HUMAN–COMPUTER INTERACTION

INTRODUCTION

This article begins with a summary of the historical roots of human–computer interaction (HCI), followed by a summary of issues related to use and context. The next section explores human characteristics, and the following one describes usability principles that help explain why a particular design is easy to learn and operate, while another seems complicated and confusing. Next, there is an overview of principles of design as well as the design evaluation process. The article then investigates several prominent human–computer interaction paradigms, which range from the traditional to the more esoteric. The article summarizes several related domains in which HCI issues play a significant role. The next part surveys important issues related to the role of HCI in risks of computer technology and safety-critical systems. The last section concludes the article with an analysis of the major challenges ahead for this discipline.

HISTORY

The origins of HCI as a discipline trace back to human factors issues during World War II, in particular related to human attention in radar monitoring and interface design for aircraft control. Ergonomics, which is the engineering of work systems to improve the fit between humans and jobs, and thereby increase performance, emerged from this context. With digital computers making their entry into business applications in the 1950s, these systems began to transform the way people worked. However, the technology did not seem to integrate with existing work practices. This observation led Licklider (1) to propose man–computer symbiosis (1) to augment human intellect with the memory and computational power of computer hardware, which free the human from mundane tasks. The idea of the *man–machine interface* was born.

Through the 1960s, computer graphics, the mouse, and the ARPANET, which is the first wide-scale computer network, were invented, and by the 1970s, personal computers (PCs) began to appear on individual desktops. With the development of the graphical user interface (GUI) paradigm, several benefits resulted. The style of interaction changed from a command-line language to one of menu selections, which in turn, provided visibility of possible actions, often more tightly constrained, and permitted rapid feedback from the interface. By this point, the term *human–computer interaction* was more appropriate to describe what had become an area of research more focused on the cognitive aspects of interaction with computers than on the physical work system of earlier generations.

Computer adoption surged through the late 1980s and 1990s, with desktop computers becoming commonplace both in the home and industry. Although standards and guidelines for interface design were available, software was often developed in an ad hoc manner. Despite some notable exceptions (2,3) during this period, usability testing was generally left until the end of the development process. This disregard for HCI led to unnecessary frustration as many users resigned themselves to the belief that "computers are too complicated for me." As developers began to realize that design problems were costly to correct late in the product lifecycle, these concerns prompted a focus on the user rather than only on functionality.

Today, HCI is relevant to almost every use of computers, from desktop to hand-held PCs as well as embedded systems found in consumer electronics, home appliances, and industrial devices. It offers techniques for improving existing interfaces, evaluation methods to determine whether new designs and interaction paradigms improve human performance yet are easy to learn, and helps minimize risks of failure.

USE AND CONTEXT

The importance of HCI as a discipline has much to do with the use and context of computers in society. This discipline involves the nature of human work, social organizations, and the impact of computers on them, for example, in terms of productivity, job satisfaction, or quality of life (4). The applications in which computers are found span a wide variety of domains, which include text and document processing, human–human communication, information display, design activities, and control systems. In each such area, different considerations apply to how humans and machines interact. HCI is therefore concerned with characterizing and understanding the relevant properties of each to offer appropriate design guidelines tailored to particular applications. HCI also considers how well a specific design provides a fit between human and machine, which party must adapt to the other, and at what point such adaptation occurs. These questions have significant implications to issues that range from systems reliability to user perception of new products.

Wiley Encyclopedia of Computer Science and Engineering, edited by Benjamin Wah.

HUMAN CHARACTERISTICS

"Know thy user" (5), which is the imperative to design for the needs and abilities of ones intended community, is widely considered as the first rule of usability. The interface requirements of domain experts differ considerably from those of office software users. Thus, terse symbolic output may be appropriate for a skilled power plant operator who instantly recognizes its meaning, whereas a control pedestal teeming with buttons, knobs, dials, levers, and multiple reconfigurable graphical displays is perfectly acceptable for a commercial aircraft pilot who needs instant access to these functions. Furthermore, early adopters of technology often desire functionality above all else and are not daunted by a complex interface, but as a technology matures, respect for ease-of-use becomes increasingly significant.

Familiarity with various concepts, experience in the everyday world, and cultural bias all play a determining role in how users relate to a particular interface. The computer mouse, for example, initially confused most users before they had observed its operation in action, whereas deleting a file by dragging it to the trash can (or recycle bin) icon might seem unusual to a culture that simply burns all its refuse. Even the significance of various colors is culturally influenced; whereas red indicates a warning to most westerners, it is a sign of good luck in China. As a result, terminology such as *user-friendly, natural*, or *intuitive* must be applied appropriately to the context (cultural, social, and work domains) to be meaningful. The ease-of-use of a particular interface is inherently related to its *familiarity*, but it does not necessarily imply superiority (6). With respect to learning time, user productivity, and error reduction, an improved interface may be dramatically unfamiliar.

Models and Theories

HCI builds on the knowledge of cognitive psychology in understanding the capabilities and limitations of human perceptual and information processing, problem solving, learning, as well as active coordination of motor skills. Considerable efforts in this area relate to modeling the operation of the human mind as an information processor. The best known example is the Model Human Processor (7) abstraction, which consists of perceptual, motor, and cognitive subsystems. Based on this abstraction, Card, Moran, and Newel introduced the GOMS model of user behavior, which allow for predictions of performance time for specified tasks. GOMS consists of sets of *goals, operators, methods*, which are sequences of operators that accomplish goals, and *selection rules* for choosing between competing methods. Although the technique is complex and suffers numerous shortcomings, it nevertheless offers a useful approach to evaluating or comparing interfaces.

With respect to motor skills, Fitts' Law (8) describes the index of difficulty in moving from a start position to a target as $b \log_2(\frac{D}{W}+1)$ where D is the distance to the target and W its width. This law is widely recognized for its reliability in predicting the performance of various graphical user interface schemes. Fitts' Law correctly predicts that the corners and edges of the screen are the most easily reached targets. Although originally formulated only for one-dimensional movement, it has been generalized to two (9) and higher dimensions (10) in addition to general trajectories (11). A similar formulation, Hick's Law (12), which is otherwise known as the Hick-Hyman law, holds that the average reaction time for a user to decide between n equally probable choices is $b \log_2(n+1)$ for some constant b. One of the best-known examples from cognitive psychology, with implications to the design of speech interfaces, is that the maximum number of chunks of information humans can store in short-term memory is 7 ± 2 (13).[1]

Although GOMS and Fitts' Law relate to low-level human–computer interaction tasks or ways of modeling human goals into low-level actions, various theories also address higher level elements of human behavior and interaction with computers. For example, Information Foraging Theory (15) deals with information seeking, gathering, and consumption as a behavior optimization problem, which is based on similar principles that govern foraging behavior in animals. The theory introduces the concepts of *information scent*, or perception of cues that relate to the expectation of success (finding the desired information) along a path and *information diet*, or the choice of which clusters of information are considered worthy of pursuing (16). It suggests that users will continue in a particular path if they perceive themselves to be making sufficiently fast progress toward the goal, that is, if the scent is growing stronger, relative to the effort expended, and the perceived value of other search opportunities. This concept is particularly relevant for web developers, as it helps explain why users will visit one website or leave it for another. Similarly, it motivates the use of link labels that offer explicit descriptions of the destination contents and navigation cues that indicate the user's current location in the site.

