Human Factors in Computing Systems. pp. 245–251.

Laurel, B.K., 1986. Interface as mimesis. In: Norman, Draper (Eds.), User Centered System Design. Erlbaum, Hillsdale, NJ.

LeCompte, M.D., Preissle, J., 1993. Ethnography and Qualitative Design in Educational Research. Academic Press, San Diego.

Lewis, C., Polson, P., Wharton, C., Rieman, J., 1990. Testing a walkthrough methodology for theory-based design of walk-up-and-use interfaces. In: Proceedings of the CHI Conference on Human Factors in Computing Systems. pp. 235–242.

MacKenzie, S., 1992. Fitts' law as a research and design tool in human–computer interaction. Human–Computer Interaction 7, 91–139.

Mackinlay, J., 1986. Automating the design of graphical presentations of relational information. ACM Trans. Graph. 5 (2), 110–141.

MacLean, A., Young, R.M., Bellotti, V.M.E., Moran, T.P., 1991. Questions, options, and criteria: Elements of design space analysis. Human–Computer Interaction 6, 201–250.

Macleod, M., Bevan, N., 1993. MUSiC video analysis and context tools for usability measurement. In: Proceedings of the INTERCHI Conference on Human Factors in Computing Systems. p. 55.

Mantei, M.M., Teorey, T.J., 1988. Cost/benefit for incorporating human factors into the software lifecycle. Commun. ACM 31 (4), 428–439.

Mayhew, D.J., 1992. Principles and Guidelines in Software User Interface Design. Prentice-Hall, Englewood Cliffs, NJ.

Mayhew, D.J., Mantei, M., 1994. A basic framework for cost-justifying usability engineering. In: Bias, Mayhew (Eds.), Cost Justifying Usability. Academic Press, Boston, MA.

McCauley, M.E., 1984. Human factors in voice technology. In: Muckler (Ed.), Human Factors Review, Human Factors Society, Santa Monica, CA.

Meister, D., 1985. Behavioral Analysis and Measurement Methods. Wiley, New York.

Moran, T.P., 1981. The command language grammar: A representation for the user interface of interactive computer systems. Int. J. Man–Machine Studies 15, 3–51.

Muller, M.J., 1991. PICTIVE – An exploration in participatory design. In: Proceedings of the CHI Conference on Human Factors in Computing Systems. pp. 225–231.

Mullet, K., Sano, D., 1995. Designing Visual Interfaces. SunSoft Press, Mountain View, CA.

Myers, B.A., 1989. User-interface tools: Introduction and survey. IEEE Software 6 (1), 15–23.

Myers, B.A., 1993. State of the art in user interface software tools. In: Hartson, Hix (Eds.), Advances in Human–Computer Interaction. Ablex, Norwood, NJ.

Myers, B.A., 1995. State of the art in user interface software tools. In: Baecker, Grudin, Buxton, Greenberg (Eds.), Readings in Human–Computer Interaction: Toward the Year 2000. Morgan Kaufmann, San Francisco, CA.

Nielsen, J., 1989. Usability engineering at a discount. In: Salvendy, Smith (Eds.), Designing and Using Human–Computer Interfaces and Knowledge-Based Systems. Elsevier, Amsterdam.

Nielsen, J., 1992. Finding usability problems through heuristic evaluation. In: Proceedings of the CHI Conference on Human Factors in Computing Systems. pp. 373–380.

Nielsen, J., 1993. Usability Engineering. Academic Press, San Diego.

Nielsen, J., Mack R.L. (Eds.), 1994. Usability Inspection Methods. Wiley, New York.

Nielsen, J., Molich, R., 1990. Heuristic evaluation of user interfaces. In: Proceedings of the CHI Conference on Human Factors in Computing Systems. pp. 249–256.

Norman, D.A., 1986. Cognitive Engineering. In: Norman, Draper (Eds.), User Centered System Design. Erlbaum, Hillsdale, NJ.

Norman, D.A., Draper, S.W. (Eds.), 1986. User Centered System Design: New Perspectives on Human–Computer Interaction. Erlbaum, Hillsdale, NJ.

Nowell, L.T., France, R.K., Hix, D., Heath, L.S., Fox, E.A., 1996. Visualizing search results: Some alternatives to query-document similarity. In: Proceedings of the SIGIR'96 ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 67–75.

Payne, S.J., 1991. Display-based action at the user interface. Int. J. Man-Machine Studies 35, 275–289.

