HUMAN COMPUTER INTERACTION, 1985, Volume 1, pp. 209-242

The Prospects for Psychological Science in Human-Computer Interaction

Allen Newell
Carnegie-Mellon University

Stuart K. Card
Xerox Palo Alto Research Center

ABSTRACT

This paper discusses the prospects of psychology playing a significant role in the progress of human-computer interaction. In any field, hard science (science that is mathematical or otherwise technical) has a tendency to drive out softer sciences, even if the softer sciences have important contributions to make. It is possible that, as computer science and artificial intelligence contributions to human-computer interaction mature, this could happen to psychology. It is suggested that this trend might be prevented by hardening the applicable psychological science. This approach, however, has been critized on the grounds that the resulting body of knowledge would be too low level, too limited in scope, too late to affect computer technology, and too difficult to apply. The prospects for overcoming each of these obstacles are analyzed here.

This essay presents the substance of an address by the first author to the ACM CHI '85 Conference on Human Factors in Computing Systems in San Francisco, California, on April 14, 1985.

Authors' present addresses: A. Newell, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA 15213; S. K. Card, Intelligent Systems Laboratory, Xerox Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, CA 94304.

CONTENTS

1. INTRODUCTION

Human-computer interaction is clearly thriving. To index the accelerating activity, one need only point to the ACM CHI (Computer-Human Interaction) and the IFIP Interact meetings, and to the recent appearance in the field of two new journals, *Behavioural and Information Technology* and *Human-Computer Interaction*. The field of human-computer interaction is a confluence of many disciplines: computer graphics, human factors, cognitive psychology, and artificial intelligence. In this paper we wish to single out one of these—psychology.

What are the prospects of psychology playing a *significant* role in this growing field? The question is important for two reasons. First, the improvement of interaction between humans and computers is a universally shared goal. The use of results from psychology is viewed by many to be a key to this effort—we share this view. Second, a considerable amount of psychological research is beginning to be done on human interaction with computers. Thus, assessments of whether basic and applied psychological research will pay off in improved designs are useful to guide individual researchers' choices of research problems and funding agencies' choices of individual researchers.

The case for psychological research might seem hardly to need arguing. With a name like human-computer interaction, it would appear that the role of psychology is firmly established. Yet parents frequently name their children after great men and women with no apparent impact on the child's eventual greatness. The truth is that the success of psychology in this field can by no means be assured. This is the issue we wish to assess here.

We first discuss the problems psychology will have in playing a significant role in the development of human-computer interaction, then present a vision of what this role might be. Next we turn to the problems of this vision. Finally, we address the vision's prospects. Our focus on psychology does not mean that other aspects of human-computer interaction are not vital. Indeed, many of the other technologies and their supporting sciences are clearly vital. Our concern is to assess whether psychology is also vital.

Such a concern does not lend itself to a purely technical discussion, but includes large doses of opinion on the state of our field and its disciplinary structure. This paper, therefore, is intended as an essay—and a bit of an exhortation.

2. THE ROLE OF PSYCHOLOGY IN HUMAN-COMPUTER INTERACTION

2.1. Gresham's Law

Let us begin by considering a venerable law of economics. During the mid-16th century, Sir Thomas Gresham, a founder of the London Royal Exchange, formulated a law to describe the fact that newly minted coins tended to disappear from circulation, leaving only the worn ones. Presumably, this was because the new coins were intrinsically worth more melted down than the older coins of the same denomination. Succinctly stated, Gresham's law is that:

Bad money drives out good.

A similar phenomenon appears to govern in the intellectual economics of science, namely that:

Hard science drives out soft.

Given the existence of some hard science—quantitative or otherwise technical—and the existence of some soft, qualitative science, the hard science will tend to be used and the soft science ignored, regardless of whether all the important issues are within the scope of the hard science. As a consequence, when hard science appears, those issues that have mostly soft science associated with them tend to be ignored. This phenomenon occurs despite an explicit belief by all concerned that the qualitative factors are important. Qualitative factors are left aside when they do not fit into the technical analysis.

The principle can be given meaning by a few examples:

Example 1: Operations Research. In operations research and management science there has long been a concern that it is difficult to cause qualitative factors to be taken into account. The hard side of operations research consists of linear programming, queueing theory, and mathematical models of assorted pedigrees. The soft side consists of issues of values, of manager's beliefs and expectations, and assorted conceptual entities, such as a firm's good will. For instance, although entrepreneurs can bargain about the worth of a firm's good will and assign a dollar figure, analysts cannot, as a practical matter, include good will in a linear programming model to compute its change with varying circumstances. In fact, there is a substantial literature in operations research decrying the lack of qualitative considerations. Yet, despite this protest literature which reaches back to the beginnings of operations research, hard science continues to drive out soft.

Example 2: Human Factors. Consider next the difficult life and times of human factors. Human factors as a discipline goes back 40 years to World War II (although its industrial engineering roots go back another 20 years). There has certainly been much brave talk about the central role that human factors should play in the development of machines. And indeed, there have ensued some real, if modest, gains, such as in the aircraft industry, where human factors has played a genuine role in cockpit design. But after all these years there remains a continuous stream of discussion within the field of how human-factors specialists are not taken as seriously as they would wish to be. Muckler's lament in a recent issue of the *Human Factors Society Bulletin* is typical:

> Many computer system designers appear to have no knowledge of human factors, are not aware that the human-computer interface is vital to their systems, or that a substantial human-factors database exists to help them. (Muckler, 1984, p. 1)

There is thus much evidence in the human-factors literature that human-factors practitioners are not really in the center of the world in which they work. Hard science, in the form of engineering, drives out soft science, in the form of human factors.

Example 3: Programming Languages. A third example comes from the computer-science world: Millions for compilers but hardly a penny for understanding human programming language use. Now, programming languages are obviously symmetrical, the computer on one side, the programmer on the other. In an appropriate science of computer languages, one would expect that half the effort would be on the computer side, understanding how to translate

the languages into executable form, and half on the human side, understanding how to design languages that are easy or productive to use. Yet we do not even have an enumeration of all of the psychological functions programming languages serve for the user. Of course, there is lots of programming language *design*, but it comes from computer scientists. And though technical papers on languages contain many appeals to ease of use and learning, they patently contain almost no psychological evidence nor any appeal to psychological science.

Some research on the human side does exist, of course. But imagine a scale with Shneiderman's *Software Psychology* (1980) and an armload of books weighed against the two volumes of Aho and Ullman (1972) and the library of books on compiler construction, parsing, program specification, correctness proofs, denotational semantics, applicative languages, LR(k) grammars, and structured programming. Relative to what is found on the human side, the technology of programming languages is technical and deep—all those algorithms to know, all those theorems on syntactic language classes, all those operations on data structures. There are also things to know with respect to the psychology of programming languages—but far fewer and much of that consists of the details of idiosyncratic experiments. The human and computer parts of programming languages have developed in radical asymmetry (Card & Newell, 1984). This comparison does not imply that knowledge about the human is less useful than knowledge about compilers, it just shows the operation of Gresham's law. Interestingly, the technology of programming languages has its own ways of diverting attention from the missing human part of the enterprise. Currently, the panacea in the programming world is "rapid prototyping." The idea is that if a designer has sufficiently rapid feedback from what happens when a user uses a program, nothing else is required. Rapid prototyping thus bypasses the need to know anything about the human. In other words, hard science, in the form of computer technology, drives out the soft science of the user.

2.2. The Hard and Soft Sciences of the Interface

The hard science of the human-computer interface is computer science—the technologies of computer graphics, command languages, interface programs, and microcomputers. This is true even though computer science itself is often viewed skeptically by the harder-yet sciences of physics and chemistry. Be that as it may, computer science is still far more mathematical and technical than the soft sciences of the interface, and this relative difference is all that is required for Gresham's law to work.

The soft science of the human-computer interface is, of course, psychology. But consider the maxims often quoted in books about designing interfaces:

> Know the user. (Hansen, 1971, p. 528)
> Communicate with metaphors. (Heckel, 1984, p. 41)

> For many applications it may be desirable to distinguish blanks (keyed spaces) from nulls (no entry at all) in the display of data forms. (Smith & Mosier, 1984, Guideline 2.1.2–15, p. 117)

Even though these maxims are considered part of the appropriate lore for human engineering the interface, they have almost no technical psychological content. They require no real contribution, experimental or theoretical, from a psychology of the user. They derive, essentially, from a little common sense, plus placing a value on serving the user well.

And now comes artifical intelligence. Yet another technology and another science with something to contribute to the interface. At least when compared to psychology, AI is, if not a hard science, then a harder science. With AI it is possible to apply hard science to the cognitive aspects of the interface, yet not to be psychological. Interestingly, AI also has its peculiar ways of diverting attention from the missing human part of the science, namely, if the interface is intelligent, then it is not necessary to know anything about the user, because the interface will be able to interact with the user intelligently.

3. THE VISION

The burdens of Gresham's law were already clear when (with Tom Moran) we began our research into human-computer interaction at Xerox PARC in the mid-1970s. But we also had a vision at the time of how psychology could beat Gresham's law and play a significant (we hoped enabling) role in the progress of human-computer interaction. Already in the 1970s, cognitive psychology seemed sufficiently ripe to support the development of an applied information processing psychology for human-computer interaction. Over the next few years, the vision grew for the shape of such contribution.