Shneiderman's Syntactic/Semantic Model (17) differentiates the way in which novice users think of an interface in terms of syntactic commands, whereas experts form higher level semantic representations that are less system dependent. Syntactic knowledge, such as the specific keystrokes needed to perform a certain operation, requires rote memorization. Semantic knowledge, such as the mechanism for moving a sentence between two paragraphs, is built hierarchically from lower level concepts. Such knowledge tends to be system independent and is

[1]Note, however, that Miller's results have often been misinterpreted, which led some to believe that the same rule limits our ability to process more than 7 ± 2 items of information displayed simultaneously on a screen. Larson and Czerwinski (14) showed that the number of decisions a subject had to make in an information navigation process was actually far more significant a factor in determining performance than the number of options presented at each decision point.

thus often transferable across computer systems. This observation explains the preference for direct manipulation over command line interfaces, in particular for novices, as users interact directly with objects in the interface, and thus, the actions are available explicitly in the high-level domain.

Norman's Seven Stages of Action (18) breaks the structure of an action into stages of execution and evaluation. First, a goal is selected. In the execution stages, an intention is formed to carry out the actions necessary to achieve the goal. The intention is translated to an action sequence, which is then executed. In the evaluation stages, the world is perceived, the perception is interpreted, and finally, it is compared against the original goal. The *Gulf of Execution* refers to the difference between intentions and available actions. Similarly, the *Gulf of Evaluation* describes the effort to interpret the state of the system and determine whether the original intentions have been satisfied.

Accessibility

Human diversity represents a significant challenge for interface design. *Accessibility* of interfaces to disabled and elderly populations is motivated both by economic arguments and legal requirements. Although design for accessibility requires careful attention, it offers benefits to the broader community as well, as in the example of the telephone, which resulted from development of communications technologies for the deaf (19). Users of varying levels of skill have significantly different needs and expectations regarding how the interface should behave. Recognizing this diversity of the user population, computer operating systems often support multiple mechanisms for accessing common functions, such as menu options and keyboard shortcuts, as well as screen magnification, screen readers, and speech control for disabled users.

Ergonomics and Human Factors

Ergonomics and human factors gained prominence during World War II with a series of experimental studies related to aviation. A notable example was the study of aircraft that crashed during landing because of the difficulty pilots had in distinguishing between the side-by-side controls for flaps and wheels (20). A more modern example is the relationship between brightness settings and luminance contrast on the usability of an air-traffic control system.

Considerations of human physiological characteristics continue to play an important role. Awareness of physical, cognitive, and sensory limits, as well as the environmental conditions in which a computer system will be used, all govern aspects of interface design. Certain activities, such as repetitive keyboarding, can be the cause of injury. Information overload and inattention to the role of information density (21,22) on search times and user preferences cause systems to become unusable and increase risks of accidents.

With regard to the human sensory system, visual perception, based on color, brightness, and size, informs the design of information displays while auditory perception permits the use of sound cues for alarms, error signals, and confirmation indicators. Haptic perception, referring to the interplay between hand movements and derived sensations, provides feedback when button are touched and pressed or dials are turned.

USABILITY PRINCIPLES

Many principles of HCI have their roots in human factors or ergonomics, the study of human interaction with products, environment and equipment in performing tasks and activities. In the everyday world, we interact with tens of thousands of objects, yet generally manage to use them properly the first time they are encountered. Norman (18) attributes this success to a combination of adequate *visibility, feedback, constraints, mapping, and affordances* that underlie their operation.

Visibility refers to a control or display of system state being readily apparent to the user, rather than hidden or obscured by clutter. When these are visible, users can better determine what actions can and should be performed at any time. A closely related concept, *feedback*, is the timely communication of information back to the user regarding actions that have been performed and the current state of the system.

Constraints, whether physical, logical, or cultural, help guide user behavior, which prevent incorrect actions while making correct actions easily deduced (18,23). *Forcing functions*, which are a special class of physical constraints, force conscious attention on a particular action or prevent the user from performing the action until another is done. For example, a car door lock that can only be operated from the outsides prevents the driver from leaving the keys inside the car.

Mappings describe the relationships between controls and their actions, or actions and their effects. Natural mappings take advantage of physical analogies, such as a controller for a car seat that is made in the same shape and causes corresponding movements in the seat when it is manipulated. For example, lifting the front part of the button raises the front edge of the car seat. Poor mappings, in contrast, are difficult to learn and remember, which impacts on effective use of the system.

Although many accept Norman's revised definition of *affordances* as "the perceived and actual properties of the thing, primarily those fundamental properties that determine just how the thing could properly be used." Norman (18) differs from the original definition of psychologist J. J. Gibson, as existing "*independently of perception* and only as a relationship between an organism and an object" (24). Regardless, interface designers must be sensitive to both of these perspectives of affordances, asking whether the functionality of a particular object is appropriate for the intended user, and whether the user perceives that some action is possible with that object (25).

A *conceptual model* describes how the system should look and act. This model may be based on activities such

as instruction, conversing, manipulating and navigating, or exploring and browsing (26). Each of these activities suggests a different style of interaction, which should be based on addressing the users' needs and tasks. Conceptual models may also be based on objects, for example, the computerized spreadsheet (27), which is based on the familiar ledger sheet. Ideally, the resulting design provides sufficient information so that users can, with a reasonable amount of experience, form their own conceptual models that correctly predict how the system will respond to input. A correct perceptual model permits effective use of the system, whereas a faulty model can lead to incorrect actions, which is a potentially dangerous situation, in particular for safety-critical systems.

Interface metaphors are often used to describe conceptual models. These metaphors make the system resemble something familiar to simplify the learning process, such as the appearance of conventional control button symbols *(play, pause, fast-forward, record)* from consumer electronics devices, employed in a software media player. A frequently cited example is the *desktop* metaphor, with files, folders, a mailbox, and trashcan. Metaphors are successful because they take advantage of the familiarity of most users with their real-world equivalents. However, numerous criticisms have been raised against interface metaphors for violating rules of the real world, constraining the interface, limiting designer imagination, and conflicting with design principles (28). An often-cited example of such a violation is the Macintosh interface, in which dragging a file to the trash deletes it, but dragging the icon of a compact disk or other media ejects it.

A design problem frequently encountered in practice is the assignment of multiple functions to a single control or button. For example, the same button on a digital watch may be used to advance the date, hour, or minute, start and stop the chronometer, or cancel the alarm, depending on context. The affordances are clear—each button can be pushed—but the mappings are not, as there is generally no visible relation between each button and its associated function(s). At the opposite extreme, users of some video cassette recorder and digital video disk remote controls may be confronted by dozens of small, nearly identical buttons, many of which correspond to functions that will rarely be used. In this example, to provide the additional functionality, visibility has been sacrificed, which results in difficulty locating the appropriate button when it is needed. In both cases, it may be difficult for users to learn the correct mappings or gain a conceptual model of the system, which thereby creates usability challenges.

The wealth of human interaction experience with objects in the everyday world offers numerous examples on which the principles above are based. Where they have been followed correctly, we have little trouble making use of the objects effectively. Similarly, applying these principles to the design of human–computer interfaces improves both our understanding of the system and in turn, its ease-of-use.

DESIGN AND EVALUATION

The design and evaluation of human–computer interfaces has been the subject of considerable research and accumulated wisdom. The principles, guidelines, design approaches, and evaluation methods described here resulted from systematic efforts to improve the interface based on an understanding of users and their tasks. These efforts represent alternatives to simple intuition and offer a valuable starting point in the design and evaluation of a successful system.

Design Principles and Guidelines

A wealth of guidelines or best practices exist for the design of human–computer interfaces, of which only a subset are relevant to any particular problem. Smith and Mosier provide a comprehensive survey of such guidelines, divided into sections dealing with data entry, data display, sequence control, user guidance, data transmission, and data protection (29). Many guidelines seek to satisfy objectives that are relevant across a variety of tasks. Important examples include *consistency*, whether that of the data display or sequences of actions required to perform certain activities, *minimizing the number of necessary actions* needed to accomplish a task, *minimizing memory load* imposed on users, and *support for customization*.

