interface

Jonathan Grudin

Aarhus University (on leave from MCC)
Computer Science Department
Ny Munkegade Bygning 540
8000 Aarhus C Denmark

"Terror. You have to confront the documentation. You have to learn a whole new language. Did you ever use the word 'interface' before you started using a computer?"

-- Advertising executive Arthur Einstein[1]

interface: ... a: the place at which independent systems meet and act upon or communicate with each other b: the means by which interaction or communication is effected at an interface

-- Webster's Seventh New Collegiate Dictionary

INTRODUCTION

This is an essay on "the user interface" to a computer and "the computer interface" to a user or users. It also addresses "a user's interface" (or "a group's interface") to a computer and "a computer's interface" to a user or users. After noting the further distinction of users' interfaces to their *work*, it concludes with a discussion of "the designer" and designers' "models of users."

The goal is not to split semantic hairs. "At a certain stage in the development of every science a degree of vagueness is what best consists with fertility," wrote William James (1890), and the field of human-computer interaction has not yet advanced beyond that stage. The goal is to show that the way we use these words conceals important changes in our field. The term "user interface" came into use when our field was very different than it is now. At that time, it served a useful purpose. Most of us now feel quite comfortable with it, although we may not use it entirely consistently. This paper details the possibilities for confusion and misdirection in our use of this and related terms in the changing environment of computer design and use. Perpetuating the current usage may reenforce and bind us to an obsolete perspective. The power of words is not total, but they may subtly and indirectly inhibit the adoption of new areas of research and approaches to development.

The term "user interface" originated in the engineering environment. Virtually all computer users had been engineers and programmers, but a new kind of user was emerging: the non-programming user. These users often reacted more negatively to difficulties in dealing with the machine. Easier forms of interaction were needed, a new

interface -- attention flowed to "the user interface." In the next section, the word "user," which was helpful in early engineering environments, is shown to be problematic in today's broader context. Then the term "interface" is explored, noting that a user's interface to a computer does *not* match or complement a computer's interface to a user. "User interface" is often used to describe a computer's interface to the user, rather than a user's interface to the computer. Finally, the static concept of "the interface designer" is contrasted with the radically changing roles of actual system designers. All three analyses support one conclusion: our use of terms that originated in the engineering environment systematically obscures changes that have occurred in how interfaces are designed and used. Of particular concern is the concealment of aspects of the environments in which computers are now used -- work environments that are usually very unlike the engineering environment. This distortion is unfortunate because understanding these work environments is increasingly important in computer systems development.

1. "USER"

Ironically, "user interface" is a technology-centered term. The interface is between users and computers. We have asymmetrically abridged "user-computer interface," retaining the name of only one of the two actors. The computer is assumed, the user must be specified. There was a good reason for this: In the engineering environment, a computer's architecture includes many internal "interfaces." The interface to the user was one of many interfaces that had to be discussed, so labelling it "the user interface" was an obvious and noncontroversial choice. There was no need to call it the "user-computer" interface, because in the engineering environment, the computer could indeed be safely assumed! But while "user" was a convenient identifier among engineers, its use has spread beyond the engineering environment, creating confusion in several ways.

Computer users don't consider themselves "users." Terms such as "user manual" were initially resisted, in part because some people associated "user" with "drug user." In contrast, similar manuals for automobiles are called "owner's manuals"; they have little to do with the rights and responsibilities of car ownership, but their readers typically do identify themselves as owners. As computer use becomes more commonplace, people who happen to spend some time working with computers are less likely to identify themselves naturally as "users."

The term "user" retains and reenforces an engineering perspective. "Casual users" is a term often used to describe managers and executives -- who are often not "casual" at all! "Novice" or "naive" users are often *expert* or *sophisticated* at their jobs -- while the expertise of "expert users" may not extend beyond computer use. These terms simply assume that everything is in reference to a computer. This systematically distorts our perception of the user-computer partnership (Bannon, 1990). This is unfortunate because attention to the work environment of computer users is a critical element of interface design, and our terminology *conceals* the variability in users' environments by selectively focusing on that which is shared: computer use. For example, it might be helpful for designers to think of the people at the terminals as "skilled nurses" or "experts at writing" rather than as "naive" or "casual" users.[2] Gaines and Shaw (1986) identify other examples of problematic terminology: "non-professional user" and "non-specialist user" (meaning not computer professionals or specialists; again, assuming computers to be the domain of importance). They note that this perspective can create an unfortunate attitude toward users, exemplified by a paper describing "idiot-proof programs."