One principle of this vision is that design is where the action is, not evaluation. Evaluation time is too late for any but minor adjustments to a system; only at design time are there an adequate number of degrees of freedom to make a difference, including the ability to make trades with parts of the system that have no direct connection with the user. If design is where the action is, then the designer is the point man. Moving responsibility away from the designer to human-computer interaction specialists organized as advisers, evaluators, consultants, overseers, even bosses—is to attenuate the effect seriously. Bridges are designed by teams of engineers, not by some breed of professional designers who know nothing about structural engineering, but call upon engineering consultants or submit their designs to engineering evaluation (although both activities are important supplements for specific purposes). Likewise, computer interfaces are to be designed by teams of engineers who know about human behavior at the interface, as well as about the internal mechanics of making the interface work.

A second principle is the need for an engineering-style theory of how the user interacts with the computer. This implies three important characteristics: task analysis, calculation, and approximation. *Task analysis* means predicting important factors about user behavior from a symbolic description of the task to be done. *Calculation* means that predictions are to be made by explicit operations on mathematical models of the situation, rather than by some other means—the judgment of the theorist, the experience of the practitioner, the assessments of users, or even the empirical evaluation of systems. *Approximation* reflects an appreciation that human behavior is so complex that real-world calculations cannot be expected to be very accurate—so complex, in fact, that unless calculation with very approximate theories is accepted, success is not possible. It is not that more approximation is better. Rather, approximation is a necessary ingredient in the right kind of theory. However, hidden in the prescription to approximate is the belief that the developing science will gradually replace poor approximations with better ones, whereas little progress can be made by waiting for initial theories that are highly accurate on the first bounce.

We developed a line of research that embodied and substantiated this vision. The result was a 1983 book (with Tom Moran), *The Psychology of Human-Computer Interaction,* hereafter abbreviated *PHCI.* The book provided some approximative and calculational theories useful for task analysis: the *Model Human Processor,* the *GOMS* family of models, and the *Keystroke-Level Model.* It developed these tools in the task domain of interactive text editing and provided an extended empirical treatment of that domain. A brief review of these pieces will help fill out this original vision.

Model Human Processor. The Model Human Processor (Figure 1) is an approximate cognitive model of the user to be employed by the designer in thinking about the human interacting with the computer at the interface. The model is an attempt to refine the information processing diagrams that have been a staple of cognitive psychology—to make them into a calculational tool, rather than just a summary of the high-level structure of the cognitive system. It encapsulates the psychological literature in a simplified model of the human in terms of memories, processors, and a few quantitative parameters of each. The model permits some simple approximate calculations, such as how fast people can type. These calculations are done with ranges of parameter values as a way of taking into account the approximate nature of the model structure, parameters, and task analysis.

The GOMS Model. The GOMS family of models of the user (Figure 2) is an approximate way of characterizing user behavior in terms of *goals,* basic *operations* that the user could perform, *methods* for achieving the goals, and *selection* rules for choosing among alternative methods. The model allows a way of

Figure 1. The Model Human Processor. Depicted schematically in the figure are the memories, processors, and constants used for making simple computations: Sensory information flows into Working Memory through the Perceptual Processor. Working Memory consists of activated chunks in Long-term Memory. The basic principle of operation of the Model Human Processor is the Recognize-Act Cycle of the Cognitive Processor: On each cycle of the Cognitive Processor, the contents of Working Memory initiate actions associately linked to them in Long-Term Memory; these actions in turn modify the contents of Working Memory. The Motor Processor is set in motion through activation of chunks in Working Memory. Predictions are made using a set of associated *Principles of Operation:* (P0) The Recognize-Act Cycle of the Cognitive Processor, (P1) The Variable Perceptual Processor Rate Principle, (P2) The Encoding Specificity Principle, (P3) The Discrimination Principle, (P4) The Variable Cognitive Processor Rate Principle, (P5) Fitts's Law, (P6) The Power Law of Practice, (P7) The Psychological Uncertainty Principle, (P8) The Rationality Principle, and (P9) the Problem Space Principle. (Reprinted from Card, Moran, & Newell (1983, Fig. 2.1) with the permission of Lawrence Erlbaum Associates, Inc., Publishers.)

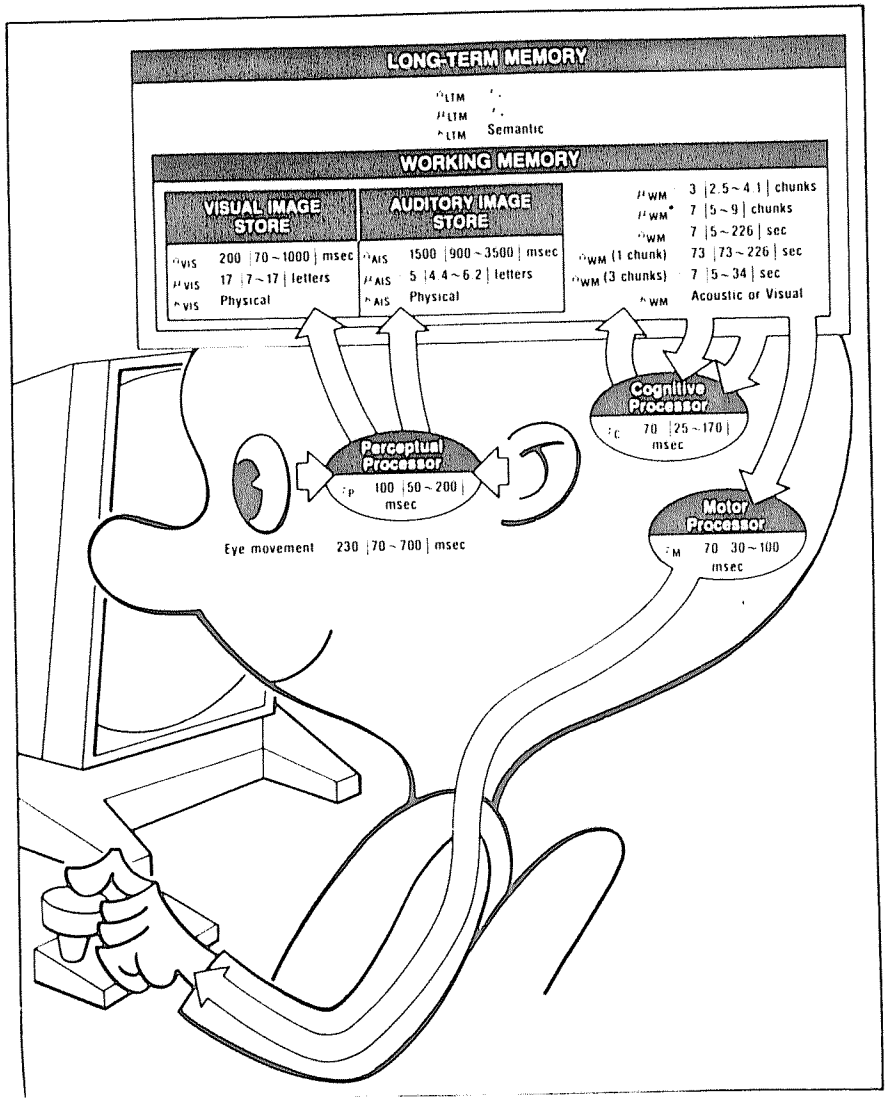


Figure 2. The GOMS (Goals, Operators, Methods, and Selection rules) Model. A portion of a GOMS model for the POET editor. The goals are GOAL:EDIT-MANUSCRIPT, GOAL:EDIT-UNIT-TASK, GOAL:ACQUIRE-UNIT-TASK, GOAL:EXECUTE-UNIT-TASK, and GOAL:MODIFY-TEXT. The operators are GET-NEXT-PAGE, GET-NEXT-TASK, and VERIFY-EDIT. The methods are named USE-QS-METHOD, USE-LF-METHOD, USE-S-COMMAND, and USE-M-COMMAND; in a more detailed model they would be expanded into more goals and operators. Selections are indicated by [select ...]. The actual selection rules themselves are not given in the figure. (Reprinted from Card, Moran, & Newell (1983, Fig. 5.12) with the permission of Lawrence Erlbaum Associates, Inc., Publishers.)

```
GOAL:EDIT-MANUSCRIPT
. GOAL:EDIT-UNIT-TASK                    repeat until no more unit tasks
. . GOAL:ACQUIRE-UNIT-TASK               if not remembered
. . . GET-NEXT-PAGE                      if at end of manuscript page
. . . GET-NEXT-TASK
. . GOAL:EXECUTE-UNIT-TASK               if an edit task was found
. . . [select USE-QS-METHOD
            USE-LF-METHOD]
. . GOAL:MODIFY-TEXT
. . . [select USE-S-COMMAND
            USE-M-COMMAND]
. . VERIFY-EDIT
```

recording possible users' methods so that users' behavior could be predicted. There is a family of such models, rather than a single one, because the level of aggregation can be pegged at different durations. The basic operators can be taken from the half-second level to the two-second level, depending on the type of analysis that needs to be done.

The Keystroke-Level Model. The Keystroke-Level Model (Figures 3 and 4) is a simplified, practical instance of the GOMS family. It is cast in terms of low-level operators (keystrokes, button pushes, and mouse moves), and it preserves only a single mental operator. Indeed, it is defined essentially as the approximation that follows if the number of mental operations is restricted to one.

PHCI focused on computer text editing, although several extensions were explored, such as page and circuit layout. Thus, the GOMS models were defined for text editing (as in the Keystroke-Level Model), although they arise from a general characterization of how skilled humans perform in task environments with discrete actions. The book was not purely theoretical, but contained empirical and experimental data on text editing, most of it seen through the lens of the theoretical models (e.g., the actual durations of operators in the GOMS models).