More broadly defined principles of design have been proposed, including the following usability heuristics of Nielsen and Molich[2](30):

- Visibility of system status: Keep the user informed through appropriate feedback given within reasonable time.
- Match between system and the real world: Employ words, phrases and concepts familiar to the user.
- User control and freedom: Provide a clearly marked "emergency exit" and support undo/redo operations.
- Consistency and standards: Avoid confusion of different words or actions to mean the same thing.
- Error prevention: Even better than good error messages, where possible, prevent errors from occurring.
- Recognition rather than recall: Minimize user's short-term memory load by making objects, actions, and options visible.
- Flexibility and efficiency of use: Provide shortcuts or accelerators that are unseen by the novice user.
- Aesthetic and minimalist design: Simple and natural dialog, which avoids irrelevant or rarely needed information.
- Help users recognize, diagnose, and recover from errors: Error messages should precisely indicate the problem and constructively suggest a solution.

[2]Although originally defined using slightly different terminology (31), we provide the list as refined by Nielsen (32), in which the tenth heuristic of help and documentation was added.

- Help and documentation: Although it is preferable if the system can be used without it, such documen tation may be necessary.

These heuristics have since been extended and specialized for domains, such as website evaluation, and have also spawned a number of related inspection methods. Other such lists include Norman's principles of design (32), Tognazinni's first principles of interaction design (33), and Shneiderman's *Eight Golden Rules* (34).

Additional design principles relate to tailoring the interface as appropriate to the users' skill level (5), the tasks to be performed (35,36), integrating automation while preserving human control, and selecting an appropriate interaction style, whether direct manipulation, menu selection, form fill-in, command language, or natural language (34).

User-Centered and Participatory Design

As human–computer interfaces have evolved from punchcards and line printers to today's multimodal systems that employ a wide variety of input devices, an accompanying need has emerged to concentrate on the user rather than the system at the core of the design effort. This concept is best embodied by the philosophy of *user-centered design*, which is predicated on the realization that *usability*, or ease of use, is distinct from the *utility* or *functionality* of a system.

User-centered design emphasizes an early focus on users, which typically involves observations of users at work as an initial step. This step is followed by early user testing on simulations or simple prototypes, and it continues for many iterations as the interface evolves through testing and in response to user feedback (37).

Participatory design takes this approach one step further, on the basis that it is better to design *with* the user than *for* the user. As such, the scope and duration of user involvement in the design process as well as their control over the design decisions is greatly expanded. However, some argue that users often lack the necessary design background and understanding of the technical aspects of development to contribute practically to this process.

Regardless, as a recommended step early in the process, one should develop *user scenarios* that describe a typical set of activities that will be performed by users. These scenarios are then refined through *task analysis*, in which the specifics of how each task is performed are elaborated, without reference to interface details. This provides an organizational framework for initial development of a prototype or mock-up of the system, on which user testing can then be conducted.

Evaluation

Testing at the early stages of product design is intended to focus on high-level conceptual problems rather than detailed interface issues. It thus makes use of storyboards or low-fidelity prototypes rather than fully operational systems. Later stages of testing may employ software mock-ups or simulations of the actual system, typically with limited functionality. An important rule is that testing should involve the subject performing typical tasks and not being shown what to do. Product demonstrations are valuable for marketing purposes but they do not elicit useful feedback regarding flaws in the design.

In *heuristic evaluation*, as proposed by Molich and Nielsen (30,31), several independent evaluators seek to discover usability problems with the system by asking whether it satisfies the heuristics described in Section 6.1. Some critics have pointed out that many of the heuristics used in evaluation have never been validated. For this reason, other criteria by which a product can be assessed (38), developed as a set of international standards, have been suggested as an alternative. Using such guidelines, a *usability expert review* can be conducted by a single evaluator.

The *cognitive walkthrough* (39) is intended to determine whether the system is easy to learn. This testing methodology has the developers or test group consider, for each task, what are the goals and knowledge of the user, and what sequence of actions need to be performed to accomplish these goals. The walkthrough also considers whether any discrepancies exist between the interface and the user's likely expectations.

Usability testing typically involves quantitative experiments in which users are observed performing representative tasks with the system. The time for task completion as well as the type and number of mistakes provide concrete measures of performance against some predetermined target and help isolate problematic aspects of the design. Because such measures can only be gathered from a relatively functional system, usability testing is appropriate only for later stages of system development.

It is unfortunately all too common, even among many leaders of the information technology industry, to ignore design principles and issues of usability, instead developing products with no user testing until the very last minute, under the misguided assumption that following HCI guidelines is too costly and time consuming. However, analyses have demonstrated that prototyping, early user testing, and iterative design, are more likely to reduce both cost and development time by avoiding costly redesigns late in the development cycle (40). Correcting problems at the design stage is 10 times less expensive than doing so during development, and possibly 100 times less expensive than after product release (41). Significantly, from the user perspective, the result is generally a superior product. Some companies have learned this lesson through painful experience, after investing millions of dollars developing systems that are essentially unusable before recognizing the critical role of HCI.

INTERACTION PARADIGMS

Computer interfaces have evolved from batch mode processing of punch cards, through text-based timesharing consoles, and command line control, to the presently dominant GUI. Although the description of the original Macintosh computer was one that was constrained by "highly impoverished communication channels between the user and

computer" (42), numerous advances of input and output technologies have occurred in recent years, significantly enriching the range of available interaction capabilities. Keyboard-and-mouse interfaces, which are often criticized for restricting expressivity and for the level of distraction they impose on users, nevertheless play an important role in many present-day interactive computing applications. However, as the traditional desktop computer is being challenged by other forms of interactive computing, which include table top and wall displays, ultra-mobile devices, and background information displays, new interfaces and interaction paradigms are evolving that may prove more effective in specific domains.

The remainder of this section summarizes well-established interface technologies before proceeding to cover emerging technologies.

GUIs and the Desktop Model

The GUI developed from the pioneering work of Ivan Sutherland (Sketchpad (43), a pen-based drawing system), Douglas Engelbart (NLS (44), which featured the popular computer mouse, hypertext, and shared-screen collaboration), and Alan Kay (Flex Machine and Dynabook (45)). It was then developed into a working system by the Xerox Alto (46) in 1973 and popularized by the Apple Macintosh.

The graphical desktop metaphor at the core of such user interfaces has been characterized by its constituent elements of windows, icons, menus, and pointers (WIMP). The interaction style is typified by direct manipulation (47) in which the user manipulates interface elements through gestures that correspond approximately to those that would be applied to objects in the physical world. Documents may be placed on the desktop, stored in folders, possibly arranged hierarchically, and emptied into a trash can.

Although the model does not accurately reflect the manner in which users organize and manipulate objects on a physical desktop, a more significant criticism is related to its bias toward business applications, which were the dominant use of computer systems at the time the paradigm was developed (48). However, the multimedia and network applications run by computer users today typically bear little resemblance to the business tools of the 1970s and early 1980s. As such, the learned skills associated with the desktop metaphor may be an impediment to efficient use of the computer.

Furthermore, the need to associate documents with files names (49) and refer to documents exclusively through these handles imposes an unnecessary cognitive burden on the user, who often maintains thousands of such files on a single machine. This problem has stimulated several efforts to explore three-dimensional interfaces, although none, as yet, have achieved commercial success. Perhaps more promising are the advances being made to facilitate content retrieval without filename specification by offering powerful keyword search capabilities on databases that include file contents as well as meta-data.[3]

[3]Current examples include Quicksilver and Spotlight (OS X), tracker and beagle (Linux), Windows Search, and Google Desktop (Windows).

Post-WIMP interfaces, which are described in the following sections, employ such features as speech, image, and language understanding, possibly driven by knowledge bases.