Pernice, K., Butler, M.B., 1995. Database support for usability testing. Interactions (January), 27–31.

Shneiderman, B., 1983. Direct manipulation: A step beyond programming languages. IEEE Comput. 16 (8), 57–69.

Shneiderman, B., 1992. Designing the User Interface: Strategies for Effective Human–Computer Interaction. Addison–Wesley, Reading, MA.

Shneiderman, B., 1993. Beyond intelligent machines: Just do it! IEEE Software (January), 100–103.

Shneiderman, B., 1997. Designing information abundant websites: A survey of issues. To appear in International Journal of Human–computer Studies, Special Issue on Human–computer Interaction and the World Wide Web.

Shubin, H., Falck, D., Johansen, A.G., 1996. Exploring color in interface design. Interactions (July/August), 36-48.

Smith, L., Thomas, D., 1964. Color versus shape coding in information displays. J. Applied Psych. 48 (3), 137–146.

Smith, S.L., Mosier, J.N., 1986. Guidelines for Designing User Interface Software, ESD-TR-86-278/MTR 10090, MITRE Corporation, August.

Snow, M., 1996. Charting presence in virtual environments and its effects on performance. Ph.D. Dissertation, Virginia Tech, Blacksburg, VA.

Tufte, E.R., 1983. The Visual Display of Quantitative Data. Graphics Press, Cheshire, CT.

Tufte, E.R., 1990. Envisioning Information. Graphics Press, Cheshire, CT.

Tufte, E.R., 1997. Visual Explanations: Images and Quantities, Evidence and Narrative. Graphics Press, Cheshire, CT.

Virzi, R.A., Sokolov, J.L., Karis, D., 1996. Usability problem identification using both low- and high-fidelity prototypes. In: Proceedings of the CHI Conference on Human Factors in Computing Systems. pp. 236–243.

Weller, H.G., Hartson, H.R., 1992. Metaphors for the nature of human–computer interaction in an empowering environment: Interaction style influences the manner of human accomplishment. Computers in Human Behavior 8 (3), 313–333.

Wharton, C., Bradford, J., Jeffries, R., Franzke, M., 1992. Applying cognitive walkthroughs to more complex user interfaces: Experiences, issues, and recommendations. In: Proceedings of the CHI Conference on Human Factors in Computing Systems. pp. 381–388.

Whiteside, J., Bennett, J., Holtzblatt, K., 1988. Usability engineering: Our experience and evolution. In: Helander (Ed.), Handbook of Human–Computer Interaction. Elsevier, Amsterdam.

Wickens, C.D., Andre, A.D., 1990. Proximity compatibility and information display: Effects of color, space, and objectness on information integration. Human Factors 32 (1), 61–77.

Williges, R.C., Williges, B.H., 1995. Travel alternatives for the mobility impaired: The surrogate electronic traveler (SET). In: Edwards (Ed.), Extra-Ordinary Human–Computer Interaction: Interfaces for Users with Disabilities. Cambridge University Press, New York.

Williges, R.C., Williges, B.H., Elkerton, J., 1987. Software interface design. In: Salvendy (Ed.), Handbook of Human Factors. Wiley, New York.

Wixon, D., Holtzblatt, K., Knox, S., 1990. Contextual design: An emergent view of system design. In: Proceedings of the CHI Conference on Human Factors in Computing Systems. pp. 329–336.

Zhai, S., Milgram, P., 1993. Human performance evaluation of manipulation schemes in virtual environments. In: Proceedings of the IEEE Annual Virtual Reality International Symposium. pp. 155–161.

H. Rex Hartson is Professor of Computer Science at Virginia Tech, Blacksburg, VA, where he was founder of the Human–Computer Interaction Research Group in 1979 and is a fellow of the Center for Human–Computer Interaction. His degrees are from the University of Michigan and Ohio State University. He has worked in industry as an engineer and researcher for the Xerox Corporation. He currently does research and development in human–computer interaction, user interface design and evaluation, the relationship between human–computer interaction and software engineering, interface development methodologies, design representation techniques, and usability methods. Dr. Hartson was series editor for *Advances in Human–Computer Interaction* published by Ablex; has served on various panels, conference committees, and workshops; is co-author of *Developing User Interfaces: Ensuring Usability Through Product & Process*, (Wiley, 1993); and has published numerous papers in human–computer interaction. Since 1964, Prof. Hartson has consulted for many organizations in business, industry, and government.