The term "user" suggests that there exists a typical user or range of users. When we catch ourselves talking about "the user," we may hastily remind ourselves that there are different kinds of users -- novice, casual, and expert, for example (!!) -- but even then we may envisage a static cross-section or range of computer users. However, the user population has been changing radically. The first computer users were engineers and programmers. In the past decade, non-programmers, or so-called "end users" (yet another

term rooted in the engineering perspective),[3] have become the principal computer users. And another shift in the target of computer design is gathering momentum, as the CSCW conference series itself indicates -- the shift toward regarding a group or organization as a collective computer user. The term "user interface" conceals this change. Computers have always had interfaces to their users, but the term was not used when all users were programmers and engineers, and will be awkward for systems that are designed for groups. Malone (1985) has suggested a new term, "organizational interface," to capture this second shift toward collective support.

There is a fundamental continuity of interface development that is obscured by segmenting history into "programmer interface," "user interface," and "organizational interface" periods. If one considers instead the *"computer interface"* to the user and the world, a smooth progression emerges. The computer interface to its environment has moved steadily away from the hardware of the computer itself, out into the environment -- first to the software, then to the individual at the terminal, and now to the work group (Friedman, 1989; Grudin, 1990a). The history of interface development appears more coherent if we position ourselves at a distance and think of this "computer interface" to the world. This perspective affords us a single view that takes in the period before the term "user interface" was used and extends more gracefully into a future of widespread computer support for groups and organizations.

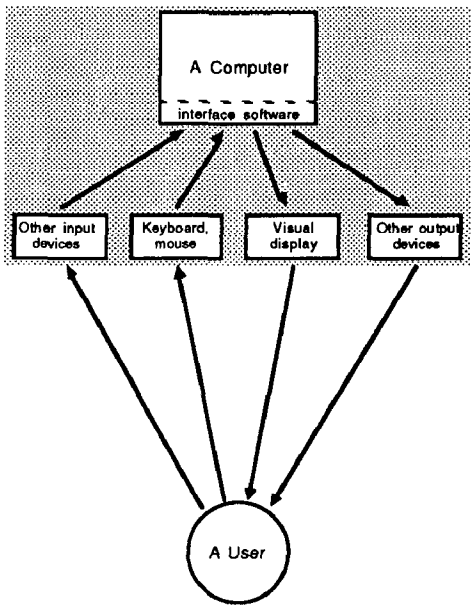
Should we abandon the use of "user interface" and adopt "computer interface"? Perhaps not, for we shall see that each of these terms has its uses. But when no compelling reason exists for doing otherwise, neutral terms such as "the interface," "the human-computer interface," or "human-computer interaction" seem preferable. And there is value in sometimes thinking in terms of the "computer interface" -- taking *users* as a given, for a change. It removes one from the engineering context and naturally guides one to regard computers from the perspectives of their users. Consider the following exercise: Describe the human-computer interface to "non-technical" people, neutrally avoiding specific terms, and then ask, "Which term best captures what I have described: *the user interface* or *the computer interface*?" Not surprisingly, I find that they choose the latter.

2. "INTERFACE"

Following a recent talk, someone observed that I had not explicitly defined "interface," noting that to him, the word signified the segment of the software program that handled dialogue with users. We generally do identify "the user interface" with the software that controls I/O devices and processes. This is reflected in the name "User Interface Management System," for example. Similarly, the "User Interface Group" of a development project may consist entirely of Software Engineers. This view of the interface is captured in Figure 1.

Consider the two faces to the user-computer interface. Is a user's interface to a computer the mirror image of the computer's interface to the user? It may seem that it should be, but on reflection it is not, unless one defines "interface" extremely narrowly. The user's interface to the computer may *center* on the software-controlled dialogue, but it also includes any documentation and training that are part of using the computer. It includes colleagues, consultants, system administrators, customer support and field service representatives, when they are available. These artifacts, processes, and people are so significant in shaping our interactions with a computer that it is myopic not to see them as part of a user's interface to the computer (e.g., Clement, 1990). In fact, those responsible for documentation and training overwhelmingly feel that these should be developed in concert with the software interface, however infrequently this occurs in practice (Grudin and Poltrock, 1989).

