Cognitive Modeling and Human–Computer Interaction

Wayne D Gray
ARCH Laboratory

Erik M. Altmann
ARCH Laboratory

Laboratory for Applied Research in Cognition and Human Factors
Department of Psychology
George Mason University
Fairfax, Virginia

1 The writing of this article was supported by grants from the Office of Naval Research (#N00014-95-1-0175) and the Air Force Office of Scientific Research (AFOSR#F49620-97-1-0353).
Correspondence concerning this article should be addressed to Wayne D. Gray, George Mason University, msn 3f5, Fairfax, VA 22030. gray@gmu.edu
COGNITIVE MODELING AND HUMAN-COMPUTER INTERACTION

“There is nothing so useful as a good theory” (Lewin, 1951).

“Nothing drives basic science better than a good applied problem” (Newell & Card, 1985).

1 Introduction

The quotations from Lewin and from Newell and Card capture what motivates those who apply cognitive modeling to human-computer interaction (HCI). Cognitive modeling springs from cognitive science. It is both a research tool for theory building and an engineering tool for applying theory. To the extent that the theories are sound and powerful, cognitive modeling can aid HCI in the design and evaluation of interface alternatives. To the extent that the problems posed by HCI are difficult to model or cannot be modeled, HCI has served to pinpoint gaps or inconsistencies in cognitive theory. In common with design, science is an iterative process. The symbiotic relationship between modeling and HCI furthers the scientific enterprise of cognitive science and the engineering enterprise of human factors.

Cognitive modeling is a form of task analysis and, as such, is congenial to many areas and aspects of human factors. However, the control provided by the computer environment, in which most dimensions of behavior can be easily and accurately measured, has made HCI the modeler’s primary target. As modeling techniques become more powerful and as computers become more ubiquitous, cognitive modeling will spread into other areas of human factors.

We begin this article by discussing three cognitive models of HCI tasks, focusing on what the models tell us about the tasks rather than on the details of the models themselves. We next examine how these models, as well as cognitive models in general, integrate constraints from the cognitive system, from the artifact that the operator uses to do the task, and from the task itself.
We then explore what sets cognitive modeling apart from other types of cognitive task analysis and examine dimensions on which cognitive models differ. We conclude with a brief summary.

2 Three Examples of Cognitive Modeling Applied to HCI

The three examples span the gamut of how models are used in HCI. We discuss them in the order of most applied to most theoretical. However, it would be a mistake to think of these as application versus research as each has contributed strongly to theory and each has clear applications to HCI issues.

2.1 Project Ernestine: CPM-GOMS

In the world of the Telephone Company, time is literally money. In the late 80’s, NYNEX calculated that if the length of each operator-assisted call decreased by 1 sec the company’s operating costs would be reduced by $3 million per year. Potential savings on this scale provided an incentive to shave seconds from the time that toll and assistance operators (TAOs) spent on operator assisted calls.

A major telecommunications equipment manufacturer promised to do just that. For an equipment investment of $60 to $80 million, the old TAO workstations could be replaced by new, ergonomically engineered workstations. The manufacturer’s back-of-the-envelope style calculations predicted that the new workstations would shave about 4 s from the average call for an estimated savings of $12 million annually.

Project Ernestine involved a combination of cognitive modeling and field trial to compare the new workstations with the old (Gray, John, & Atwood, 1993; Gray, John, Stuart, Lawrence, & Atwood, 1995). The cognitive models created in Project Ernestine used the GOMS task analysis technique developed by Card, Moran, and Newell (1983). GOMS analyzes a task in terms of Goals, simple Operators used by the person performing the task, and sequences of operators that
form Methods for accomplishing a goal. If alternative methods exist for accomplishing a goal, then a Selection rule is required to choose among them. GOMS is best suited to the analysis of routine, skilled performance, as opposed to problem solving. The power of GOMS derives in part from the fine-grain level of detail at which it specifies the operators involved in such performance. (For a fuller exposition on GOMS, see John & Kieras, 1996a; John & Kieras, 1996b.)

