A COMPUTATIONAL MODEL OF SKILLED USE OF A GRAPHICAL USER INTERFACE

Muneo Kitajima

Industrial Products Research Institute
1-1-4 Higashi
Tsukuba Ibaraki 305, Japan
+81 298 54 6731
i8001@ipri.go.jp

Peter G. Polson

University of Colorado
Institute of Cognitive Science
Boulder, Colorado 80309-0345
+1 (303)492-5622
ppolson@clipr.colorado.edu

ABSTRACT

This paper describes a computational model of skilled use of a graphical user interface based on Kintsch's construction-integration theory [4, 8]. The model uses knowledge of a detailed representation of information on the display, a user's goals and expectations, knowledge about the interface, and knowledge about the application domain to compute actions necessary to accomplish the user's current goal. The model provides a well-motivated account of one kind of errors, action slips [14], made by skilled users. We show how information about the intermediate state of a task on the display plays a critical role in skilled performance, i.e., display-based problem solving [10].

KEYWORDS: user models, graphical user interfaces, display-based problem solving, action slips

INTRODUCTION

The goal of this paper is to present a computationally-based performance model of the skilled use of applications with graphical user interfaces like that of the Apple Macintosh. The model provides a theoretical analysis of some basic attributes of skilled human-computer interaction showing in detail how information about the intermediate state of a task on the display plays a critical role in skilled performance, that is, display-based problem solving [10]. Skilled users make a surprising number of errors [3, 14], and the model provides a well-motivated account of this behavior as well as other phenomena consistent with the assumption that skilled performance is not based on detailed, prestored plans [13, 15]. We summarize results from four sets of simulation experiments exploring the parameter space, demonstrating how skilled users can make errors, and validating the sufficiency of the model.

Display-based Human-Computer Interaction

Recent empirical studies of display-based human-computer interaction have provided evidence against standard plan-based theories (*e.g.* [3, 7]) of expertise in HCI. Mayes, Draper, McGregor and Oatley [13] report that experienced MacWrite users have poor recall memory for the names of menu-items. In addition, Payne [15] has shown that experienced users do not have complete knowledge about effects of commands.

These results provide support for theoretical frameworks that assume that a sequence of user actions is not pre-planned. Each action is selected making use of display feedback during the course of generating a sequence of actions necessary to complete a task [10, 15]. The display plays a crucial role in successful and smooth interaction; the interaction is *truly* mediated by the display.

Other researchers have developed theories of skilled performance in which successful interactions are mediated by the representations of intermediate states of a task presented in a display. Howes and Payne [5] extended the task action grammar framework to display-based, menu systems, which is a competence model of users' knowledge

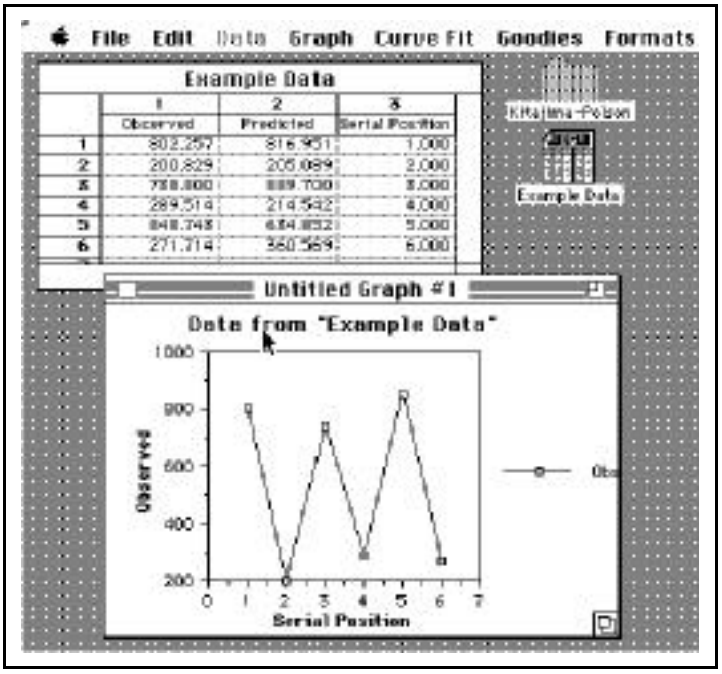

Figure 1. Example of the display in the task to plot the data from document "Example Data" using Cricket Graph, a Macintosh application.

of display-based systems for evaluating the consistency of an interface. Larkin [10] calls her framework display-based problem solving.