Figure 1. A computer's interface to a user



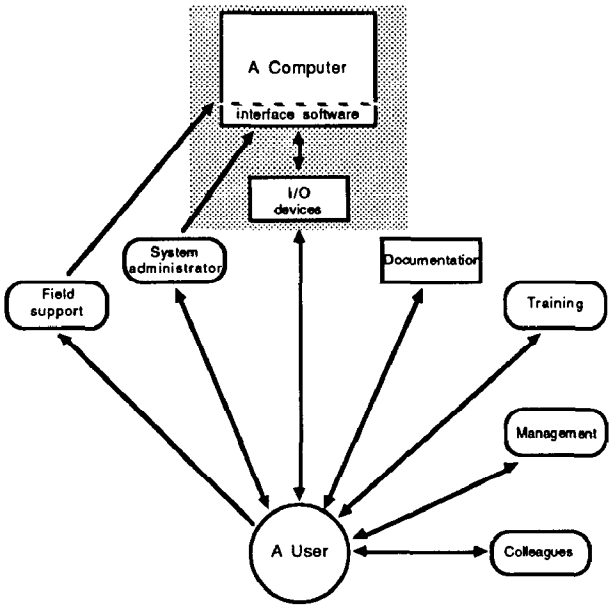


Figure 2. A user's interface to a computer

Figure 2 illustrates a user's interface to a computer. The user consults documentation, is trained, and solicits advice from colleagues, system administrators, and others. The system administrator may modify the system on behalf of the user. Following a hardware failure (a less frequent part of the user-computer interaction as hardware reliability improves), a field service engineer may directly modify the system. These activities shape the nature of the interaction that takes place through the input and output devices.

The computer's interface to users, on the other hand, is quite reasonably defined as being the software controlling the dialogue -- the engineering perspective shown in Figure 1. Unlike its users, the computer does not consult with nearby people or objects. (Or does so relatively little. A system administrator who sets up an environment for a user is part of the computer's interface to that user, just as someone who advises a user is part of that user's interface to the computer.) In conclusion, our equation of "the user interface" to software and I/O devices means that *"user interface" denotes the computer's interface to the user* (Figure 1), *not the user's interface to the computer* (Figure 2). A good demonstration of our acceptance of this circumscribed use of the term "user interface" is in Gould's (1988) excellent chapter on "designing for usability." He also emphasizes the importance to "usability" of documentation, training, field support, etc., but he restricts the use of the term "user interface" to I/O devices. Again, the original engineering perspective is perpetuated in our use of terms. And once again, this may subtly discourage a "user perspective" -- a focus on the users, their work, and their environments.

In her book *Through the Interface,* Bødker (1990) observes that computers are a tool through which people interface with *their work.* Sayeki (1989) also distinguishes between the human-computer interface and the interface between the human working with the computer and the world. He illustrates this aspect of tool use by considering a blind man with a cane. The handle of the cane is one interface. The point of the cane is another, providing contact with the world. The former is important for the quality of access it provides to the latter -- the interface to the world is a tool user's principal concern. Our use of the word "interface" is restricted to the interface to the tool, instead, which rightly or wrongly tends to be a tool *developer's* principal concern.

As more advanced computers appear, the computer interface to users is changing. The computer interface may expand to include on-line documentation, on-line help, and on-line training, reducing the need for mediators. More reliable hardware eliminates other intermediaries. Better software -- and interfaces -- reduce our reliance on system administrators and consultants. These advances offer the hope that the computer interface and the user interface may come into closer alignment, as shown in Figure 3. In this optimistic view, the computer has assumed many of the support roles, even summoning hardware support directly when needed. Colleagues may always play a role, although perhaps focused more exclusively on higher-level aspects of a user's tasks. But the time when a user's interface meshes with the computer's interface is not yet here. We need to use all the means at our disposal to focus on the work environments of computer users, loosening our ties to the engineering perspective that served us well in the past and that lives on in our patterns of speech and thought.