Speech and Non-Speech Audio

The day in which we interact with computers as effortlessly as we do with other human beings likely remains far off. However, considerable advances have been made in automatic speech recognition (ASR) and speech synthesis (text-to-speech), which thereby provides an alternative form of human–computer interaction to the keyboard, mouse, and screen.

ASR may take the form of grammar-constrained recognition, which is based on a defined grammar specification, or the less restricted, and hence, more challenging, natural language recognition, which employs statistical models of expected responses to specific questions. Because the most effective ASR results are obtained with modest, constrained grammars, systems that employ speech interaction typically encourage users to speak short phrases. General purpose dictation systems simply transcribe speech without the requirement for semantic understanding. These are now claimed to achieve recognition rates of up to 99 percent under optimal conditions with training, and outperforming typical human typists on text entry speeds.[4] Computer speech synthesis has matured to the point where it incorporates rules of prosody and is often indistinguishable from true human speech[5] and can convey basic emotions effectively.

Examples of non-speech audio include alarms, warning sounds, auditory icons, and *earcons*, which use structured audio to represent information about a particular object. *Auditory displays* employ sound to display data or otherwise enhance the user interface, either as a replacement or supplement to a purely visual display. In general, the use of audio output is appropriate when information to be conveyed is short, simple, and temporally significant, or when the visual channel is already occupied by other tasks.

Although audio interfaces remain less efficient for most users than visual display and keyboard input, these interfaces are useful as an interaction paradigm where display size and input space shrink or become nonexistent. This approach is relevant not only for devices such as the telephone, but also for a wide range of applications in which the user cannot be expected to attend constantly to the display. However, it should be noted that speaking and listening use the same part of the brain that supports problem solving and short-term memory, which increases the difficulty of performing cognitive processing while speaking (50).

Haptics

Haptic devices take advantage of the bidirectional nature of the haptic channel to facilitate communication between

[4]See http://www.redorbit.com/news/technology/348531/voice_recognition_technology_has_it_come_of_age/. (Archived at http://www.webcitation.org/5bhx4fywf).

[5]See http://emosamples.syntheticspeech.de for an extensive collection of examples.

humans and machines (51). This dual function can be appreciated in various types of controls, such as the purposefully heavy radio tuning dial compared with the click-to-frame feature of a video editing jog dial. The haptic system provides users with a perception of system state and guides movement as sensing and action occur simultaneously.

Haptic interfaces typically accept input through manipulation of an articulated and actuated mechanical controller held by the user or by the deformation of a data glove. With appropriate control, this interface paradigm results in a perception of interacting with virtual objects. The intention is to produce specific sensations appropriate to a task, either *symbolic*, as in the warning an aircraft pilot of an impending stall flight condition, or *iconic*, as in the representation of the visco-elastic properties of objects for use in surgery simulation. In a similar manner, a *haptic display* can be associated to graphical information as an alternative visualization mechanism. Haptic interfaces can also be useful for visually impaired users or to improve awareness where the visual channel is inadequate. Finally, the dynamics of vibration and impact resembling that of actual machines may be synthesized. This approach is relevant to computer game controllers (rumble), force feedback joysticks, steering wheels, and virtual reality applications.

Multimodal Interfaces

Multimodal interaction is often used as a synonym for the addition of audio to the interface, but in reality, it refers to the use of more than a single sensory mode (sight, sound, touch, smell, or taste) of communication at a time. Speech is typically combined with pen input, manual gestures, lip movement, or gaze tracking; the earliest example is Bolt's "Put that there" system (52). Considerable use of haptics as an output modality has been coupled with graphics or audio, the latter, in particular for the visually impaired. Although some results have been achieved for interfaces employing smell and taste, these methods are not considered practical.

Multimodal interfaces may provide benefits that include improved ease-of-use, decreased error rates, and reduced burden on the visual channel. The complexity of processing multiple input streams in parallel raises a challenge, in particular determining when and which streams should be combined, and how to interpret the resulting fusion, given that multimodal information is usually not simultaneous (53). However, some benefit is obtained from cross-channel constraints to counter this extra processing burden, as complementary, rather than redundant, information tends to be available from the different modalities. This result is supported by cognitive studies, which indicate that people naturally partition information across their communication channels.

Virtual Reality

Virtual and augmented reality represent different ends of the spectrum of *mixed reality* systems (54). The former synthesizes an alternative, completely computer-generated environment with which the user interacts, whereas *augmented reality* requires interaction with the physical world. Augmented reality is discussed in the next section.

Virtual reality (VR), which is a term coined by Lanier, initially implied immersive environments, such as the CAVE Automatic Virtual Environment (55), in which the user is sensorially isolated from the physical world. However, the term may be used to describe nonimmersive (desktop) simulated environments as well. Visual information typically dominates VR, with data observed either on a large computer screen or through a head-mounted display, usually exploiting stereoscopy to create the impression of a three-dimensional environment. Auditory output is often employed, and haptic interaction to a lesser degree. Manual input may be provided through a data glove, although various keyboard and joystick devices are also popular. For immersive environments, the user's head position and orientation, which determine the view perspective, are often obtained by tracking devices.

Despite the increasing power of graphics engines and processors, most VR environments exhibit a lack of realism that requires a certain suspension of disbelief to be effective. Regardless of this limitation, VR continues to find application in entertainment and simulation, such as games and flight training systems. However, notable success of the technology has also been achieved in therapeutic use, in particular, for helping subjects overcome various phobias.

Augmented Reality

In contrast with virtual reality, augmented reality systems involve interaction with the physical world through a computer-mediated layer to enhance the user's experience. Real world objects are typically overlaid with projected graphics or a portable, hand-held device is employed to display additional information related to the user's current location or to an object in front of the user.

The *DigitalDesk* (56), is an interesting example of the former approach that seamlessly combined the physical world of a desktop and paper documents with the benefits of computer processing. Users could place a paper document on the desk and use their fingers or a stylus to select regions of the page, whose contents were captured by an overhead video camera and automatically imported. The graphical content could then be manipulated by drawing software functionality such as cut-and-paste while text content could be recognized through optical character recognition and inserted into a spreadsheet or calculator program. The final result could then be printed to return to the physical world.

Handheld examples of augmented reality include the *Chameleon* (57), which featured a position-sensitive liquid crystal display (LCD) that explained the operation of an audiovisual equipment stack as it was moved between the various rack-mounted devices and the *Navicam* (58), which overlaid a live video representation of the scene with context-specific information, such as directions to an individual's office or historical background on a painting. Localization information for augmented reality systems is

often obtained from Global Positioning System technology for outdoor applications and visible tags, observed by a video camera or radio frequency identification tags for indoors.

Another approach builds on the concept of auditory displays, which augment the environment with sound, delivered through portable wireless headphones. This method offers the advantage of providing personalized, context-sensitive information, without requiring an explicit query, for example, interaction with a PDA, nor active attention to a video display (59). For example, a greeting could be played when passing by the office of a colleague who is presently away, or the audio content might provide some indication of the colleague's activity since the last time the user visited her office.

Tangible Interfaces

A closely related paradigm to augmented reality, tangible or graspable user interfaces consist of a set of dedicated physical artifacts, which are computationally coupled to the representation and control of digital information. This paradigm allows the designer to exploit shape, size, and position of controls to increase functionality without a corresponding increase in complexity. Additional benefits of this approach include the persistence of a particular physical object being mapped to a function and the ability to employ spatial reasoning skills in manipulating these physical projects (60).

For example, Bishop's *marble answering machine* uses marbles as a physical representation of voice messages. Incoming messages cause the machine to produce a marble, which can be placed into an indentation to play the corresponding message. Similarly, placing the marble onto an augmented telephone dials the caller (61). Another example of interaction through tangible interfaces is the manipulation of physical models or *phicons* of buildings on a graphical display surface to control the display of a map (62).