An Example Task

We want to account for the performance of a skilled user in tasks like the one shown in Figure 1. A user is plotting the data in the document "Example Data" using the Macintosh application Cricket Graph. On the screen are numerous objects that represent the current state of this task: the active window displaying the default version of the graph, a spreadsheet containing the data, the document icon for the data file, and an open folder.

An intermediate state of the task is shown in the display presented in Figure 1. The user has just completed plotting the column of numbers labeled "Observed" as a function of "Serial Position" and is starting to perform the subtask of editing the default version of the graph title. The user positions the arrow pointer on the title. The next correct action is to double-click on the title.

A Comprehension-Based Theory of Human-Computer Interaction

Our model is based on Kintsch's construction-integration theory of text comprehension [8]. It is related to similar models proposed by Mannes and Kintsch [12], Doane, Kintsch, and Polson [4], and Wharton and Lewis [18]. These models generate in real-time an action sequence necessary to perform a task in a manner analogous to a skilled reader constructing a contextually appropriate interpretation of a text during reading. In these models, the action selected in a specific situation is generated by a comprehension process which takes as input a representation of the user's current situation. This situation is defined by the user's current goal and expectation, information on the display, and facts about objects on the display and the task retrieved from long-term memory.

A MODEL OF SKILLED USE OF A GRAPHICAL USER INTERFACE

In this section, we begin with an overview of our theory of skilled performance of tasks using a graphical user interface. Next, we describe the construction-integration process. More details about the theory are provided in [8]. The following sections describe the different kinds of knowledge used by the model to generate correct action sequences, how parts of this knowledge are retrieved from long-term memory, how all of this knowledge is interconnected in a network, and the parameters of the model that control the retrieval and the integration processes.

An Overview

The model selects an eligible action given the current state of the display, user's current goal, and user's expectation concerning a desirable display state. Actions in the model are defined at a very small grain size. They include using the mouse to move the arrow pointer to an object, single-clicking an object, double-clicking an object, pressing and holding the mouse button, releasing the mouse button, typing, and the like. However, there is no direct representation in the model of sequences of actions that perform common subtasks like selecting a command from a menu. The model generates a sequence of actions when a sequence of display states and an associated goal and expectation are provided to it.

The Action-Selection (Comprehension) Processes

The construction-integration theory [8, 12] assumes that selection of the correct action (comprehension) is a two-phase process with construction and integration phases. In the construction phase, propositions representing a user's current goal and expectation, the state of the display, and knowledge retrieved from long-term memory are incorporated into a network using argument overlap to define links between individual propositions. The construction processes are driven bottom-up; they are not constrained by context. As a result, the network contains inconsistent information retrieved from long-term memory and representations of the correct action as well as many wrong actions.

In the integration phase, the correct action is selected using a spreading activation process. The action that is selected for execution is the one whose representation has the highest activation value and whose condition for execution is satisfied. The condition is considered to be satisfied when all propositions in it are found in the network. Some propositions in the condition describe the display state; these are incorporated in the network as display representations. The others extracted from knowledge stored in long-term memory; these are incorporated in the network by a probabilistic sampling process in the construction phase. The action changes the state of the display, defining a new display state. If necessary, a new goal or new expectation is retrieved. The cycle starts again with a new display and, if changed, a new goal or expectation as input to the construction phase.

The Knowledge

There are five kinds of knowledge used by the action selection process: goals, expectations, the representation of the display, general knowledge, and action plan knowledge. All are represented as propositions. A proposition is a tuple of the form (predicate, argument_1, argument_2, argument_3, ...,argument_n). In this paper, we will paraphrase the propositional representation paraphrased as simple sentences enclosed in square brackets. The predicate is represented in plain type. Arguments referring to specific objects on the screen are represented in italics, and bold arguments refer to abstract concepts.

Goals

Goals are representations of a user's intentions to perform actions on objects [6]. The same goal can be associated with a long sequence of display states. The goal proposition for the example task shown in Figure 1 is shown in (1).

[Goal is to perform **Edit Graph-Title** on the object *Graph-Title*] . (1)

Expectations

An expectation is the representation of the consequences of an action or sequence of actions in terms of the appearance of one or more objects on the display. It is associated with one or a sequence of display states. The model assumes that expert users can generate detailed representations of consequences of the next action. Expectations are closely related to Selz's anticipation schema [17]. The expectation proposition for the example task shown in Figure 1 is shown in (2).

[Expectation is to see entry into **Edit Graph-Title** environment associated with *Graph-Title*] . (2)

Goals and expectations are stored in long-term memory and are retrieved at the appearance of an associated display state. However, the retrieval process is not simulated in the current model.