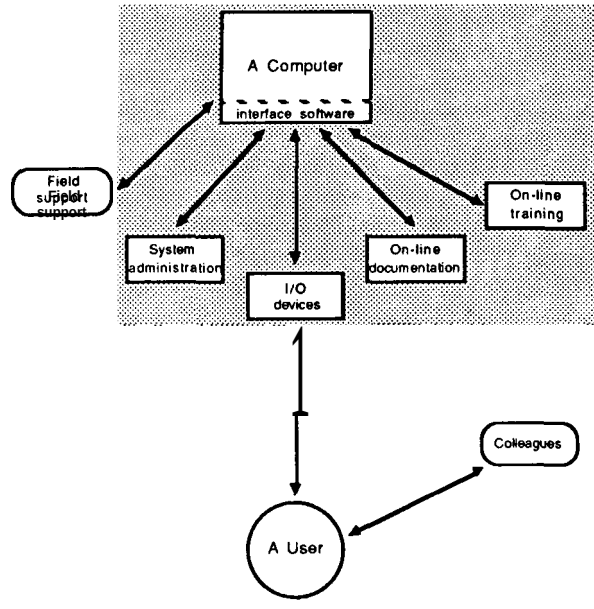


Figure 3. A user's interface to a hypothetical future computer

3. "DESIGNER"

Finally, consider our colleagues, interface designers. We have noted that the terms "user" and "interface" inadvertently establish an engineering focus. The term "designer" more naturally evokes the engineering context, of course, but its use, too, has gradually come to disguise a changed reality. In brief, it encourages a view of "the designer" as an individual who assumes (or should assume) responsibility for all aspects of an interface design, bringing together broad knowledge of computer users and systems. This view, perhaps once valid, conceals the cumulative and collaborative nature of most system design today. By reenforcing an obsolete perspective, researchers' use of terminology puts them at risk of misdirecting their efforts.

Many researchers, concerned with deep issues of representation, have addressed designers' cognitive or "mental" models of computer systems and computer users. Some wrote before the changes in the field noted here took place and perhaps all would agree with the

conclusions of this paper. Their central points may be valid -- but the precedents they established for the use of terminology should be examined.

In a seminal work on mental models in the context of computing, Moran (1981) proposed that "to design the user interface of a system is to design the user's model. The designer needs to understand... what knowledge goes into the user's conceptual model." Young (1983) asked whether "the Designer should be encouraged to share the... User's conceptual model (of the system)" or whether "the Designer will need to employ a cruder User's conceptual model..." Carroll (1984) distinguished "the designer's model (that is, the understanding that the analyst develops of what it is that the user knows)" from the user's model. In a similar vein, Norman (1986) described "the generalized 'typical user' model that is what the designer develops to help in the formulation of the... conceptual model (of the system) held by the designer." In a *tour de force* treatment, Streitz (1988) reviews these works and devises a formalism for representing models. For example, D(U(f)) represents "the designer's conceptualization of the user's mental model (of the functionality of a system)." A yet more complex elaboration is proposed by Nielsen (1990), in which a "user description model (or) type DU model (is often) whatever the designer is thinking about the users."

The "interface designer" described in such papers is typically regarded as designing only the software dialogue (although including off-line documentation in Norman, 1986 and Nielsen, 1990, and training in Streitz, 1988). This again obscures the context of use and reenforces the engineering perspective, "the computer's interface to the user." But a more significant problem is that a focus on "the designer's model of the user" obscures *design* contexts. To begin with, just as variability in work contexts leads to differences among users, development contexts vary enormously, which strongly influences design practice (Grudin, 1990b). In addition, an image is suggested of "the designer" working in isolation. This conception of "the interface designer" was once relatively veridical -- each interactive system or application required a software interface and the programmer started with a more or less blank slate. But this no longer resembles typical design practice: designers do not work in a vacuum. Speaking of "the designer," while not entirely illegitimate, obscures the forces separating designers in many design contexts from the need to work with such "user models."

Design knowledge is cumulative and often implicit. Although some interface design knowledge is "retired" by advances in technology, we do learn from our collective experience. A substantial body of interface lore has been accumulated through research and trail-and-error development. Even without the conservative pressures of the "installed base," we have reason to stand on the shoulders of those who came before us. The well-known SRI and Xerox PARC antecedents of the innovative Macintosh interface illustrate the continuity of interface development. And designers often use precedent without absorbing an underlying knowledge or model. For example, one can make use of a pair of contrasting colors that have proven effective without knowing the model of human color vision that first generated them (or, for that matter, whether they were first discovered through trial and error instead).