Project Ernestine employed a GOMS variant, CPM-GOMS, to analyze the TAO’s task. CPM-GOMS specifies the parallelism and timing of elementary cognitive, perceptual, and motor operators, using a schedule chart format that enables use of the critical path method to analyze dependencies between these operators.

Contrary to expectations, the cognitive models predicted that the new workstations would add about 1 s to the average call. Rather than reducing costs as predicted by the manufacturer, this increased time would result in $3 million in additional operating costs. This prediction was borne out empirically by a 4-month field study using live telephone traffic. A sample of the CPM-GOMS model for the beginning part of one call type is shown in Figure 1.

Beyond its prediction, CPM-GOMS was able to provide explanation. The manufacturer had shown that the proposed workstation reduced the number of keystrokes required to process a typical call and from this inferred that the new workstation would be faster. However, their analysis ignored the context of the call, namely the interaction of customer, workstation, and TAO. CPM-GOMS captured this context in the form of a critical path of cognitive, perceptual, and motor actions required for a typical call. By filling in the missing context, CPM-GOMS showed that the proposed workstation added more steps to the critical path than it eliminated.
This qualitative explanation made the model’s prediction credible to telephone company executives.

---

2.2 Postcompletion Error

An adequate theory of error “is one that enables us to forecast both the conditions under which an error will occur, and the particular form that it will take” (Reason, 1990, p. 4). Such a theory was developed by Byrne and Bovair (1997) for a phenomenon that they named postcompletion error.

The tasks that people want to accomplish are usually distinct from the devices used to accomplish them. For example, the task might be to withdraw cash from a bank account and the device might be an automated teller machine (ATM). From the perspective that task and device are distinct, any action that the device (the ATM) requires us to perform after we complete our task (withdrawing cash) is a postcompletion action. An omitted postcompletion action is thus a postcompletion error. Postcompletion errors include leaving the card in the ATM after taking the money; leaving the originals on the photocopier after taking the copies; and forgetting to set the video cassette recorder (VCR) to record after programming it to videotape a show.

The striking characteristic of postcompletion errors is that, although they occur, they do not occur often. Most people, most of the time, take both the money and the card from the ATM (else, we suspect many fewer of us would use ATMs). What, if anything, predicts the occurrence of a postcompletion error?

Byrne and Bovair’s postcompletion error model is based on the notion of activation of memory elements. Activation is a hypothetical construct that quantifies the strength or salience
of information stored in memory. The postcompletion error model was constructed using CAPS, a programmable model of the human cognitive architecture (Just, Carpenter, & Keller, 1996). CAPS assumes that a memory element is accessible only if it has enough activation. It also assumes that total activation is limited. Activation flows from one memory element to another if the two are related and if one is the focus of attention. This spreading activation accounts for standard psychological effects like semantic priming, in which, for example, focusing on the notion of “doctor” might spread activation to related concepts like “nurse.”

In Byrne and Bovair’s error model, as long as the focus is on a task goal like getting money, related device actions like take the card continue to receive activation. However, when a task goal is accomplished, attention shifts away from it. When this shift occurs, the device actions associated with the task begin to lose activation. This is fine for actions like take the money, which are necessarily complete, but problematic for postcompletion actions like take the card. If these postcompletion actions lose enough activation, they will simply be forgotten.

Beyond its explanation, the postcompletion error model offered a prediction. Like most memory theories, CAPS assumes that unused memory elements decay over time; that is, their activation decreases. Because activation in CAPS is a common resource, decay of one memory element makes more activation available for other elements. Commensurately, Byrne and Bovair found fewer postcompletion errors in a condition that included a prolonged tracking task. Apparently the tracking task, which involved no memory load itself, allowed completed actions of the main task to decay. Postcompletion actions continued to receive activation because the task goal was not yet accomplished. In addition, they received the activation lost by the actions that decayed. This additional activation reduced postcompletion error.
As an example of applied theory, the postcompletion error model is important for several reasons. First, its explanations and predictions flow from existing cognitive theory, not from ad hoc assumptions made by the analysts. The model functioned primarily as a means of instantiating a theory on a particular problem. Second, the prediction comes from the model not the modeler. Any analyst could run the postcompletion error model with the same outcome. The debate over this outcome is then limited to and focused by the representational assumptions and parameter settings reified in a running computer program.