Representation of the Display

The model assumes that the visual image of a screen is parsed into a collection of objects, each represented by propositions. Display-objects represent the state of a screen. They include objects that define the style of a particular interface, such as windows, menus, dialog boxes, file icons, the mouse, the pointer, and the keyboard. Other display objects are defined by an application or task, such as editing or drawing a graph, and are usually the contents of windows.

Figure 2 shows the representation for Figure 1. Each object is represented by at least three propositions. The first asserts that the object is on the screen. The second identifies the object as a display object. The third establishes a token-type relationship. Additional propositions for an object describe additional information about that object. For example, [*Column-1* holds *Observed*] represents a component of *Example-Data*.

The representation of displays contains limited information about appearance. Only information reflecting underlying system states that are relevant to the user are included in the representation. The representation of an object reflects the current internal state of the system and is used to let the user know legal actions on the object. For example, the mouse pointer in the Macintosh interface changes its appearance depending on the internal system state. In Figure 2, such representations as [*Pointer-Shape* is **Arrow**] and [*Pointer-Shape* is not **I-beam**] are indicating system's internal state. The mouse pointer takes on the arrow shape on the desktop and the I-beam shape when over an object that is editable. This distinction is crucial in defining the result of an operation such as horizontal dragging. With the I-beam pointer, it results in selection of that portion of the text; with the arrow pointer, it results in dragging the pointed-at object.

General Knowledge

General knowledge is knowledge about the components of the display objects, the attributes of an object or component, and the functions of or operations that can be performed on an object or component. It is represented as a collection of propositions. For descriptive purposes, we have partitioned this collection into different domains corresponding to the interface, tasks, and applications. This knowledge is stored in long-term memory and is incorporated in the network during the construction phase by a probabilistic memory retrieval process described in a later section.

For the task described in Figure 1, nine domains of knowledge are assumed. The Macintosh graphical user interface environment is represented in the domains of dialog box, icon, interface objects, menu, and mouse. The remaining domains are general facts about application, application objects like graphs and spreadsheets, and application functions like editing. The number of propositions totals to 271.

Figure 3 is an example of the general knowledge about the components of the graph domain, the properties of these components and the functions that can be performed on a component. Propositions stating [**X** has **Y**] define properties or components. Examples are [**Graph** has **Title**], and [**Graph** has **Horizontal-Axis**]. Propositions of the form [**X** afford to **Y**] assert that **Y** can be performed on **X**. An example is [*Graph-Title* afford to **Grab** for *Move*] .

Action Plan Knowledge

The users modeled in this paper are assumed to have highly generic low-level action plan knowledge required to use the interface. The action plan knowledge is a set of plan elements that represent knowledge about relations between a user's action and the interface's visible response. An action, such as pointing or single-click, is associated with multiple plan elements which describe different responses of the system depending on the current states of display objects. The types of actions and the number of variation of each action used for the simulation experiments described later are six plan elements for pointing, three for single-click, four for press-and-hold, two for release, two for double-click and two for type.

Plan elements are stored in long-term memory in a generic form and do not refer to specific objects in the current display. However, when retrieved from long-term memory and incorporated into the network in the construction phase, the variables in the generic form are bound to objects in the display. All plan elements are always incorporated into the network in the construction phase.

A plan element consists of three fields. An example of a plan element is shown in Table 1. The name field describes the action, double-clicking on particular graph title, and its result, bringing up a particular edit dialog box. The condition is a set of one or more propositions that must be contained in the network in order for the plan to be able to

be executed. The outcome holds a set of one or more propositions that describe the consequences of executing the plan element in terms of changes in the display.

Links Within the Network

The network is defined by links between individual propositions. Different kinds of links have different strengths. The link strengths between propositions are defined by the following six parameters; the argument overlap weight, W*arg*, free association weight, W*assoc*, plan relation weight, W*plan*, plan inhibition weight, W*inhib*, the goal magnification factor, F*goal* and the expectation magnification factor, F*exp*. The free association weight is defined in the discussion of the retrieval process.

Argument Overlap Weight

Argument overlap is the most basic link between propositions. This link type connects propositions defining goal, expectation, the display representation, general knowledge and the name fields of plan elements. In the example used above, the argument *Graph-Title*, which appears in the goal, expectation, representation of the display, and the general knowledge retrieved from long-term memory, serves as an argument for linking among them. For example, the display representation, [*Graph-Title* is a kind of **Title**], has connection to [**Graph** has **Title**], [*Graph-Title* has link to *Edit-Dialog-Box*], [*Graph-Title* affords to **Grab** for *Move*], and [*Graph-Title* is not a kind of **Text**] stored in long-term memory. When two propositions share one argument, they are connected by a link of strength of W*arg*.