A noteworthy extension of tangible interfaces was the design of computationally augmented *DataTiles* (63) as a modular platform for mixed physical and graphical interactions. The transparent tiles can be stacked side-by-side on a flat display surface to create simple queries on demand and display their output graphically. Various tiles provide sliders and dials for the supply of input parameters while others offer application context. Placing the *time dial* tile against the *photo album* tile allows the user to scroll backward and forward through the album, while adding a *paint* tile provides a mechanism for adding illustration to the currently selected photograph.

Ubiquitous Computing

Weiser proposed Ubiquitous Computing (Ubicomp) (64) in response to what he saw as a disappointing trend of human–computer interface design toward systems that dominate the user's attention. Ubicomp, also known as *pervasive computing* or *calm technology*, is an alternative in which technology recedes to the background, rather than remaining the focus of activity. In this manner, the computer effectively becomes invisible such that "we use it without even thinking about it."

Early efforts to create the infrastructure for this paradigm led to the development of location-aware tabs, pads, and boards, which served, respectively as prototypes for personal digital assistants (PDA), tablet computing, and digital whiteboards. The intent was that these devices would be used as computing-enhanced post-it notes, sheets of paper, and whiteboards, flip charts, and bulletin boards. However, unlike the single, general-purpose personal computer, Weiser suggested that the full potential of Ubicomp would be realized through the interaction between such devices, which respond to changes in their environment and act automatically based on user needs and preferences.

Norman, who promotes a similar ideal, argues that the computer, as infrastructure, should be invisible. In contrast to miniaturization, in which the computer is simply reduced to the size of a PDA, such invisibility should be achieved through information appliances. These appliances are specialized in function, just as tools, but they are networked together to retain the benefits of the PC in terms of information sharing between applications (32).

The philosophy of calm technology mandates that the user is peripherally aware of system state (background awareness), just as one is aware of other cars while driving. However, unusual activity engages direct attention (foreground awareness) so that the user may attend to it. This concept is demonstrated by the *dangling string*, a long piece of plastic spaghetti, which is suspended from a ceiling-mounted motor and expresses Ethernet traffic as a function of its rotation speed (65). Whereas a quiet network causes only occasional motor movement, a suddenly busy network causes the whirling string to make an obvious noise.

Calm technology avoids cognitive overload by allowing for seamless transitions back and forth between background and foreground awareness. Buxton recognized the importance of such transitions in describing how a video-conference user could glance at a grid of periodically updated low-resolution snapshots of colleagues to check whether someone was available for a meeting, and if so, initiate a video-conference session with that individual by clicking on their snapshot (66).

Wearable Computing

Unlike ubiquitous computing, in which the computing infrastructure is distributed and embedded within the environment, wearable computing places the entire infrastructure directly on the user. Beyond the everyday applications of wearable computing in cellular telephones and bluetooth-equipped headsets, digital wristwatches, MP3 players, wearables are in use by numerous workers, which include telephone repair technicians, power plant and security personnel, hotel staff, and broadcast journalists.

Early wearable computing tended to be bulky and awkward, and required cumbersome head-mounted displays that isolated the user's senses from the physical world. However, these systems are increasingly attractive as the underlying components become lightweight, power-efficient, durable, and largely unnoticeable. Thumbnail-sized computers, thin, flexible keypads, wireless earbud speakers, and eyeglasses with miniature active displays combine to offer a practical infrastructure in which computing services can be made available to the user any place, anytime.

Output may be provided as audio through earbuds, or graphically via an LCD display or scanning retinal projection technology. For the latter, an intensity-modulated laser diode beam is scanned in both *x*- and y-directions to *draw* each pixel on the retinal surface of the viewer. To keep one hand free for other tasks, text entry is typically performed through a small keypad, which is possibly strapped to the user's arm, or a chord keyboard, which affords greater entry speed, at the expense of a steep learning curve. Speech recognition systems are also popular, although subject to environmental constraints such as background noise.

Another direction for wearables is in fashion. Thermochromatic fibers, when woven into the fabric, can be controlled to generate arbitrary patterns on one's clothing; electronic ink can be incorporated into wearable animated displays or jewelery; and other light emitting materials can be employed to produce aesthetic results often in response to movement or physiological state of the wearer.

Additional applications envisioned, based on active manipulation of the wearer's view (67), include a visual memory prosthesis and an active visual filter. The former searches a face database and overlays pertinent data about the individual in front of the wearer, such as name, birthday, and location last seen, whereas the latter might replace highway advertisements with extensions of the scenery.

Mobile Computing

The popularization of mobile computing, including PDAs, ultra-mobile personal computers (UMPCs), and feature-rich cell phones has motivated a large body of work on interface design for systems with limited screen real estate and input capability, and addressing the limited attention users can devote. The Palm Pilot design strategy was predicated on reducing the feature set to an absolute minimum, putting frequently used features on-screen but hiding all others behind drop-down menus, in a manner analogous to our organization of an office, with frequently used items on the desktop and others in drawers or cabinets (68). However, a careful balance had to be struck with the number of taps required to accomplish any particular task.[6]

[6]David Pogue refers to the "tap-counter" at Palm Computing, who ensured that a maximum of "three taps" on the touch screen was necessary to accomplish any particular task. Available: http://www.nytimes.com/2006/01/05/technology/circuits/05pogue.html. (Archived at http://www.webcitation.org/5bhxUnnEg).

Small screen constraints have led to zooming interfaces and a greater use of audio, in particular for output. Text-entry systems including Unistrokes, Graffiti, and ShapeWriter for stylus devices, predictive text entry (T9) for telephone keypads, and a variety of pointing stick (trackpoint) devices have also been developed specifically for this domain (69).

Affective Computing and Biofeedback

Although humans employ facial movements, expressions, gestures, and varying tone of voice in their interactions with each other, information technology is generally oblivious to the user's emotional or *affective* state. Affective computing (70) seeks to remedy this shortcoming by employing recognizers for facial movements, head gestures, and body pose to influence the response of the computer, such as providing additional information when it detects stress or frustration, which are likely indicators that the user is encountering difficulty with the system. Research in this field tackles the problem of determining whether a user's emotional state has been affected as intended through interaction as well as testing techniques for affective responses throughout the design cycle. Much work has also been invested on output technologies, which include facial animation for intelligent agents and incorporating elements of emotion in speech synthesis.

Affective computing can be considered as a form of biofeedback, which involves measurement of physiological information, such as blood pressure, galvanic skin response, or electroencephalographic activity, and it provides real-time feedback to the user. Although this has typically been associated with medical therapy, sports training, and performance art, several applications to human–computer interaction are developing, where the sensed signals are mapped to interesting multimedia display, or used to control elements of a computer game (71). The possibility of sensing brain activity directly to control responses of a computer is of particular importance for those with severe motor impairments, with applications to prosthesis control or biofeedback therapy (72).

RELATED DOMAINS

HCI issues play a prominent role in several domains, of which a few well-known examples are summarized in this section.

Computer-Supported Cooperative Work

Computer-supported cooperative work (CSCW) refers to any group activity supported by computers, whether the users are working simultaneously on multiple computers (groupware), at different times, in different places, or both asynchronously and geographically distributed. Computer-mediated communications applications dominate this field, with email, newsgroups, Wikis, and blogs falling into the asynchronous category, on-line games, instant messaging, and Voice over Internet Protocol, the synchronous category. CSCW also includes data collaboration tools such as a

shared whiteboard and facilities for workspace sharing, such that all users can see and optionally control the same computer display. An important design issue for collaboration in CSCW environments is the mechanism by which awareness of individual and group activities is supported (73).