### 2.3 Information Access

Information in the world is useful only if we can find it when we need it. For example, an illustration in a book is helpful only if we know it exists, if we recall its existence when it is needed, and if we can find it. This view of information adds a cognitive dimension to research into information access (e.g., the HCI subareas of information retrieval and interface design). How do we recall the existence of the helpful illustration? What was stored in memory about the illustration, and exactly what is being recalled? What were the cues that prompted the recollection? From the cognitive perspective, the process of information access is complex. However, with a better understanding of the role of memory, we can engineer memory aids that support this process.

Altmann and John (in press) studied the behavior of a programmer making changes to code that had been written over a series of years by a team of which the programmer was a member. Verbal and action protocols (keypresses and scrolling) were recorded throughout an 80-min session. During this session, the programmer would trace the program for several steps, stop it, interrogate the current value of relevant variables, and so on. Over the course of the session
2,482 lines of code were generated and displayed on the programmer’s screen. On 26 occasions, she scrolled back to view information that had appeared earlier but had scrolled off the screen.

Of interest was the role of memory in these scrolling episodes. The volume of potential scrolling targets was huge and the programmer's need for any particular target was small. However, the protocol data revealed that scrolling was purposeful rather than random, implying a specific memory triggered by a specific cue. These constraints meant that the programmer's memory-encoding strategy must have been both sweeping in its coverage of potential targets and economical in terms of cognitive effort.

Altmann and John developed a computational cognitive model of episodic indexing that simulated the programmer's behavior. The model was developed using Soar, which, like CAPS, is a cognitive theory with a computational implementation (Newell, 1990). Based on the chunking theory of learning, Soar encodes sweeping amounts of information economically in memory, but retrieval of this information depends on having the right cue. When the episodic-indexing model attends to a displayed item (e.g., a program variable), Soar creates an episodic chunk in memory that maps semantic information about the item to episodic information indicating that the item was attended. A second encounter with the item triggers recall of the episodic chunk, which in turn triggers an inference that the item exists in the environment. Based on this inference, the model decides whether or not to pursue the target by scrolling to it.

The episodic indexing model suggests that memory depends on attention, not intent. That is, episodic chunks are stored in memory as a by-product of attending to an object, with no need for any specific intent to revisit that object later. The implication is that people store vast amounts of information about their environment that they would recall given the right cues. This, in turn, suggests that activities like browsing are potentially much better investments than we might have
thought. The key to unlocking this potential is to analyze the semantic structure of the knowledge being browsed and to ask how artifacts might help produce good cues later when the browsed information would be relevant.

3 The Cognition-Artifact-Task Triad

Almost everything we do requires using some sort of artifact to accomplish some sort of task. As Figure 2 illustrates, the interactive behavior for any given artifact-task combination arises from the limits, mutual constraints, and interactions between and among each member of the Cognition-Artifact-Task triad. Cognitive modeling requires that each of these three factors be incorporated into each model.

Traditional methodologies generally consider cognition, artifact, and task pairwise rather than altogether. For example, psychological research typically seeks experimental control by using simple tasks that require little external support, thereby focusing on cognition and task but minimizing the role of artifact. Industrial human-factors research often takes the artifact itself to be the task, largely ignoring the artifact’s purpose. For example, the proposed TAO workstation had an ergonomically designed keyboard and display but ignored the TAO’s task of interacting with the customer to complete a call. Finally, engineering and computer science focus on developing artifacts, often in response to tasks, but generally not in response to cognitive concerns. The price of ignoring any one of cognition, artifact, and task is that the resulting interactive behavior may be effortful, error-prone, or even impossible.