Figure 3. **The Keystroke-Level Model. The figure shows the analysis of a particular method of doing a particular task. The steps of the method are described in English in the left-hand column, then coded in terms of the Keystroke-Level Model on the right. The equation at the bottom of the figure is derived from this coding (e.g., t_M is the time to do an M operation). (After Card, Moran, & Newell, 1983, p. 289.)**

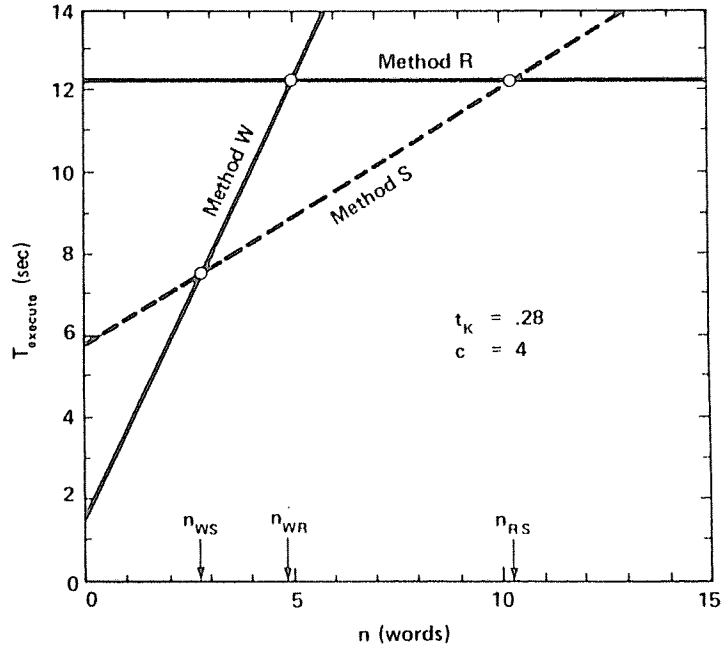
TASK: CORRECT BAD WORD *N* WORDS BACK
METHOD R: Point back to bad word, replace, resume typing

1. TERMINATE TYPE-IN	M K [ESC]
2. REPLACE BAD WORD	
2.1. Home hand on mouse	H[MOUSE]
2.2. Point to target word	P[word]
2.3. Select it	K[MOUSEBUTTON2]
2.4. Home hand on keyboard	H[KEYBOARD]
2.5. Invoke Replace command	M K[r]
2.6. Type new word	4.5K[word]
2.7. Terminate Replace	M K[ESC]
3. RESUME TYPING	
3.1. Home hand on mouse	H[MOUSE]
3.2. Point to last input word	P[word]
3.3. Select it	K[MOUSEBUTTON2]
4. RE-ENTER TYPE-IN MODE	
4.1. Home hand on keyboard	H[KEYBOARD]
4.2. Invoke Insert command	M K[i]

$$T_{\text{execute}} = 4T_M + 10.5T_K + 4T_H + 2T_P$$
$$= 12.1 \text{ sec}$$

In sum, *The Psychology of Human-Computer Interaction* attempted to provide a paradigmatic vision of the psychology of human-computer interaction and to instantiate that vision with studies to make it real. The book, of course, is only a symbol, convenient for its explicitness and because, naturally enough, it expresses the vision that we wish to convey. But this vision also exists in a number of heads. Anderson (1984), Robertson and Black (1983), Norman (1983; Norman & Draper, in press), Polson and Kieras (1985), and Young (1981), for example, have all enunciated the same general vision. At the level we wish to discuss it, variations of the vision are unimportant. Yet while this vision may be shared among a few, we do not believe it is common. Other visions still dominate. One dominating vision is the human-factors vision of human-

Figure 4. **An analysis result for text editing. This figure shows the result of a parametric analysis with the Keystroke-Level Model of the method of Figure 3 and two alternative methods for analyzing an editing task. The task is to delete a word *n* words back then resume typing. Each method is found to be faster than the others for some range of *n*. (Reprinted from Card, Moran, & Newell, 1983, Fig. 8.12a, with the permission of Lawrence Erlbaum Associates, Inc., Publishers.)**



computer interaction as primarily an evaluative, experimental field. A second, dominating vision, not directed at application per se but having a pronounced effect on it, is the psychological view of theory as a vehicle for explanation, not prediction. That is, the theory game is to prove that a theory is right, not to make useful, if approximate, calculations. These two dominating visions are both valuable; however, they are not sufficient to ensure major contributions of psychology to the field of human-computer interaction. They will never beat Gresham's law.

4. PROBLEMS OF REALIZING THE VISION

Although *The Psychology of Human-Computer Interaction* seems to have been fairly well received (Meyer, 1984; Park, 1983; Sebrechts, 1983; Swigger, 1983; Thomas, 1984), it is important to look at where this vision has so far failed or proved incomplete. In fact, there has been some explicit criticism; and these critics can help us to see the problems that exist with the vision. The

book remains a convenient symbol for the vision—making the criticism pointed and concrete—though no doubt confounded somewhat by other aspects. Shneiderman (1984) complained, in effect, that the book was too narrow, too incomplete, "The KLM [Keystroke-Level Model] applies the reductionist method to its limit, dismissing vital factors that, I believe, influence user performance" (p. 240). He complained that the book only looked at time as a measure and at editing as a task, which provided too narrow a basis. Allen and Scerbo (1983) complained that the Keystroke-Level Model, which represented the part closest to actual application, seriously underestimated the actual times. Thus, "while the Keystroke-Level Model makes a contribution in acknowledging and attempting to quantify predictions that include mental times, it seems that many additional insights into behavior and cognitive processes will be needed before a truly useful model of command-language use can be constructed" (p. 177).

These criticisms have their replies. But rather than debating the critics, let us instead add our own concerns to theirs. When we do this, we get the following short but pithy list of difficulties:

1. The science is too low level.
2. The scope of the science is too limited.
3. The science is too late.
4. The science is too difficult to apply.

Let us consider these in turn, retaining the book as a stand-in for the vision.

Too Low Level. The science provided is too low-level. *PHCI* emphasizes microscopic operations such as keystrokes and immediate commands for editing. But the real problems of the interface involve much more than this. They involve the organization of multiple tasks, such as editing, composition, and retrieval. They involve the principles on which the extended dialogue between the computer and the user occurs. They involve the design and construction of entire systems, not just the keyboard. They involve the learnability of the total system, the degree of compatibility between the system and the embedding context. Thus, however useful *PHCI* might be in dealing with the microstructure of the interface, it is not sufficient.

Too Limited. The scope of the phenomena and the contexts covered is too limited. There are many other aspects of the interface the vision does not address—the visual displays, the use of natural language, the problems of novice users, the questions of learning, the probability and effects of errors, the preferences of users, the effects of fatigue. The list is almost endless. Whatever success the book might have depends on it working with just the few aspects (duration of expert performance) that yield to such theoretical formulations.

Too Late. Science done this way comes too late for technological progress. By the time the research is done, the technology to which it applies is obsolete. The work of *PHCI* concentrated on line-at-a-time editors, yet by the time *PHCI* was published (1983), the display editor was already in wide use and multiwindow displays were becoming a substantial component in advanced environments. Thus, it is argued, theory is only useful for the questions that have been around long enough to be answered by experience anyway. On the currently important questions—where experience is not yet available—theory is silent.

Too Difficult to Apply. The gap to application is too big. The vision does not deal with real problems. Although *PHCI* may have the form of applicable theory, in fact few successful applications are discussed.

4.1. Prospects

In light of Gresham's law and the above difficulties with the vision, what are the prospects for the role of psychological science in the field of human-computer interaction? There seem to us just two possibilities.

The first, and most plausible, future is for psychology to have a relationship to human-computer interaction like that of classical human factors. The human-factors field has striven to find its place in the sun where it can make its contribution to better man-machine systems. The situation currently faced by psychology is the same in many respects—indeed, it can be viewed as nothing but the movement of human factors into a new and rapidly expanding technological domain. Thus, there seems little reason to project a different outcome.

Actually, the classical human-factors relationship is not really all that bad. They need you around. You get modest lip service and modest support. You can point with pride to a few places where you were taken into account. But the position has all the earmarks of second-class status. The important consequences, of course, are not those on the social status of a particular professional group, but those on the domain. Many aspects of the human use of interfaces will not be taken into account. Certainly the ever-continuing human-factors criticism of existing man-machine systems has a large measure of truth. The human dimension is often neglected and systems are much more difficult and less pleasant to use than they might be with proper attention and consideration.

The second future avoids continuation of the classical human-factors role by causing Gresham's law to become inapplicable. Unfortunately, the only way to cause Gresham's law to become inapplicable is to transform the psychology of the interface into a hard science. By that we mean producing engineering theories of the user—task analysis, calculation and approximation. Such theories need to be usable either directly or through the abstractions for

design thought that are their consequence. We see no other way of defeating Gresham's law. Hardening the science is certainly not the easy path. It counsels getting the human concerns into the interface by just producing better science. There is no cheap way out.

Success on this score would make psychological considerations full competitors for attention by the design team. Trade-offs, such as between the effort to learn a complex interface and the power of having it, could be understood enough to affect the types of interfaces explored. The design tasks for complex aspects, such as overlapping multi-windows, could be driven by an understanding of the psychological constraints and loads involved. Psychological aspects would not come to dominate considerations—design is inherently a trade-off among all the dimensions of cost and performance—but they would attain full considerations.

Several caveats must be noted, so that our position will not be misunderstood.