Social Networks

Social networks constitute a similar domain, in which multiple computer-mediated communication services are employed to connect particular social communities. The massive growth of social networking technologies, which are supported by web-based collaborative tools for both social and business use, led to the coining of terms including *Web 2.0* and the *Semantic Web* to describe the changes in how the World Wide Web is being used. An important early example of social networks is the Usenet (74), which is organized into categories of newsgroups, to which users could post messages. Popular present-day examples include websites such as *MySpace* and *Facebook*, each of which supports multiple forms of communication.

Video-conferencing

An important application domain of HCI to communications technology concerns the design of videoconferencing systems. Not only are audio-visual fidelity and latency (75) important determinants of user satisfaction, but support for gaze awareness and eye contact between participants (76) can significantly improve the perception of *presence* between participants. HCI issues also relate to supporting the social relationships between participants in a video-conference communication, eye contact, gaze awareness, and camera control, for example, whether automatically steered to the active speaker or telemanipulated.

Information Visualization

Information or scientific visualization is concerned with the problems of visual representation and display of large numbers (possibly millions or more) of items, as well as the relationships between them, in an effective manner (77). This concept often extends to choosing the most appropriate means of interacting dynamically with the data contents, performing queries or isolating specific elements of relevance to the user. Shneiderman describes the visual information-seeking mantra as "overview first, zoom and filter, then details on demand." (78). Typical examples of visualized content include files, geographical data, library contents, photographic database, medical histories, and social networks. Work in visualization has been closely tied to advances in computer graphics, and thus, the two fields are often treated together.

Various interface techniques have emerged from work in information visualization, including multi-scale approaches such as the hyperbolic browser (79), varying focus or zoom techniques, and hierarchical (tree or multidimensional) representations (80). Dynamic queries (81) support manipulation of various widgets that control query parameters, as the results are continuously updated.

The related fields of sonification and auditory display use non-speech audio to convey information or perceptualize data (82). These methods often leverage the unique strengths of the human auditory system, which can provide constant awareness of sound in the background and attend to synchronous information from any source position, whereas the visual system is limited to a field of view. Well-known examples include the audible clicks and tones made by Geiger counters, metal detectors, and personal entertainment systems.

HUMAN ERROR AND RISKS

HCI plays a key role in the risks of computer technology and safety-critical systems. Although it is well understood that humans are imperfect, operator error is frequently blamed as the cause of major accidents, even when a problem in design was responsible for the conditions that led to the error in the first place.

The addition of computer technology, software in particular, to an already complex system often exacerbates potential problems and increases the possibility of introducing new ones. Automation tends to shift the responsibility of humans from active control to a more supervisory role, but with less feedback, and reduced opportunities for interaction with the system, informed decision-making becomes difficult. Worse, the system response to a detected problem may hide the underlying cause, which makes the situation all the more perilous, as was the case where the autopilot in a China Airlines 747 compensated for a loss of power to the right outer engine but did not inform the pilots until the limit had been reached (83).

One of the most powerful examples of HCI in computer risks, in which all of the factors above played a contributory role, involved the Therac-25 computerized radiation therapy machine (84). Although the radiation accidents resulted in several deaths and serious injuries, one positive outcome from these events was a critical re-evaluation of safety issues, which led to a better understanding of the role of system and software engineering in the prevention of such accidents.

Systems should be designed to minimize, or in the ideal, eliminate errors. As this approach has its limits, designing for the possibility of error requires that the discovery and correction of errors be made easy and importantly, early. Because many errors stem from legal and otherwise reasonable actions, the consequences of these actions should be made understandable to the user. In a similar vein, users should be empowered to take responsibility for their actions, which thereby promotes a sense of control over the system and involvement in the decision-making process. Finally, mechanisms should be provided to reverse actions and irreversible actions made more difficult (85).

Errors can be classified as *mistakes*, which are inappropriate intentions of action, and *slips*, which are unin-

tended actions. HCI offers design suggestions for minimizing the latter, of which most fall into the categories of *mode errors, capture errors*, or *description errors*. Mode errors result when the user does not know the current mode of the system and performs an action that is incorrect for the current context. Capture errors occur when a frequently performed action of a particular sequence is inadvertently selected instead of that for an intended, but infrequent action. Such errors are caused by an overlap between the two corresponding command sequences. Both mode errors and capture errors can be reduced through better feedback. Finally, description errors refer to an erroneous action being performed because of inadequate specification, which often results from a lack of consistency.

These guidelines follow from basic principles of HCI, yet a failure to respect them in the design and operation phases has contributed to major accidents in the industrial, medical, and transportation fields. Commonly recurring factors in such accidents include poor placement of instrument controls, inconsistent mappings, lack of adequate or correct feedback, and an overload of visual or auditory information to the user. Notable examples include the Phobos I satellite control interface, in which a single letter omission in a series of commands triggered a read-only memory test sequence that resulted in the loss of the spacecraft (86), the drum-pointer altimeter, which, despite frequent misreadings by pilots, remained in operation for many years and possibly contributed to several aviation accidents. Application of design techniques that are based on preventing errors and providing mechanisms for error recovery can greatly improve the quality of computer systems and in turn, reduce the risks of catastrophic failures.

CHALLENGES

The interface between human and computer is ultimately what determines the quality of user experience. As such, this aspect demands attention throughout the product life cycle. Furthermore, the role of HCI in averting risks and minimizing the consequences of error cannot be overlooked.

Industry has been reluctant to adopt the effective methods that resulted from HCI research and practice. Companies that develop software often ignore HCI issues completely. In some cases, they acknowledge the discipline in part by conducting usability testing when products near completion, only to discover problems too costly to correct. Introducing HCI issues to improve products as part of the complete development lifecycle remains an elusive goal. Some have argued that this situation will not change until HCI professionals make a convincing case that their methods will save money, increase product sales, shorten service lines, and reduce returns (87). However, the problem may lie in better communication channels between HCI practitioners and software developers, or in the inclusion of HCI in the education of computer scientists and software engineers.

Shneiderman points to the challenges of universal usability, the goal of designing systems that are usable by everyone, which span the heterogeneity of hardware, software, and network access capabilities, differences among users, and the knowledge gap between user existing and required knowledge to use a system (88).

Other important challenges include better accommodating and supporting human diversity (34) and addressing issues of privacy and trust (89). Application areas such as medicine, bioinformatics, government services including electronic and online voting, and creativity support tools, are just a few of the practical areas that will benefit from and push more advances in HCI.

Arguably, the most significant challenge to HCI remains the design and development of the next generation of interface technology. The hope is that such interfaces will eventually support the full richness of natural language, emotion, facial expression, and bimanual gesture with which humans interact with one another. Regardless of whether the general-purpose PC fades into oblivion, it is doubtful that the human-computer interface we associate with the machines of today will remain the norm.

ACKNOWLEDGMENTS

The author wishes to express his thanks to the various readers who offered their feedback on earlier drafts, in particular, Sidney Fels, Vincent Hayward, Dan Levitin, Jerome Pasquero, François Rioux, and Frank Rudzicz, whose input was invaluable in clarifying several sections of the text. The article benefited most significantly from the insightful critique and additions suggested by Bill Kules, for which the author is most grateful.

BIBLIOGRAPHY

1. J. C. R. Licklider, Man-computer symbiosis, *IRE Trans. Human Factors Elec.*, **HFE-1**: 4–11, 1960.
2. W. L. Bewley, T. L. Roberts, D. Schroit, and W. L. Verplank, Human factors testing in the design of Xerox's 8010 "Star" office workstation, *Proc. CHI '83: Human Factors in Computing Systems*, Proc. New York, 1983, pp. 72–77.
3. G. B. Salomon, Designing casual-user hypertext: the CHI'89 InfoBooth, *Proc. CHI '90: Human Factors in Computing Systems*, New York, 1990, pp. 451–458.
4. T. Hewett, R. Baecker, S. Card, T. Carey, J. Gasen, M. Mantei, G. Perlman, G. Strong, and B. Verplank, Curricula for human-computer interaction, Technical Report, Association for Computing Machinery Special Interest Group on Computer-Human Interaction Curriculum Development Group, 1992, Available: http://sigchi.org/cdg/cdg2.html.
5. W. Hansen, User engineering principles for interactive systems, *Proc. Fall Joint Computer Conference*, 1971, pp. 523–532.
6. J. Raskin, Intuitive equals familiar, *Commun. ACM*, **37** (9): 17–18, 1994.