Interface design responsibilities are distributed. Interfaces have become too complex for one person to manage well. In a large product development company, product marketing may define some of a new product's interface features, software engineers may do the bulk of the interface design, with human factors engineers contributing to some features and graphic or industrial design engineers contributing to others. If the entire "users' interface to the computer" is considered, technical writers may be responsible for the documentation, an education group may develop training, and so forth. They all contribute to the interface design. They may become involved at different times and may not even communicate with one another (Grudin and Poltrock, 1989). Interface design specialization continues: professionals in linguistics, video, sound, and other areas are

more frequently consulted. Smaller development organizations, lacking the resources to employ specialists, may have to hire consultants -- or simply license or "borrow" interface concepts in circulation.

"New" interfaces are constrained by existing interfaces. Computer users are increasingly likely to have habits or expectations that cannot be ignored. For example, a new spreadsheet design for PCs that incorporates dialogue conflicting with that of Lotus 1-2-3 has severely limited chance of acceptance. The growing tendency of vendors to seek a "consistent look and feel" within a product family further encourages the borrowing of design elements. Hardware and software advances that facilitate platform-independent software and isolate aspects of interface code makes such "reuse" easier. Formal and de facto standards make it necessary. A design can be borrowed without a complete understanding of the original underlying model, if one even existed. The risk of unexpected consequences in the new setting may be reduced through acquiring such an understanding, of course. But that risk is not always great and may also be handled through prototyping and iterative design practices.

In summary, each designer at most contributes incrementally to a large enterprise, helping to refine a collective design. It is a poor choice of words to say, "it is still the designer who does the implementation" and "all models... are 'in the head' of a person," (Streitz, 1988). Modern interfaces incorporate a tremendous amount of implicit knowledge of computer users -- their perceptual psychology, motor coordination, cognitive psychology, even a little social psychology -- but the totality of this knowledge is held by no one individual. In general, while an individual can make a difference, the design of human-computer interfaces evolves under the pressures of technological and social change, which act through designers as a group, distributed over space and time. The fiction of "the interface designer" focuses our attention on the small part of the human-computer interface managed and understood by one person. It diminishes our appreciation of the continuous growth of implicit understanding of human psychology and work organization represented in the design of computer systems.

The researchers quoted above were addressing this complexity of design, but their use of terms may have diminished their own effectiveness. For example, Moran's (1981) framework was established to help designers focus systematically on different interface "levels" and predated most existing work on interface design. Today, the image of "the designer" working at all interface levels may encourage researchers to overemphasize low-level issues that are more tractable but that are increasingly standardized and ignored by real design teams (Grudin, 1989). Similarly, Barnard and Harrison (1989) proposed a framework for developing a formal, cumulative model of computer users and tasks with the commendable aim of freeing designers from having to keep all relevant information "in the head." The goal is a simulation model that is "a means of encapsulating designer hypotheses about how the user perceives the system," with structures that "will vary depending on which theories are in the designer's mind about the potential use of the system." But for the low-level tasks illustrated in the paper, today's interface designers may keep little of this information in mind, and may do little such hypothesizing. Therefore, such a model may be of limited utility in real design environments. Although these papers note that designers must address a range of interface issues, from "low-level" perceptual-motor to "higher" cognitive and even social aspects of human-computer interaction, each has led to a research paradigm that focuses selectively on the "low" end of the spectrum. This choice may in part be due to the greater tractability of low-level issues, but may be further encouraged by the misleading images of "the designer" and "the user" removed from their real work contexts. Research at these lower levels has made a scientific contribution, but its significance for systems development can only be assessed in the actual design and use environments that the terminology obscures.

Streitz (1988) suggests that "the user also makes inferences about the designer's conceptual model," and "the user (asks) him or herself such questions as 'what might the designer have had in mind when designing the sequence of interaction this way?'" I am not sure that users do ask such questions, but if they do, they may be wrong to do so. They might better ask, metaphorically or anthropomorphically, "why did the computer do that? What does it have in mind?"

The anthropomorphic view of the computer is just one view, bringing with it some unwanted baggage. But it has the virtue of identifying computer systems as highly complex artifacts with independent existence in the world, products of a continuous design process that has engaged countless individuals over decades, during which time an impressive level of implicit understanding of people and their work has been incorporated. The terminology brought forward from the field's origins in engineering can obscure this history and can obscure the need to maintain and even strengthen the focus on human psychology and work.

---

[1] Quoted in Wright (1990).

[2] Of course, developers should consider the experience level of the eventual users of a system. But this will follow inevitably from acquiring a deep understanding of the users' work environments, whereas the converse is not true: little about the work environment is necessarily learned through following general advice to "consider the needs of novice, casual, and experienced users."