Experimentation is Needed. Although we emphasize engineering-style theory, we do not thereby deny that experimentation is required. We will say it explicitly: *Much experimentation is required.* This emphasis is necessary, because historically human factors have emphasized the need to get actual data on how humans actually use (and are prevented from using) specific machines. An emphasis on theory can easily appear to be a de-emphasis on experimentation. It is not. However, this caveat notwithstanding, experimentation in our view takes on a somewhat different role than in classical human-factors work. Its primary focus is on developing an engineering-level theory, rather than developing the end-user application.

The Vision is Limited to the Psychology of the Interface. Our topic is the psychology of the human-computer interface, not all of psychology. This distinction is important, because the proposal is not to make all of psychology better. The human-computer interface is, in fact, a psychologically limited microworld. Many issues of the wider world of psychology do not arise. From the point of view of this essay, one can be thankful for whatever limitations exist. They make the development of the science of the interface that much more feasible.

There are Other Interactions Between Humans and Computers. This proposal does not include the totality of relationships between humans and computers: programming, social psychology, organizational sociology, social impact. These are all important. They may be as important, or even more important, than the issues we are considering. We are addressing one part of a wider problem and not presuming to answer all problems. Divide-and-conquer on the issues is a key part of making progress feasible at all.

The Payoff for Design Must be Demonstrated. We take as obvious that the action is in design. Even so, it is incumbent to show how a hardened science—an effective calculational science—produces the desired effect on improving design. Conceivably, it could have little effect or no more effect than current experimental approaches. Conceivably, also, alternatives would be much better. For instance, the right thing to do might be to invent or propose design principles directly (Gould & Lewis, 1983; Nakatani & Rohrlich, 1983). Our argument is to the contrary; that if the science is there, it will be possible to package it into design principles, but that this cannot be successfully done the other way around.

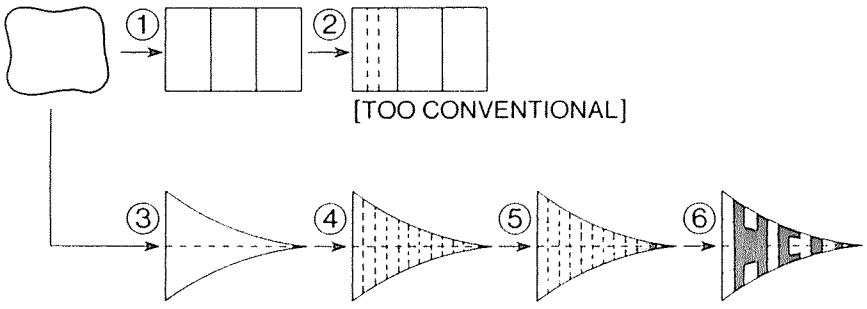
4.2. Psychology and the Designer's Tools for Thought

The link between theory and design is so important that it deserves additional consideration. Figure 5 is a highly simplified view of design. The designer starts with a key idea of what is wanted. The idea is embodied in some partial representation, with little detail. This representation, along with the goal for what must be included in the final design, defines a design space. The designer searches this space by successive refinements of the initial partial representation. At each step there is imaginative generation of additional aspects of the design and evaluation of the whole based on the partial information available.

This simple model of the design process describes how having a theory of the sort we have described makes a difference. An appropriate mathematical theory can provide the design representation itself. It can establish the framework for imagining the refinements and it can permit the evaluation of partial representations. For example, if we know that time for hand movement with the mouse and other analogue pointing devices follows Fitts's law (Fitts, 1954), then we begin thinking of devices in terms of their Fitts's law slopes. We begin thinking of selection targets on the screen in terms of Fitts's law scaling, leading us to make targets larger for longer mouse movements (Fitts's law tells by how much). *The way to get psychology into the inner design loop is to alter the abstractions, the representations for thought of the designer*—a kind of Whorfian hypothesis of design. This is the point of Norman's trade-off analysis (Norman, 1983a), in which he explicitly attempts to construct for the designer a representation for thinking about interactions between the psychological costs and benefits of different parts of the design.

Providing tools for thought is not the only way in which a theory can make a difference. Another way is through explicit computer program tools for the design. The theory is embodied in the tool itself, so that when the designer uses the tool, the effect of the theory comes through, whether he or she understands the theory or not. The tools may be of any kind—simulation, measurement, analysis, or system building. The Human Operator Simulator (Lane, Streib,

Figure 5. **The process of design. Hypothetical steps a designer might go through in designing a logo for human-computer interaction. The design begins as a nebulous idea and is refined in steps (numbered in the figure). From time to time the designer abandons a design path that is proving unsatisfactory and falls back several steps. The steps in the design are heavily dependent on the designer's representation.**



Glenn, & Wherry, 1981) provides an example. It is a system that has a human operator model embedded in a general-simulation system. The idea is that the analyst programs the task environment into the simulator, but the human part of the simulation is already fixed as part of the system. The system can be used for experiments with variations that are too expensive to individually prototype (Lane, Strieb, & Leyland, 1979). Although there are issues surrounding Human Operator Simulator on its ease of use, scope of applicability, and quality of embedded science, there is no question of the power of its central notion.

5. HARDENING THE SCIENCE

Hardening the science is, of course, a version of the original vision. As we have seen, this vision is not without problems. Still, our conclusion is that, if psychological science is to be a significant factor in human-computer interaction, then this vision must not be abandoned; rather, its problems must be overcome. It is thus appropriate to spend the remainder of this essay considering the prospects for overcoming the problem of level, the problem of scope, the problem of lateness, and the problem of remoteness from application cited above.

5.1. Overcoming Low-Level Science: Psychology Sets the Framework for Task Analysis

Let us start with the first problem of the vision, that the science is too low level to affect the design of real systems. Many of the problems of human-computer interaction are at the level of how to invent systems for activities like coordinated work, computer-aided design of VLSI chips, authoring of papers

and reports, or computer tutoring. Yet what seems relatively easy to obtain is psychological science on the legibility of displays, the arrangement of keyboards, the naming of commands, and the like. As an empirical observation, scientific psychology seems to traffic heavily at the low end of things, and to tread more lightly (pop psychology aside) at the high end. Yet the high-level tasks of authoring papers or tutoring surely involve humans as much as do the low-level tasks.

Is there something fundamental in this state of affairs? Or is it just that psychology is harder to do at higher levels, or that there is a natural time lag in developing the science from the bottom?

The Time Scales of Human Action. In fact, there does seem to be something basic about the different time scales on which human action occurs that leads to different classes of theories for actions of different duration. In Figure 6, a logarithmic time scale is associated with characteristic human actions having approximately the indicated duration. For example, at the duration of about a second, that is, roughly a third of a second to three seconds, the actions are those of basic psychological operations such as a choice reaction time or the addition of two digits. Associated with each action is a characteristic form of human memory where changes resulting from the action reside. For example, at the duration of about a second, the memory changes are those of short-term memory. Such a diagram has a certain cavalier and procrustean character. But in return for such crudity, we get to see a larger pattern. In particular, the diagram suggests that the theories used to describe human behavior fall into several bands.

1. *The domain of natural law.* The lowest band—below roughly 30 msec—is the domain of natural law. Physics, chemistry, and biology provide the scientific frameworks.
2. *The domain of psychology.* Starting just below a tenth of a second we enter the domain of psychology. Symbolic processing and mental mechanics dominate. Cycle times, operators, and unit tasks are the units of action; sensory buffers, short-term memory, and long-term memory are the units of memory change (using the terminology of the Model Human Processor). There are laws, but they differ radically in character from the laws that describe neural activity, being those of an information-processing architecture. The symbolic engine spans about a factor of 1000, from 30 msec to 30 sec.
3. *The domain of bounded rationality.* Above a minute, we enter a region in which human activity is described by giving the goals or ends being attempted. We talk about editing a manuscript, debugging a program, writing a letter, studying a chapter, or playing a game. These activities are elastic in time, lasting 10 minutes or a whole afternoon. The cognitive texture of such activities is described by the means-ends hierarchy of goals and subgoals, generated in re-

Figure 6. **The time scales of human action. Time duration of an action is scaled in powers of 10 for seconds (with the approximate unit equivalent indicated in parentheses). A characteristic action of this duration and an associated memory that provides inputs and outputs of the action is given for each time. Finally, the classes of theories that give an account of the actions are indicated. There seem to be four broad bands of theory, each at a different range of time. The actions associated with human-computer interaction fall into two of these bands: the psychological and the bounded-rationality bands.**

TIME (sec)	TIME (common units)	ACTION	MEMORY	THEORY
10^9	(decades)	Technology	Culture	
10^8	(years)	System	Development	Social and
10^7	(months)	Design	Education	Organizational
10^6	(weeks)	Task	Education	
10^5	(days)	Task	Skill	
10^4	(hours)	Task	Skill	Bounded
10^3	(ten mins)	Task	LTM	Rationality
10^2	(minutes)	Task	LTM	
10	(ten secs)	Unit task	LTM	
1	(secs)	Operator	STM	Psychological
10^{-1}	(tenths)	Cycle time	Buffers	
10^{-2}	(centisecs)	Signal	Integration	Neural and
10^{-3}	(millisecs)	Pulse	Summation	Biochemical

sponse to the details of the task. The theory that applies to this band is the rational calculation of means and ends by the user, deciding what to do in light of what the user knows about the task and its constraints in order to attain the user's goals. The user's calculations are of course limited or bounded (Simon, 1982), both by this knowledge and by the user's computational abilities.

4. *The domain of social and organizational theory.* As the time scale becomes longer—weeks, months, and years—we enter a world where the isolated actor is a rarity and where social theory comes to dominate. Although individuals continue to operate in intendedly rational ways, their social interactions play increasingly stronger roles, leading to statistical laws and aggregate effects. Different terms are used to describe the characteristic changes: education, development, and cultural change. The actions are organizational: design of a user-computer interface (months); development of a new interface type, for example, the personal computer (years); and the emergence of digital technology itself (decades).