7. S. K. Card, T. P. Moran, and A. Newell, *The Psychology of Human-Computer Interaction*, Hilldale, NJ: Lawrence Erlbaum, 1983.
8. P. M. Fitts, The information capacity of the human motor system in controlling the amplitude of movement, *J. Exp. Psych.*, **47** (6): 381–391, 1954.
9. I. S. MacKenzie and W. Buxton, Extending Fitts' law to two--dimensional tasks, *Proc. CHI' 92: Human Factors in Computing Systems*, New York, 1992, pp. 219–226.
10. T. Grossman and R. Balakrishnan, Pointing at trivariate targets in 3D environments, *Proc. CHI '04: Human Factors in Computing Systems*, New York, 2004, pp. 447–454.
11. J. Accot and S. Zhai, Beyond Fitts' law: Models for trajectory-based HCI tasks, *Proc. CHI '97: Human Factors in Computing Systems*, New York, 1997, pp. 295–302.
12. W. E. Hick, On the rate of gain of information, *Quarterly J. Exp. Psych.*, **4**: 11–26, 1952.
13. G. A. Miller, The magical number seven, plus or minus two: Some limits on our capacity for processing information, *Psych. Rev.*, **63**: 81–97, 1956.
14. K. Larson and M. Czerwinski, Page design: Implications of memory, structure and scent for information retrieval, *Proc. CHI '98: Human Factors in Computing Systems*, New York, 1998, pp. 25–32.
15. P. Pirolli, *Information Foraging Theory: Adaptive Interaction with Information*, New York: Oxford University Press, 2007.
16. P. Pirolli and S. Card, Information foraging in information access environments, *Proc. CHI '05: Human Factors in Computing Systems*, New York, 1995, pp. 51–58.
17. B. Shneiderman and R. Mayer, Syntactic/semantic interactions in programmer behaviour: A model and experimental results, *Int. J. Comput. Infor. Sci.*, **8** (3): 219–238, 1979.
18. D. A. Norman, *The Psychology of Everday Things*, New York: Basic Books, 1988.
19. A. Edwards, A. D. N. Edwards, and E. D. Mynatt, Enabling technology for users with special needs, *Proc. CHI '95: Human Factors in Computing Systems (Conference companion)*, New York: 1995, pp. 351–352.
20. S. N. Roscoe, The adolescence of engineering psychology, in S. M. Casey, (ed.), *Factors History Monograph Series*, Vol. 1, Santa Monica, CA: Human Factors and Ergonomics Society, 1997.
21. T. S. Tullis, *Predicting the usability of alphanumeric displays*. PhD thesis, Houston, TX, Rice University, 1984.
22. G. Perlman, An axiomatic model of information presentation, *Proc. of Human Factors Society 31st Annual Meeting*, 1987, pp. 1229–1233.
23. D. A. Norman, Affordance, conventions, and design, *Interactions*, **6** (3): 38–43, 1999.
24. J. J. Gibson, *The Ecological Approach to Visual Perception*, Hillsdale, NJ: Lawrence Erlbaum, 1979.
25. G. Torenvliet, We can't afford it!: The devaluation of a usability term, *Interactions*, **10** (4): 12–17, 2003.
26. J. Preece, Y. Rogers, and H. Sharp, *Interaction Design*, New York: Wiley, 2002.
27. T. Winograd, *Bringing Design to Software*, New York: Association for Computing Machinery, 1996.
28. T. H. Nelson, The right way to think about software design, in B. Laurel, (ed.), *The Art of Human-Computer Interface Design*, Boston, MA: Addison-Wesley, 1991.
29. S. Smith and J. Mosier, Guidelines for designing user interface software, Technical Report ESD-TR-86-278, The MITRE Corporation, 1986.
30. J. Nielsen, Heuristic evaluation, in J. Nielsen and R. Mack, (eds.), *Usability Inspection Methods*, New York: Wiley, 1994, pp. 25–62.
31. R. Molich and J. Nielsen, Improving a human-computer dialogue, *Commun. ACM*, **33** (3): 338–348, 1990.
32. D. A. Norman, *The Invisible Computer: Why Good Products Can Fail, the Personal Computer is so Complex and Information Appliances are the Solution*, Cambridge, MA: MIT Press, 1998.
33. B. Tognazzini, http://www.asktog.com/basics/firstPrinciples.html, 2003.
34. B. Shneiderman and C. Plaisant, *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, 4th ed. Boston, MA: Addison Wesley, 2004.
35. R. W. Bailey, *Human Performance Engineering: Designing High Quality Professional User Interfaces for Computer Products, Applications and Systems*, 3rd ed., Upper Saddle River, NJ: Prentice-Hall, 1996.
36. J. T. Hackos and J. C. Redish, *User and Task Analysis for Interface Design*, New York: Wiley, 1998.
37. J. D. Gould and C. H. Lewis, Designing for usability: Key principles and what designers think, *Commun. ACM*, **28** (3): 300–311, 1985.
38. ISO 9241-110:2006, *Ergonomics of Human-System Interaction–Part 110: Dialogue Principles*. Geneva, Switzerland: ISO, 2006.
39. C. Wharton, J. Rieman, C. Lewis, and P. Polson, The cognitive walkthrough method: A practitioner's guide, *Proc. Usability Inspection Methods*, New York: willy 1994, pp. 105–140.
40. C. M. Karat, Cost-benefit analysis of usability engineering techniques, *Proc. of the HFS Society*, 1990, pp. 839–843.
41. R. S. Pressman, *Software Engineering: A Practitioner's Approach*, 3rd ed., New York: McGraw-Hill, 1992.
42. D. Gentner and J. Nielsen, The anti-Mac interface, *Commun. ACM*, **39** (8): 70–82, 1996.
43. I. E. Sutherland, Sketchpad: A man-machine graphical communication system, *Proc. AFIPS Spring Joint Computer Confere.*, 1963, pp. 329–346.
44. D. C. Engelbart and W. K. English, A research center for augmenting human intellect, *Proc. AFIPS Fall Joint Computer Conference*, 1968, pp. 395–410.
45. A. C. Kay and A. Goldberg, Personal dynamic media, *IEEE Comput.*, **10** (3): 31–41, 1977.
46. C. Thacker, Personal distributed computing: The Alto and Ethernet hardware, *Proc. The History of Personal Workstations*, New York, 1986, pp. 87–100.
47. B. Shneiderman, Direct manipulation: A step beyond programming languages, *IEEE Comput.*, **16**(8): 57–69, 1983.
48. J. Johnson, T. L. Roberts, W. Verplank, D. C. Smith, C. H. Irby, M. Beard, and K. Mackey, The xerox star: A retrospective, *IEEE Comput.*, 11–29, 1989.
49. J. Raskin, *The Humane Interface: New Directions for Designing Interactive Systems*, New York: Addison-Wesley, 2000.
50. B. Shneiderman, The limits of speech recognition, *Commun. ACM*, **43** (9): 63–65, 2000.
51. V. Hayward, O. R. Astley, M. Cruz-Hernandez, D. Grant, and G. Robles-De-La-Torre, Haptic interfaces and devices, *Sensor Rev.*, **24** (1): 16–29, 2004.

52. R. A. Bolt, "Put-that-there": Voice and gesture at the graphics interface, *Proc. SIGGRAPH '80: Computer Graphics and Interactive Techniques*, New York, 1980, pp. 262–270.