[3] Don Norman pointed out the following passage from Liddle (1989): "...We don't think about the end user, in terms of their job. Most of us don't like to pay much attention to end users at all. And we certainly don't know what they want to do in their job. I mean they're not even in the computer industry! End users, in fact, are really interesting. 'End users' is a term I like, originally coined by IBM. 'End users' is a retronym. ... A retronym is a word that used to have a perfectly good meaning and now has to be retrofitted by a leading adjective. Like an analog watch. You know, (a watch) used to be this thing with hands that went around, and then this miraculous thing, the digital watch appeared. And now all watches are digital. So if you happen to have one like this, you have to say that it's analog. 'Human-readable' falls into this category. Anyway, we call them end users because 'Oh, you mean the end! Oh, that guy out there! I thought you meant the MIS guy that's more reasonable and rational. God! The end user?'"

ACKNOWLEDGMENT

Members of the Aarhus University Information and Media Sciences Department inspired this paper by probing my use of these terms. Phil Barnard, Susanne Bødker, Tom Erickson, Hiroshi Ishii, John Bowers, Allan MacLean and Jakob Nielsen provided helpful suggestions.

REFERENCES

Bannon, L., 1990. From human factors to human actors. In J. Greenbaum and M. Kyng (Eds.), *Design at work.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Barnard, P. and Harrison, M., 1989. Integrating cognitive and system models in human computer interaction. In A. Sutcliffe and L. Macaulay (Eds.) *People and computers V.* Cambridge: Cambridge University Press, 87-103.

Bødker, S., 1990. *Through the interface: A human activity approach to user interface design.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Carroll, J.M., 1984. Mental models and software human factors. *IBM Research Report RC 10616.* Yorktown Heights, NY: IBM.

Clement, A., 1990. Cooperative support for computer work: A social perspective on the empowering of end users. In these Proceedings.

Friedman, A.L., 1989. *Computer systems development: History, organization and implementation.* Chichester, UK: Wiley.

Gaines, B.R. and Shaw, M.L.G., 1986. From timesharing to the sixth generation: the development of human-computer interaction. Part 1. *Int. J. Man-Machine Studies, 24,* 1-27.

Gould, J.D., 1988. How to design usable systems. In M. Helander (Ed.) *Handbook of Human-Computer Interaction.* Amsterdam: North-Holland.

Grudin, J., 1989. The case against user interface consistency. *Communications of the ACM, 32,* 10, 1164-1173.

Grudin, J., 1990a. The computer reaches out: The historical continuity of interface design. In *Proc. CHI'90 Human Factors in Computing Systems,* (Seattle, April 1-4).

Grudin, J., 1990b. The development of interactive systems: Bridging the gaps between developers and users. Manuscript submitted for publication.

Grudin, J. and Poltrock, S., 1989. User interface design in large corporations: Coordination and communication across disciplines. *In Proc. CHI'89 Human Factors in Computing Systems,* (Austin, April 30-May 4).

James, W., 1890. *The principles of psychology.* New York: Holt.

Liddle, D., 1989. What makes a desktop different. In S. Alsop (Producer), *Proc. Agenda 90,* 69-70.

Malone, T.W., 1985. Designing organizational interfaces. In *Proc. CHI '85 Human Factors in Computing Systems,* (San Francisco, April 14-18).

Moran, T.P., 1981. The Command Language Grammar: a representation for the user interface of interactive computer systems. *Int. J. Man-Machine Studies, 15,* 3-50.

Nielsen, J., 1990. A meta-model for interacting with computers. *Interacting with Computers, 2*, 2.

Norman, D.A., 1986. Cognitive engineering. In D.A. Norman and S.W. Draper (Eds.), *User centered system design.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Sayeki, Y., 1989. Human interface and cognitive engineering. *Information Processing* (Journal of the Information Processing Society of Japan), *30*, 1, 2-14.

Streitz, N.A., 1988. Mental models and metaphors: Implications for the design of adaptive user-system interfaces. In H. Mandl and A. Lesgold (Eds.), *Learning issues for intelligent tutoring systems.* New York: Springer-Verlag.

Wright, K., 1990. The road to the global village. *Scientific American*, March, 57-66.

Young, R., 1983. Surrogates and mappings: Two kinds of conceptual models for interactive devices. In D. Gentner and A.L. Stevens (Eds.) *Mental models.* Hillsdale, NJ: Lawrence Erlbaum Associates.