Relationship Between the Psychological and Bounded-Rationality Bands. The time duration of human-computer interaction lies in both the psychology and bounded-rationality bands. In the psychology band are the time constants for the machinery of reasoning: the limits on rates of processing, encodings, working memory size, and reliability. Using these limited processing and memorial capabilities, the user performs the rational task analysis that characterizes the longer duration actions of the bounded-rationality band. This task analysis of the user done by the user in the service of the user's own goals is, of course, to be distinguished from the objective task analysis done by an observing scientist. In addition to basic psychological mechanisms, the user brings to the task situation a quota of acquired knowledge and skill. Indeed, the memory column of Figure 6 shows LTM and skill extending far into the bounded-rationality band into hours and days (and even further, though concealed by the changed terminology).

New psychological laws of information processing do not arise at longer durations. There are, instead, the accumulated effects of long-term memory and skill acquisition. These effects are influenced by the user's goal-oriented decisions about what to attend to and what to practice. Thus, new psychological phenomena occur as time increases, but their theoretical explanation is to be found in the interplay of the limited processing mechanisms of the psychological band and the user's intendedly rational endeavors. Of course, the paucity of specific cognitive mechanisms that emerge at durations above a minute or two is an empirical question. And it has to be said that there are general biological mechanisms at longer time scales, such as the sleep-wake cycle as well as psychological phenomena of boredom, fatigue, and mood belonging to the bounded-rationality region (although cognitive psychology has not yet been able to integrate these into its view of information processing). But the crucial point is that above a minute all theorizing about the human must adopt a framework of the user's rational endeavors on his own behalf.

Psychology and the Level of Human-Computer Interaction. Psychological science seems to be at too low a level, because the time scale of human-computer interaction falls across the time bands of two classes of theory, only one of which is primarily psychological. The lower band (tenths of seconds up to perhaps a minute) is that of cognitive psychology, of psychological laws and mechanisms as described in the Model Human Processor. This domain belongs to cognitive psychology. Psychological phenomena abound and cognitive psychologists are the scientists who ferret them out, describe them as data, and capture them in theory. However, this band is also the *low level* of the first criticism. The user interface, we are told, has bigger mental fish to fry.

The upper band (minutes up to days) is that of intendedly rational means-ends calculation. It is where the larger fish are to be found—the help systems, the intelligent extended discourses, and the workbenches for intellectual activity. This domain belongs to the task-domain specialist. The crucial ingredient is task analysis of the situation faced by the user, and discovery of what knowledge would help and what activities would support the attainment of the user's goals. The cognitive psychologist has no unique capabilities for such investigations of specific task domains—of CAD/CAM, sales support, or music composition.

Our conclusion, therefore, is that the criticism of the vision being too low level to cover the phenomena of the interface is partially correct. The domain that marks psychology's unique contribution is indeed limited to the band below a few minutes, which indeed is below the level of much that is critical in human-computer interaction. But that is not the whole story—for two reasons. First, the two bands are connected because the rational task analysis is being done by a human user of limited capabilities, as described by the lower cognitive model. *Psychology delivers to the bounded-rationality band the model that must be used to understand what helps the user and how.* The vision says that psychology should deliver that model in a useful form. Second, understanding the task domain is certainly the first requirement in the bounded-rationality domain; as the study of human-computer interaction improves, carrying out the analyses using a model of the human processor will become increasingly central. This is unclaimed intellectual territory and could well become part of an expanded psychology's contribution.

5.2. Overcoming Limited Scope: Fill Out the Model Human Processor

We now turn to the second criticism of the vision, that the scope of the phenomena and context covered by this vision are too limited. In fact, the scope of the vision has always been broader than the specific studies in *The Psychology of Human-Computer Interaction.* Those studies were the results of a particular research path. The prominence of performance-time measures, for instance, occurred because of the locus of the initial research successes. The larger vision was reflected in the Model Human Processor, which included among other things principles on problem solving and memory retrieval that went beyond what the particular studies could use. The inadequacies and incompleteness of that model can be taken as one part of an agenda for what needs to be done to increase the useful scope of psychological science for doing human-computer interaction. Again, the Model Human Processor is taken to mean a codification of results from the literature assembled into a calculational framework.

Is it feasible to increase the scope of the Model Human Processor? In fact, there appear to be a number of recent advances or promising lines of inquiry

that are ripe for integrating into this framework that could substantially increase its scope. We mention here a few of these as newly opened topics for research.

Integrated Models of Cognition. As shown in Figure 1, the Model Human Processor was an attempt to convert to operational form the descriptive block diagram of human mental processing that appears widely in articles and textbooks in cognitive psychology. It did this by adding operating principles to the model, which permitted some calculations to be made. Though it was not a complete model, it did have the virtue of putting a large array of psychological phenomena into a single framework and permitting calculations of some of these. This integration is critical. Perceptual, cognitive, and motor constraints all operate in unison to limit the user's performance. Learning, performance, and the development of skills all occur concurrently, although each with its characteristic rate. Thus, it is not enough to have theories of the separate aspects of cognition; a theory that permits them all to be taken into account in a single task is a real requirement.

Genuine progress has occurred on the overall structure of such an integrated model, with the development of the ACT* cognitive architecture (Anderson, 1983). ACT* is a production-system architecture that has evolved over the last decade until it provides the ability to simulate a wide range of human performance and learning. It covers not only the sort of chronometric performance tasks featured in the Model Human Processor, but acquisition and retrieval from long-term memory, the organization of long-term memory, and the acquisition of skills (see below). It is a more tightly integrated system than the Model Human Processor of *PHCI,* so that computer simulations can be run. For example, early stages of language acquisition have been simulated in ACT*. Equally important, Anderson has shown that it is also possible to make some approximate calculations about how ACT* would behave in a given task, so that it is not always necessary to run it in simulation mode to obtain useful answers. As we have repeatedly emphasized, this approximate calculational character is essential for any theory to have a massive impact on human-computer interaction. Thus, there is now an improved and broadened scientific base on which to build a Model Human Processor-style calculational framework.

Calculational Models of Perception. The Model Human Processor is especially deficient in the area of perception (so is ACT*). Given the radically graphic nature of modern interfaces, this is a serious shortcoming. Perceptual issues are important in both the psychological and the bounded-rationality time bands. In the psychological time band, perception is an inherently fast process (from the psychological view); many phenomena occur at the level of a tenth of a sec. Screen contrast, motion, aliasing, detection, and color effects

are all interface issues that occur at this level. Perception, of course, is a mature area of psychology with a solid experimental base. This base is being codified in a monumental *Handbook of Perception* (Boff, Kaufman, & Thomas, in press). The handbook reveals abundant theoretical fragments, many of them arising out of linear systems theory. There is renewed interest in placing this theory into the sort of integrated calculational framework necessary for its engineering use (Watson, 1983; Watson & Ahumada, 1985).

But major problems remain as the time duration increases. We currently do not have good ways of handling the use of graphic representation or screen space and relating these to the semantics and pragmatics of the user's task. Early studies are beginning, however, to make progress on the graphical representation in the interface (Bewley, Roberts, Schroit, & Verplank, 1983; Verplank, 1985), direct manipulation interface techniques (Shneiderman, 1983; Hutchins, Hollan, & Norman, 1985), and windows (Card, Pavel, & Farrell, 1984; Draper, in press; Reichman, in press). There is some developed theory of visual search (Krendel & Wodinsky, 1960; Engel, 1977; Card, 1984), but it has not yet been shaped into a useful tool and it has not been integrated into a large model. There is thus the potential for a Model Human Processor able to allow answers to much more sophisticated perceptual questions.

Performance and Error. Task-performance time models, of the GOMS and the Keystroke-Level Model sort, are being extended by the work of Polson and Kieras (1985) to combine performance and learning and by our own research on stimulus-response compatibility (John, Rosenbloom, & Newell, 1985). With respect to less routine behavior, there has long existed a well-developed theory of problem solving, which applies both to human cognition (Greeno, 1978; Newell & Simon, 1972) and to artificial intelligence (Rich, 1983). This theory is providing the basis for a number of developments, such as the planning models in the discourse work (Allen & Perrault, 1980; Cohen & Perrault, 1979). Interestingly, however, this theory has not yet been made to yield calculated predictions of a kind useful for designing the user interface.

With respect to errors, the situation is mixed. On the one hand, much error-evoked behavior appears to be variants on types of behavior already characterized in existing theory. Some of it is routine and hence GOMS-like. This type of behavior arises because errors themselves are not always of rare occurrence, so that users repeatedly make the same types of errors and develop appropriate methods to deal with them. Some error-evoked behavior is typical problem solving, both for diagnosing a situation and for recovering from it. On the other hand, the question of what errors occur and under what conditions is in need of much more work. Historically, work on errors has consisted of creating an ad hoc set of error-types and then tabulating error occurrences in the typology. But recently, Norman (1981, 1983, 1984) has developed a theoretically-based taxonomy of errors in terms of the cognitive mechanisms that cause them (e.g., distinguishing slips from mistakes). We can hope that this will lead to accelerated progress. Even a way of accumulating data on errors that has cross-situational validity would be extremely useful. Ultimately, error categories must be integrated into the theories that describe performance and learning. There is continuing need for good work here. Even in the area of performance models, there continues to be both progress and opportunities.