53. S. Oviatt, Ten myths of multimodal interaction, *Commun. ACM*, **42** (11): 74–81, 1999.

54. P. Milgram and F. Kishino, A taxonomy of mixed reality visual displays, *IEICE Trans. Inform. Syst.*, **E77-D** (12): 1321–1329, 1994.

55. C. Cruz-Neira, D. J. Sandin, and T. A. DeFanti, Surround-screen Projection-based virtual reality: The design and implementation of the CAVE, *Proc. SIGGRAPH' 93: Computer graphics and interactive techniques*, 1993, pp. 135–142.

56. P. Wellner, Interacting with paper on the DigitalDesk, *Commun. ACM*, **36** (7): 86–97, 1993.

57. G. W. Fitzmaurice, Situated information spaces and spatially aware palmtop computers, *Commun. ACM*, **36** (7): 39–49, 1993.

58. J. Rekimoto and K. Nagao, The world through the computer: Computer augmented interaction with real world environments, *Proc. UIST '95: Symposium on User Interface Software and Technology*, 1995, pp. 29–36.

59. E. D. Mynatt, M. Back, R. Want, and R. Frederick, Audio aura: Light-weight audio augmented reality. *Proc. UIST '97: Symposium on User Interface Software and Technology*, 1997, pp. 211–212.

60. G. W. Fitzmaurice, Graspable user interfaces. PhD thesis, Toronto, Canada, University of Toronto, 1996.

61. G. Crampton-Smith, *The hand that rocks the cradle*, *I.D.* 60–65, 1995.

62. B. Ullmer and H. Ishii, The MetaDESK: Models and prototypes for tangible user interfaces, *Proc. UIST '97: Symposium on User Interface Software and Technology*, 1997, pp. 223–232.

63. J. Rekimoto, B. Ullmer, and H. Oba, Datatiles: A modular platform for mixed physical and graphical interactions, *Proc. CHI '01: Human Factors in Computing Systems*, New York, 2001, pp. 269–276.

64. M. Weiser, The computer for the 21st century, *Scientific Ame.*, **265** (3): 94–104, 1991.

65. M. Weiser and J. S. Brown, The coming age of calm technology, PowerGrid Journal, v 1.01, Available: http://powergrid.electriciti.com/1.01, 1996.

66. W. Buxton, Integrating the periphery and context: A new model of telematics, *Proc. Graphics Interface '95*, 1995, pp. 293–246.

67. S. Mann, Wearable computing–toward humanistic intelligence, *IEEE Intell. Sys.*, **16** (3): 10–15, 2001.

68. E. Bergman, (ed.), *Information Appliances and Beyond*, San Francisco, CA: Morgan Kaufmann, 2000.

69. I. S. MacKenzie and K. Tanaka-Ishii, *Text Entry Systems: Mobility, Accessibility, Universality*, San Francisco, CA: Morgan Kaufmann, 2007.

70. R. W. Picard, *Affective Computing*, Cambridge, MA: MIT Press, 1997.

71. D. Bersak, G. McDarby, N. Augenblick, P. McDarby, D. McDonnell, B. McDonald,and R. Karkun, Intelligent biofeedback using an immersive competitive environment, Proc. Ubicomp 2001: Ubiquitous Computing, 2001.

72. A. Nijholt, D. Tan, B. Allison, J. del R. Milan, and B. Graimann, Brain-computer interfaces for HCI and games, *Proc. CHI '08: Human Factors in Computing Systems (Extended abstracts)*, New York, 2008, pp. 3925–3928.

73. P. Dourish and V. Bellotti, Awareness and coordination in shared workspaces, *Proc. CSCW '92: Computer Supported Cooperative Work*, Toronto, Ontario, 1992, pp. 107–114.

74. M. Hauben, R. Hauben, and T. Truscott, *Netizens: On the History and Impact of Usenet and the Internet (Perspectives)*, New York: Wiley-IEEE Computer Society Pr, 1997.

75. W. Woszczyk, J. Cooperstock, J. Roston, and W. Martens, Shake, rattle and roll: getting immersed in multisensory, interactive music via broadband networks, *Audio Eng. Soc.*, **53**(4): 336–344, 2005.

76. W. A. S. Buxton, A. J. Sellen, and M. C. Sheasby, Interfaces for multiparty videoconferences, in K. Finn, A. J. Sellen, and S. Wilbur, (eds.), *Video-Mediated Communication*, Hillsdale, NJ: Laurence Erlbaum, 1997, pp. 385–400.

77. S. K. Card, J. D. Mackinlay, and B. Shneiderman, (eds.), *Readings in Information Visualization: Using Vision to Think*, San Francisco, CA: Morgan Kaufmann, 1999.

78. B. Shneiderman, The eyes have it: A task by data type taxonomy for information visualizations, *Proc. IEEE Visual Languages*, College Park, MD 1996, pp. 336–343.

79. J. Lamping and R. Rao, Visualizing large trees using the hyperbolic browser, *Proc. Proc. CHI '96: Human Factors in Computing Systems*, New York, 1996, pp. 388–389.

80. G. G. Robertson, S. K. Card, and J. D. Mackinlay, Information visualization using 3D interactive animation, *Commun. ACM*, **36** (4): 57–71, 1993.

81. B. Shneiderman, C. Williamson, and C. Ahlberg, Dynamic queries: Database searching by direct manipulation, *Proc. CHI '92: Human factors in Computing Systems*, New York, 1992, pp. 669–670.

82. G. Kramer, (ed.), *Auditory Display: Sonification, Audification, and Auditory Interfaces,* Boston, MA: Addison-Wesley, 1994.

83. N. G. Leveson, *Safeware*, Boca Raton, FL: Chapman & Hall, 1995.

84. N. G. Leveson and C. S. Turner, An investigation of the Therac-25 accidents, *Comput.*, **26** (7): 18–41, 1993.

85. C. H. Lewis and D. A. Norman, Designing for error, in D. A. Norman, and S. Draper, (eds.), *User Centered System Design: New Perspectives on Human-Computer Interaction*, Hillsdalte, Lawrence Erlbaum, 1986.

86. M. M. Waldrop, Phobos at Mars: A dramatic view – and then failure, *Science*, **245**: 1044–1045, 1989.

87. R. I. Anderson, Organizational limits to HCI: Conversations with Don Norman and Janice Rohn, *Interactions*, **7** (3): 36–60, 2000.

88. B. Shneiderman, Universal usability, *Commun. ACM*, **43** (5): 84–91, 2000.

89. S. Patil, N. Romero, and J. Karat, Privacy and hci: Methodologies for studying privacy issues, *Proc. CHI '06: Human factors in Computing Systems*, New York, 2006, pp. 1719–1722.

FURTHER READING

A. Dix, J. Finlay, G. Abowd, and R. Beale, *Human-Computer Interaction* Hertfordshire, UK: Prentice Hall, 1998.

B. Shneiderman and C. Plaisant, *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, 4th ed. Boston, MA: Addison-Wesley, 2004.

R. Baecker, J. Grudin, W. Buxton, and S. Greenberg, *Readings in Human Computer Interaction: Towards the Year 2000*, San Francisco, CA: Morgan Kaufmann Publishers Inc., 1995.

J. Preece, Y. Rogers, and H. Sharp, *Interaction Design: Beyond Human-Computer Interaction*, New York: Wiley, 1992.

B. Tognazinni, Ask Tog: Interaction design solutions for the real world, Available: http://www.asktog.com.

D. Norman, jnd website, Available: http://www.jnd.org

J. Nielsen, Useit.com: Usable information technology http://www.useit.com

JEREMY R. COOPERSTOCK
McGill University
Montreal, Quebec,
Canada