In contrast, cognitive modeling as a methodology is bound to consider cognition, artifact, and task as inter-related components. The primary measure of cognition is behavior, so analysis of
cognition always occurs in the context of a task. Moreover, analyzing knowledge in enough
detail to represent it in a model requires attention to where this knowledge resides -- in the head
or in artifacts in the world -- and how its transmission between head and world is constrained by
human perceptual/motor capabilities. Indeed, computational theories of cognition are now
committed to realistic interaction with realistic artifacts (see, for example, Anderson, Matessa, &
Lebiére, 1997; Howes & Young, 1997; Kieras & Meyer, 1997). Thus, given that human factors
must consider cognition, artifact, and task together, cognitive modeling is an appropriate
methodology.

4 Cognitive Modeling vs. Cognitive Task Analysis

Cognitive task analysis, broadly defined, specifies the cognitive steps (at some grain size)
required to perform a task using an artifact. Cognitive modeling goes beyond cognitive task
analysis per se in that each step is grounded in cognitive theory. In terms of the triad of Figure 2,
this theory fills in the details of the cognition component at a level appropriate to the task and the
artifact.

In Project Ernestine, for example, the manufacturer’s predictions about the proposed
workstation came from a cognitive task analysis. However, this analysis specified the cognitive
steps involved in using the workstation as the manufacturer saw them. The CPM-GOMS models,
in contrast, took into account theoretical constraints on cognitive parallelism and made
predictions that were dramatically more accurate. In the other two models the influence of theory
is strong as well. Most any cognitive analysis would identify memory failure as the cause of
postcompletion error. However, the model based on CAPS went further, linking decay of
completed goals to improved memory for pending goals. Similarly, memory is clearly a factor in
information access, but the model based on Soar detailed the underlying memory processes to highlight the potential of browsing and the importance of effective cues.

Our discussions of cognitive theory have focused on GOMS (John & Kieras, 1996a; John & Kieras, 1996b), ACT-R (Anderson & Lebiére, 1998), CAPS (Just et al., 1996), EPIC (Kieras & Meyer, 1997), and Soar (Newell, 1990). These are broad and integrated theories that deal with cognitive control (GOMS, ACT-R, and Soar), learning and memory (ACT-R, CAPS, and Soar), and perception and action (ACT-R and EPIC). There is also the class of connectionist or neural network models that offers learning and memory functions that have been highly successful in accounting for lower-level cognitive phenomena like visual attention (e.g., Mozer & Sitton, 1998). In sum, a broad range of theory is now available to elaborate the steps of a cognitive task analysis and thus produce cognitively plausible models of interactive behavior.

5 Dimensions of cognitive models

The models we have described are points in a much larger space. In general, a model simply represents or stands for a natural system that for some reason we are unable to study directly. Many psychological models, for example, are mathematical functions, like memory-retention functions or regression equations. These make accurate, quantitative predictions but are opaque qualitatively in that they provide no analysis of what lies behind the behavior they describe.

We have focused on models that characterize the cognitive processes involved in interactive behavior. Process models can make quantitative predictions, like those of the TAO model, but go beyond such predictions to specify with considerable precision the cognitive steps involved in the behavior being analyzed. To give a sense of the space of possibilities, we compare and contrast process models on two dimensions: generative vs. descriptive, and generality vs. realism.
5.1 Generative versus Descriptive

Two of our sample models are generative and one is descriptive. The postcompletion error and episodic indexing models actually generate behavior simulating that of human subjects. Generative models are implemented as executable computer programs (hence are often referred to as computational cognitive models) that take the same inputs and generate the same outputs that people do. The TAO model, in contrast, simply describes sequences of actions rather than actually carrying them out.