Knowledge Representation and Mental Models. The Model Human Processor was clear that users work in terms of internal symbolic representations of their tasks, but silent on the nature of these representations. Yet the user's representation of the task plays a critical role in determining the user's behavior. Within the last few years, however, enough research has been done to provide some basis for understanding a user's mental model of a system and for beginning to connect it with human-computer interface design (diSessa, 1985; Gentner & Stevens, 1983; Young, 1981). Halasz (1984) has provided evidence for the existence of users' models and their effects on performance, but has shown that the direct use of such models is only one of several sources of knowledge that users employ (e.g., they might remember the result of the previous time they ran the mental model, rather than running the model anew).

It is necessary to pass beyond describing users' models in existing user behavior to predicting the user's model given only a description of the task and its structure, plus the general state of knowledge and skill of the user. As everywhere else, engineering-style calculational theories must ultimately have symbolic theories that work on symbolic definitions of the task. Such theories have barely been attempted yet. Where they have, as in the intriguing analysis of hand calculators (Young, 1981), the connection to actual human behavior has remained somewhat tenuous. Work on users' mental representation is another promising way in which the scope of a Model Human Processor could be extended.

Acquisition of Cognitive Skill. Both the Model Human Processor and Figure 6 incorporate the fact that humans learn in two ways: by storing information in long-term memory and by acquiring skills. These ways of learning occur on strikingly different time scales, learning declarative knowledge (*knowing that*) being much faster than learning procedural knowledge (*knowing how*). Theories are needed for both types of learning. However, if anything, human-computer interaction depends more heavily on the latter. Learning to use a system—arriving at where performance can be described in GOMS-like terms—is consummately the acquisition of a cognitive skill.

A theory of cognitive skills acquisition has begun to emerge (Anderson, 1981) with many of the features needed for substantially expanding the scope of the Model Human Processor. Several independent efforts at model building in

different domains—learning Lisp and geometry (Anderson, 1982, 1983), instruction in subtraction (VanLehn, 1983), and expert systems (Rosenbloom, Laird, McDermott, Newell, & Orciuch, 1985)—have produced convergence on the ingredients. The learner is modeled as a production system (a set of rules of what actions will be taken under what conditions). Goals appear among the conditions of rules. Learning consists of adding a new production rule to the current set of rules. Skill is identified with the state in which, for the domain of the skill, goals are attached to the appropriate actions by a single rule (as opposed to rules that set up subgoals that, in turn, set up other subgoals). The construction of new rules cannot be done deliberately by the learner. New rules can only come about as automatic byproducts of the internal representations generated during attempts to achieve some goal (successes and failures both can yield relevant experience). Thus, what is required for the acquisition of procedural skill is some set of tasks for the learner to do that will, in turn, generate the internal intermediate information that will, in turn, generate new production rules through the automatic process. Any way of getting the learner to do tasks in the skill domain suffices—being told or instructed, following a given plan, problem solving, reading how, watching and imitating a successful solver, doing exercises, taking hints, using an analogy, and so on.

Such a theory of cognitive skill has important potential for widening the scope of a Model Human Processor. It is compatible in assumptions and mechanisms with the larger cognitive model we have been discussing. It focuses on exactly the level of greatest concern to human-computer interaction, epitomized in learning a computer language such as Lisp. It appears to be general enough to cover much procedural skill acquisition. What does need further development is how to work with this theory in a calculational mode. Such a theory can neither predict transfer effects from a symbolic description of the systems to be learned nor the constants in the approximate learning equations that can be derived from the theory. But the theory specifies the mechanisms which seem to determine these phenomena.

Partly Linear Models. A common structure appears repeatedly in these theories, which is worth pointing out because it shows one path toward building appropriate models. A given phenomenon of interest can be conceptualized as consisting of two components:

Phenomenon = Volume-part + Difficulty-part.

Something in the task occurs in volume and the user must deal with it by repeating some operation a large number of times. This generates a linear part—linear in time, errors, solution opportunities, or whatever. However, the phenomenon also has an equally strong (or stronger) nonlinear component, which often reflects some source of difficulty. Thus, the standard scientific strategy of approximating a phenomenon by its linear term cannot work unadorned.

However, in the human-computer situation it is often possible to discriminate the nonlinear component, so it can be handled separately. If the linear part can be brought under good theoretical control, it can be exploited—both for its own sake and for teasing out the nonlinear part. Thus, we can think of such theories as *partly linear models*, an analogy to *almost linear* or *quasi-linear* models.

An example is reading for comprehension. In this case, the volume part of the equation is the amount of text to be read. By taking the text item as a unit, reading becomes a linear process with a simple rate constant (e.g., 450 words per minute for light prose, 250 for technical prose). The difficult part is caused by a variety of integration and understanding sub-phenomena. There are many interesting cognitive models of the reading process (Kieras & Just, 1984) that show this dual structure—a linear part with additional constants to account for local fluctuations of reading speed, caused by sentence endings, anaphoric references, new words, etc.

The GOMS and Keystroke-Level Models provide another instructive case. The linear (volume) part is obvious—the number of operations multiplied by the average time per operator. The first interesting aspect is that the numbers of each type of operation can only be determined when the method employed by the user for doing the task is known. Thus, anlayses with these models naturally divide into two activities: determining the method and determining performance time given the method. This decomposition works because the method can be determined independently by task analysis. Also, it is significantly easier to do the task analysis of expert users than of novice users. So, initially, GOMS-like models deal with expert behavior, although there is no reason not to extend them toward novice behavior. The second interesting aspect concerns the difficulty part—the error behavior, which in text editing can be nontrivial (10% to 25%). Errors require a distinct theory, so they are split off and the GOMS and Keystroke-Level Models are cast as error-free expert behavior. This splitting works because error-free expert behavior is a useful category in its own right, for example, dealing with questions such as, "If the user were to use the proposed method, how much time would the task take?"

There are more examples. In learning declarative material, for instance, the volume part of the equation is the amount of material to learn, but as we know from much work in psychology, the unit of learning is not the textual item, but the *chunk* (Miller, 1956). If the user already has the material partly organized in chunks, then only the remaining chunks must be learned. The difficulty part appears to be predominantly due to interference. In the work of Polson and Kieras (1985), production systems provide an alternative formalism to the GOMS notation of *PHCI* and extend it to procedural learning. This list is enough to show that partly linear models arise frequently in human-computer interaction, and that they provide one underlying form for much of the calculational theory in the Model Human Processor. This form provides some guidance in asking how to build appropriate calculative theories for other aspects.

5.3. Overcoming Late Science: Cumulate the Science of the Canonical Interface

The third problem with the vision concerned the timing of science for making progress in a rapidly moving technology. By the time years of study with teletype-style interfaces reached maturity, for example, the technology had passed on to bitmapped displays in which the old problems were no longer of interest. By the time we understand bitmapped displays, the technology will have passed on to more exotic displays. And so on.

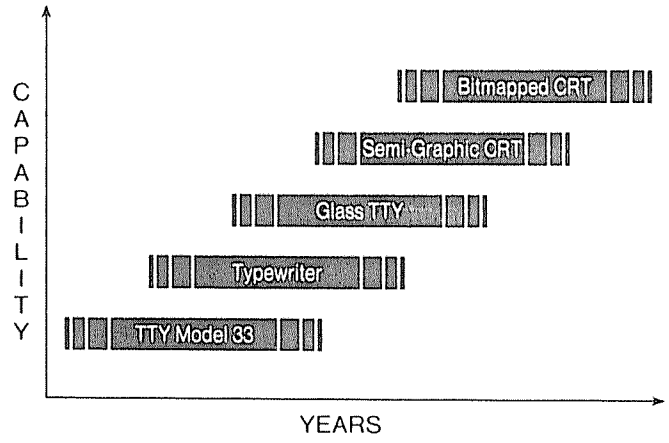
The traditional answer is that science cumulates, so that it is possible gradually to succeed. Ultimately the tools will already be in place to analyze the onrushing new technology. The path we have called for—to construct an engineering-style theory of the user—is solidly in this tradition. Though the computer is an arena where technology seems to be moving ever faster while the basic science crawls along, this is only cause to hold firm and redouble the effort. This traditional answer has to be correct.

However, there is another partial answer—or at least a little more to be said. Interface hardware is actually highly standardized across the industry. At any point in time only a few interface devices are in general use (Figure 7). Thus, we have had a progression from 10 characters/sec uppercase teletypes to typewriter-like interfaces to CRT-based glass teletypes to high-resolution multiwindow displays. (Actually the progression starts even earlier with card readers and printers.) Furthermore, other aspects of the interface correlate with the basic terminal type: interaction rates, computations per interaction cycle, display size in pixels, display area, media, memory available for the interaction, memory by the system of the user, number of computational agents working concurrently, and degree of intelligence of the interface programs.

Approximately every 5 years a new *canonical interface* (to give it a name) comes to dominate the scene. Over the years this interface passes through a typical life cycle: starting out developmental, expensive and limited to research environments, then rising to prominence, then to mature dominance, and finally undergoing a slow phase out as successive canonical interfaces take over. Throughout the life cycle, parameters gradually improve, but the gestalt of the interface does not change. Most change occurs in price, reliability, and physical packaging, which affect market penetration enormously, but not the user characteristics. Fifteen to 20 years is a reasonable figure for the total lifetime of an interface type. (If this seems long, note that bitmapped displays are already a decade old, and they have yet to become fully dominant.) As a result, a mixture of perhaps three canonical interfaces share the scene at any moment, each in a different phase of its life cycle.

Most tasks to be done on a computer go through the interface—therefore, they go through the current canonical interface. Thus, there is a specialized, standardized environment, which is the locus of human-computer interaction.

Figure 7. **The canonical interface. Most human-computer interaction takes place by means of only a few devices that are in mass use. The features of the members of each device class tend to be similar (e.g., all typewriter terminals tend to have speeds in a similar range). As years go by, the number of users with various types of equipment slowly changes. This limited and correlated variability in equipment simplifies the analysis of human-computer interaction by imposing similar constraints on different interfaces.**



Correlated features:

- Terminal device type
- Rates of interaction
- Computation per interactive cycle
- Amount of displayed information
- Media: Text, Box graphics, Full graphics, ...
- Memory of the system for the user
- Number of active agents
- Amount of intelligence

To all intents and purposes, then, the psychology of human-computer interaction is the psychology of interaction with the canonical interface. This is a remarkable situation. It sharply distinguishes the psychology of human-computer interaction from general human factors, which must deal with an immensely more variable operating environment, and hence has a much harder task. The canonical interface is the feature, if any, that might make possible a separate discipline of human-computer interaction. Because changes in the canonical interface take on the order of years, this appears to give us enough time, though perhaps just barely, to track the changes scientifically, gradually updating and improving the Model Human Processor. The interface is so specialized compared with the range of possible tasks people can do that this specialization gives us real leverage.

5.4. Overcoming Difficulty of Application: Find Ripe Application Domains

Finally, let us turn to the fourth and last criticism of the vision—that the gap between theory, on one hand, and real applications, on the other, is too large. The fundamental answer is that one needs to keep trying. When faults are discovered in proposed approximative models, we should build improved ones (as opposed to saying models are premature and throwing the whole idea out). In a favorite phrase of McCulloch (1965), "Don't bite my finger, look where I am pointing" (p. *xx*). Even a relatively poor approximation may be of significant use; the utility of powers of ten calculations in physics is an example. But improvement of these models requires their exercise by many people in different circumstances. Studies that have applied, tested, improved, or extended the approximative models of the book seem to us more on the right track (Robertson, McCracken, & Newell, 1980; Allen & Scerbo, 1983; Bewley et al., 1983; Gomez, Egan, Wheeler, Sharma, & Gruchacz, 1983; Poller & Garter, 1983; Robertson & Black, 1983; Ross & Moran, 1983; Borenstein, 1985; Good, 1985; Whiteside, Jones, Levy, & Wixon, 1985).

A second answer is to look for good application domains. The important property to seek in applications is that they are ripe for advancing science—that they will force the development of a Model Human Processor in the context of applications. Application domains are not all equally fertile and hospitable. And transferring results from one successful domain to a new one is infinitely easier than being successful the first time.

One interesting domain is intelligent tutoring systems (Anderson, Boyle, & Reiser, 1985; Sleeman & Brown, 1982). These are systems capable of tutoring a student in a specific (currently very narrow) subject matter. It presents the student with material to be learned and with exercises to be done, it analyzes the student's performance on the exercises, it provides guidance and help for the exercises, and it chooses new material and exercises tailored to the student's current state of learning. Such tutors are artificial intelligence systems; they must understand the subject matter well enough to solve the problems themselves and to analyze the student's often fumbling attempts at solving the problem. But such systems must also build up and maintain models of the student; they must infer from the student's behavior not only what he knows and what he will do, but also predict what instructional steps will help and what will hinder further learning. They must, in other words, be rather sophisticated interfaces. Although there are still not large numbers of intelligent tutoring systems, the subject and educational levels range fairly broadly, from children's subtraction problems (Burton, 1982) to Lisp (Anderson & Reiser, 1985).

Viewed one way, intelligent tutoring systems seem to be just another application area, though perhaps interesting because applying computers to education has been contentious for decades or because it intertwines artificial intelligence and human-computer interaction. From our perspective, however, the application is special. It just might be the path to major advances in hardening the psychology of the interface. The critical feature is the required inclusion of a psychological model of the user (the student). The tutor itself must use the model to predict student behavior—the very paradigm for using the Model Human Processor. It actually goes the standard paradigm one better: that a program, rather than a human psychologist, must make the evaluations and predictions, insures objective calculation and task analysis. (That the student models will also be approximate needs hardly be said.)

Thus, intelligent tutoring might be the area in which the Model Human Processor finally develops and succeeds enough to demonstrate to us all that a hard cognitive psychology of the interface is feasible. Once we see this clearly in one area, developing it in other areas will go more easily. Two additional factors provide cause for optimism. First, the intelligent tutor interface is a rich one from the point of view of the Model Human Processor, because it combines all the usual interaction phenomena with all the additional phenomena of learning. Second, cognitive science in education is very active currently, so there is much new science to draw upon. Indeed, the tutors that have been developed are strongly and explicitly linked to frontier work in cognitive science (Anderson, Boyle, Farrell, & Reiser, 1984; VanLehn, 1983).

6. CONCLUSION

Let us now state the main conclusions to our argument.

Gresham's Law: Hard Science Drives Out Soft. This principle is our task analysis of the main constraints that will determine the future role of psychological science in human-computer interaction. Our concern here has not been to cheer for psychological science or other soft science for its own sake or for professional reasons. We are unconcerned with whether psychology departments prosper or psychologists have adequate employment or whether they are influential or well-treated on interdisciplinary teams. Our sole concern is with the quality of future interfaces. We think psychological science has major potential contributions to make, but that, as soft science (compared to computer science) its actual influence could well be marginal. The only chance we see of preventing this is to *harden the science.*

There is Nothing so Useful as a Good Theory. This is an old cliché, but it serves our purpose. Striving to develop a theory that does *task analysis* by *calculation* is the key to hardening the science. We tried in *The Psychology of Human-Computer Interaction* to provide a glimmer of what might be achieved in this regard. We see additional progress in the indications that calculations are possible in Anderson's ACT* theory. The notion of *approximation* is important to keep from setting the initial sights too high.

Good Studies of the Interface Yield Theories, Not Facts. There is a strong role for empirical studies of the interface, but it is not to evaluate proposed interfaces, but to provide the basis for the theories. Most applications to the interface should be made by applying theory, not by doing special experiments. (Of course, some of the latter are useful as well.)

Psychological Research Best Affects Design by Providing the Designer Tools for Thought. Psychological theories and experiments, such as Fitts's index of difficulty or Norman's framework of trade-off analysis, can shape the way a designer thinks about a problem. Analyses of the key constraints of a problem can point the way to fertile parts of the design space. Providing tools for thought is a more effective way of getting human engineering into the interface than running experimental comparisons between alternative designs.

The Race is Between the Tortoise of Cumulative Science and the Hare of Intuitive Design. Intuitive experimental programming has been the basis for most interface innovations because it is a cream-skimming technique. In an environment of rapidly changing technology, experimental programming is the way to get there the fastest with the mostest. From time to time this intuitive approach to design gets into difficult problems, which then provide payoff for more systematic scientific investigations. Eventually the science, if it cumulates, acquires sufficient abstractions and theory to suggest new technology directions and to develop substantial leverage over intuitive design. In technology development, designs are the coin of the realm. For a science to play a significant role in technology, it must be the means by which important pieces of the design come to be. The reverse side of the coin is that this practical pressure of technology can also be of benefit to psychological science itself by providing a sustained, concrete set of problems to solve with visibly measurable results. Whether shipboard chronometers for maritime navigation or human interfaces for complex systems, nothing drives basic science better than a good applied problem.

Acknowledgments. We are grateful to Jill Fain, Bonnie John, Tom Moran, and Richard Young for comments on early drafts of this paper.

REFERENCES

Aho, A., & Ullman, J. D. (1972). *The theory of parsing, translating, and compiling, Vols. 1 & 2.* Englewood Cliffs, NJ: Prentice-Hall.

Allen, J. F., & Perrault, C. R. (1980). Analyzing intention in dialogues. *Artificial Intelligence, 15,* 143–178.

Allen, R. B., & Scerbo, M. W. (1983). Details of command-language keystrokes. *ACM Transactions on Office Information Systems, 1,* 159–178.

Anderson, J. R. (1980). *Cognitive psychology and its implications.* New York: Freeman.

Anderson, J. R. (Ed.). (1981). *Cognitive skills and their acquisition.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Anderson, J. R. (1982). The acquisition of cognitive skill. *Psychological Review, 89,* 369–406.

Anderson, J. R. (1983). *The architecture of cognition.* Cambridge, MA: Harvard University Press.

Anderson, J. R., & Reiser, B. J. (1985). The LISP tutor. *Byte, 10*(4), 159–175.

Anderson, J. R., Boyle, C. F., & Reiser, B. J. (1985). Intelligent tutoring systems. *Science, 228,* 456–467.

Anderson, J. R., Boyle, C. F., Farrell, R., & Reiser, B. J. (1984). Cognitive principles in the design of computer tutors. *Proceedings of the Sixth Annual Cognitive Science Meetings.* Cognitive Science Society.

Bewley, W. L., Roberts, T. L., Schroit, D., & Verplank, W. L. (1983). Human factors testing in the design of Xerox's 8010 'Star' office workstation. *Proceedings of the Conference on Human Factors in Computer Systems (CHI '83)* (pp. 72–77). New York: ACM.

Boff, K., Kaufman, L., & Thomas, J. (Eds.). (in press). *Handbook of perception and human performance, Vols. 1–3.* New York: Wiley.

Borenstein, N. S. (1985). The evaluation of text editors: A critical review of the Roberts and Moran methodology based on new experiments. *Proceedings of CHI '85* (pp. 99–105). New York: ACM.

Burton, R. R. (1982). Diagnosing bugs in a simple procedural skill. In D. Sleeman & J. S. Brown (Eds.), *Intelligent tutoring systems* (pp. 157–183). New York: Academic.