Generative models have several advantages. These include, first, proof of sufficiency. Running the model proves that the mechanisms and knowledge it represents are sufficient to generate the target behavior. Given sufficiency, evaluation can shift, for example, to whether the model’s knowledge and mechanisms are cognitively plausible. A second benefit is the ability to inspect intermediate states. To the extent that a model is cognitively plausible, its internal states represent snapshots of what a human operator may be thinking. A third benefit is reduced opportunity for human (analyst) error. Generative models run on a computer, whereas descriptive models must be hand-simulated, increasing the chance of error.

5.2 Generality versus Realism

Models vary in their concern with generality versus realism. Generality is the extent to which a model offers theoretical implications that extend beyond the model’s domain. Realism, in contrast, is the extent to which the modeled behavior corresponds to the actual interactive behavior of a particular operator performing a given task.

Project Ernestine showed high realism in that each model accounted for the behavior for an entire unit task; that is, one phone call of a particular call category for a particular workstation. These models were not general in that it would be difficult to apply them to any task other than
the one modeled. For example, they could not be applied to model ATM performance or VCR programming. Indeed, the existing models apply only to a particular set of call categories. If another call category were to be modeled, another model would have to be built.

In contrast, the models of postcompletion error and episodic indexing lack realism in that their accounts of behavior are incomplete. Byrne and Bovair’s model cannot perform the entire task, and Altmann and John’s model cannot debug the code. However, the implications of these models extend far beyond the tasks in which they are based. Situations involving postcompletion actions are susceptible to postcompletion error. If postcompletion actions cannot be designed out of an interface then special safeguards against postcompletion error must be designed in. Likewise, episodic indexing suggests that human cognition reliably encodes a little information about whatever it attends to. With the right cue, this information can be retrieved. These hypotheses bear on any artifact-task combination in which memory is an issue.

6 Summary

The space of cognitive process models, even within the space of models in general, is quite large (see Gray, Young, & Kirschenbaum, 1997 for an alternative cross-section). It used to be that developing process models required access to specialized hardware and software that was available only at certain locations. Fortunately, the technology of programmable cognitive theories has improved to the point where computational models can be run and inspected over the Web (e.g., most of the models discussed in Anderson & Lebiére, 1998 are available on the web). Access to such models enables the analyst to study working copies of validated models and potentially to build on, rather than duplicate, the work of others.

Cognitive modeling is the application of cognitive theory to applied problems. Those problems serve to drive the development of cognitive theory. Some applications of cognitive
modeling are relatively pure application with little return to theory. Of the three models we considered, the model of the TAO in Project Ernestine (Gray et al., 1993) best fits this characterization.

In contrast, the episodic indexing model (Altmann & John, in press) was driven by an applied question – how a programmer works on her system – but produced no new tool or concrete evaluation. Instead, it proposed a theory of how people maintain effective access to large amounts of information. This theory suggests a class of design proposals in which the artifact plays the role of memory aid.

In the middle, the model of postcompletion error (Byrne & Bovair, 1997) used existing theory to predict when an applied problem (error) was most likely to occur. On this middle ground, where theory meets problem, is where cognitive modeling will have its greatest effect – first on HCI, then on human factors.

7 References


8 Figures

Figure 1: Section of CPM-GOMS analysis for an operator-assisted call. The proposed workstation (bottom) has two fewer keystrokes totaling 6 motor and 2 cognitive steps. However, deleting these steps did not alter the critical path (shown in bold).

Figure 2: The Cognition-Artifact-Task Triad. The behavior of an operator using an artifact to perform a task is arises from the limits, mutual constraints, and interactions between and among each member of the cognition-artifact-task triad.
Perceives -
Operator, bill

Perceive-
silence (a)

Perceive-
this to

Perceive -
4 (1)

verifies bill

initiate
F1

initiate
F2

initiate
4 (1)

Perceptual
Cognitive
Motor
(right hand)

Current Workstation

Perceptual
Cognitive

Proposed Workstation

Artifact

Interactive Behavior

Cognition

Task