Card, S. (1984). Visual search of computer command menus. In H. Bouma & D. G. Bouwhuis (Eds.), *Attention and performance, X* (pp. 97–108). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Card, S., & Newell, A. (1984). The psychology of human-computer interaction, three lectures. In M. J. Elphick (Ed.), *Proceedings of the Joint IBM/University of Newcastle upon Tyne Seminar.* Newcastle upon Tyne, Great Britain: Department of Computer Science, University of Newcastle upon Tyne.

Card, S., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Card, S., Pavel, M., & Farrell, J. E. (1984). Window-based computer dialogues. *Proceedings of IFIP Interact '84,* 355–359. London: IFIP.

Cohen, P. R., & Perrault, C. R. (1979). Elements of a plan-based theory of speech acts. *Cognitive Science, 3,* 177–212.

diSessa, A. A. (1985). A principled design for an integrated computational environment. *Human-Computer Interaction, 1,* 1–47.

Draper, S. W. (in press). Display managers and the basis for user-machine communication. In D. A. Norman & S. W. Draper, (Eds.), *User-centered system design: New perspectives on human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Engel, F. L. (1977). Visual conspicuity, visual search, and fixation tendencies of the eye. *Vision Research, 17,* 95–108.

Fitts, P. M. (1954). The information capacity of the human motor system in controlling amplitude of movement. *Journal of Experimental Psychology, 47,* 381–391.

Gentner, D., & Stevens, A. L. (Eds.). (1983). *Mental models.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Gomez, L. M., Egan, D. E., Wheeler, E. A., Sharma, D. K., & Gruchacz, A. M. (1983). How interface design determines who has difficulty learning to use a text editor. *Proceedings of the Conference on Human Factors in Computer Systems (CHI '83)* (pp. 176–180). New York: ACM.

Good, M. (1985). The use of logging data in the design of a new text editor. *Proceedings of the Conference on Human Factors in Computer Systems (CHI '85)* (pp. 24–27). New York: ACM.

Gould, J. D., & Lewis, C. (1983). Designing for usability—Key principles and what designers think. *Proceedings of the Conference on Human Factors in Computer Systems (CHI '83)* (pp. 50–53). New York: ACM.

Greeno, J. (1978). The nature of problem-solving abilities. In W. K. Estes (Ed.), *Handbook of learning and cognitive processes, Vol. 5* (pp. 239–270). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Halasz, F. (1984). *Mental models and problem solving in using a calculator.* Unpublished doctoral dissertation, Stanford University, Stanford, CA.

Hansen, W. J. (1971). User engineering principles for interactive systems. *Proceedings of the Fall Joint Computer Conference, 39,* 523–532. New York: AFIPS.

Heckel, P. (1984). *The elements of friendly software design.* New York: Warner Books.

Hutchins, E. L., Hollan, J. D., & Norman, D. A. (in press). Direct manipulation interfaces. In D. A. Norman & S. W. Draper, (Eds.), *User-centered system design: New perspectives on human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

John, B. E., Rosenbloom, P. S., & Newell, A. (1985). A theory of stimulus-response compatibility applied to human-computer interaction. *Proceedings of the Conference on Human Factors in Computer Systems (CHI '85)* (pp. 213–219). New York: ACM.

Kieras, D. E., & Just, M. A. (Eds.). (1984). *New methods in reading comprehension research.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Krendel, E. W., & Wodinsky, J. (1960). Search in an unstructured visual field. *Journal of the Optical Society of America, 50,* 562–568.

Lane, N. E., Strieb, M. I., Glenn, F. A., & Wherry, R. J. (1979). The human operator simulator: An overview. In J. Moraal & K. F. Kraiss (Eds.), *Manned systems design: Methods, equipment and applications. NATO conference series, series III: Human factors, Vol. 17* (pp. 121–152). New York: Plenum.

Lane, N. E., Strieb, M. I., & Leyland, W. (1980). Modeling the human operator: Applications to system cost effectiveness. *AGARD Conference Proceedings CP-268, Modeling and Simulation of Avionics Systems and Command, Control and Communications Systems,* (pp. 39-1 to 39-10). NATO.

McCulloch, W. S. (1965). *Embodiments of mind.* Cambridge, MA: MIT Press.

Meyer, B. (1984). The human factor has knocked three times: Three books on the ergonomics of computing systems. *Technology and Science of Informatics, 3*(3).

Miller, G. A. (1956). The magic number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review, 63,* 81–97.

Muckler, F. (1984). The future of human factors. *Human Factors Society Bulletin, 27*(2), 1.

Nakatani, L. H., & Rohrlich, J. A. (1983). Soft machines: A philosophy of user-computer interface design. *Proceedings of the Conference on Human Factors in Computer Systems (CHI '83).* New York: ACM.

Newell, A., & Simon, H. A. (1972). *Human problem solving.* Englewood Cliffs, NJ: Prentice-Hall.

Norman, D. A. (1981). Categorization of action slips. *Psychological Review, 88,* 1–15.

Norman, D. A. (1983). Design principles for human-computer interfaces. *Proceedings of the Conference on Human Factors in Computer Systems (CHI '83)* (pp. 1–10). New York: ACM.

Norman, D. A. (1983). Design rules based on analyses of human error. *Communications of the ACM, 26,* 254–258.

Norman, D. A. (1984). *Working papers on errors and error detection.* San Diego, CA: Institute for Cognitive Science, University of California at San Diego.

Norman, D. A. & Draper, S. W. (Eds.). (in press). *User centered system design: New perspectives on human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Park, D. K. (1983). Review: *The Psychology of Human-Computer Interaction. Human Factors Society Bulletin, 26*(10), 3.

Poller, M. F., & Garter, S. K. (1983). A comparative study of moded and modeless text editing by experienced editor users. *Proceedings of the Conference on Human Factors in Computer Systems (CHI '83)* (pp. 166–170). New York: ACM.

Polson, P. G., & Kieras, D. E. (1985). A quantitative model of the learning and performance of text editing. *Proceedings of the Conference on Human Factors in Computer Systems (CHI '85)* (pp. 207–212). New York: ACM.

Reichman, R. (in press). Communication paradigms for window systems. In D. A. Norman & S. W. Draper (Eds.), *User centered system design: New perspectives on human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Rich, E. (1983). *Artificial Intelligence.* New York: McGraw-Hill.

Robertson, S. P., & Black, J. B. (1983). Planning units in text editing behavior. *Proceedings of the Conference on Human Factors in Computer Systems (CHI '83)* (pp. 217–221). New York: ACM.

Robertson, G., McCracken, D., & Newell, A. (1980). The ZOG approach to man-machine communication. *International Journal of Man-Machine Studies, 14,* 461–488.

Rosenbloom, P. S., Laird, J. E., McDermott, J., Newell, A., & Orciuch, E. (1985). R1-Soar: An experiment in knowledge-intensive programming in a problem solving architecture. *IEEE Conference on Pattern Analysis and Machine Intelligence 7*(5), 561–569.

Ross, B. H., & Moran, T. P. (1983). Remindings and their effects in learning a text editor. *Proceedings of the Conference on Human Factors in Computer Systems (CHI '83)* (pp. 222–225). New York: ACM.

Sebrechts, M. A. (1983). Review: *The Psychology of Human-Computer Interaction. American Scientist, 71,* 656.

Shneiderman, B. (1980). *Software psychology: Human factors in computer and information systems.* Cambridge, MA: Winthrop.

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

Shneiderman, B. (1984, January). Review: *The Psychology of Human-Computer Interaction. Datamation, 30,* 236–237.

Simon, H. A. (1982). *Models of bounded rationality, Vol. 2.* Cambridge, MA: MIT Press.

Sleeman, D., & Brown, J. S. (Eds.). (1982). *Intelligent tutoring systems.* New York: Academic.

Smith, S. L., & Mosier, J. W. (1984). *Design guidelines for user-system interface software* (Tech. Rep. No. MTR-9420). Bedford, MA: MITRE Corporation.

Swigger, K. M. (1983, November). Review: *The Psychology of Human-Computer Interaction. Computing Reviews, 24*(11), 471–472.

Thomas, J. (1984). Goodness (Human Factors) ≠ Degree (Quantification). *Contemporary Psychology, 29,* 119–120.

VanLehn, K. (1983). *Felicity conditions for human skill acquisition: Validating an AI-based theory* (Tech. Rep. No. CIS-21). Palo Alto, CA: Xerox Palo Alto Research Center.

Verplank, W. (in press). Principles of graphical interface design. In H. Peterson & W. Schneider (Eds.), *Proceedings of the Human-Computer Communications on Health Care.* North Holland, Amsterdam: International Medical Informatics Association.

Watson, A. B. (1983). Detection and recognition of simple spatial forms. In O. J. Braddick & A. C. Sleigh (Eds.), *Physical and biological processing of images* (pp. 110–114). New York: Springer-Verlag.

Watson, A. B., & Ahumada, A. J. (1985). Model of human visual motion sensing. *Journal of the Optical Society of America, 2,* 322–342.

Whiteside, J., Jones, S., Levy, P. S., & Wixon, D. (1985). User performance with command, menu, and iconic interfaces. *Proceedings of the Conference on Human Factors in Computer Systems (CHI '85)* (pp. 185–191). New York: ACM.

Young, R. (1981). The machine inside the machine: User's models of pocket calculators. *International Journal of Man-Machine Systems, 15,* 51–86.

HCI Editorial Record. This is an invited paper. First manuscript received June 4, 1985. Revision received August 5, 1985. Final manuscript received August 28, 1985. — *Editor*