Plan Relation Weight and Plan Inhibition Weight

Plan elements are different from the other kinds of knowledge since they describe procedural knowledge; they have condition and action parts and there are causal relations among them. If a condition of plan A is disabled by the execution of plan B, then plan A inhibits plan B. Or if a condition of plan A is satisfied by the execution of plan B, then plan A supports plan B. The strengths of causal relations are parameterized by W*plan* and W*inhib*.

Table 1. One of the two double-click plan elements bringing up an edit dialog box or starting an application.

Name:	Double-Click *Graph-Title* by **Arrow** for starting *Edit-Dialog-Box*
Condition:	
If	*Graph-Title* is on screen, *Graph-Title* is pointed at, above two *Graph-Title*s are identical *Graph-Title* has link to *Edit-Dialog-Box*, *Pointer-Shape* is **Arrow**, and *Pointer-Shape* is not **I-beam**
Outcome:	
Then	start *Edit-Dialog-Box*

Goal Magnification Factor and Expectation Magnification Factor

The links from goal and expectation to the rest of the network are treated differently from other links. The strengths of goal and expectation related links are calculated by multiplying the sum of W*arg* and W*assoc* by the manification factors F*goal* and F*exp*, respectively.

Retrieval of General Knowledge from Long-Term Memory Cued by Display, Goal, and Expectation

Recall that the display representation does not contain any knowledge of the attributes of an object or component, nor of the functions of or operations that can be performed on an object or component. These kinds of knowledge, which are retrieved by the probabilistic sampling process cued by display representations, goal, and expectation in the construction phase, play a crucial role in selecting a correct plan element to be executed. In order for the correct plan to fire, it must have the highest activation value among the executable plans, and all propositions in the correct plan's condition part must be incorporated in the network; part of them are found in the display representation and the rest in long-term memory. Plan elements receive activation from goal, expectation, and display representation propositions directly and/or via paths of links formed by general knowledge propositions, and from other plan elements by causal relations.

For example, the correct plan for the example task shown in Figure 1 is shown in Table 1. In order to make this plan executable, the proposition in the condition part, [*Graph-Title* has link to *Edit-Dialog-Box*], has to be retrieved from long-term memory because it is not a part of the display representation (see Figure 2 and Figure 3). The goal (1), expectation (2), and display representation (Figure 2) serve as retrieval cues for the proposition to be incorporated into the network.

The probabilistic sampling process works as follows. Each proposition in the goal, expectation, and display representation serves as a retrieval cue. The simulation first finds the set of all propositions in the general knowledge stored in long-term memory that are linked via argument overlap to a retrieval cue. Then the propositions in this set are sampled probabilistically based on the strength of the links between the retrieval cue and members of the set in long-term memory. Members of this set are sampled-with-replacement [16]. The number of retrieval attempts using the current proposition as the cue is specified by the sampling parameter, N*sample*. Because of the nature of the sampling-with-replacement process, dominant associates may be retrieved more than once, but only one example of the proposition is included in the network. Each time a proposition in long-term memory is retrieved, the link strength between the cue and this proposition is increased by W*assoc*.

RESULTS OF SIMULATION EXPERIMENTS AND DISCUSSION

We carried out four sets of simulation experiments. The first two were designed to evaluate the knowledge representations described in previous sections and to explore the model's parameter space.

The first set of simulations used a simple task of starting an application program by double-clicking its icon. The second set of experiments used a task that involved editing an icon label by inserting text.

The third set of simulations, using the same task as the second simulation, was designed to investigate the process by which the model makes errors by changing the sampling parameter a range of 4 to 14.

The fourth set of experiments simulated execution of part of the task shown in Figure 1. The simulated action sequence started with the spreadsheet containing the raw data on the screen. The model simulated two major subtasks. The first was to plot the observed data points as a function of serial position leading to the graph shown in Figure 1. The second subtask, shown in Figure 1, involved double-clicking the graph title to bring up a dialogue box which enabled the user to change the text of the title and font and point size of the text.

Exploration of the Parameter Space

A detailed report of the first two sets of simulation experiments is contained in Kitajima and Polson (in preparation). We briefly summarize the results here. The first set of simulation experiments showed that W*assoc* had little or no effect on performance of the model and therefore that parameter was set equal to 1.0 in all of the remaining experiments. In addition, W*plan* and W*inhib* should be set to 1.0 and -1.0, respectively, for optimal performance of the model.

In the second set of experiments, N*sample* was set to a large value, 14, assuring that the all necessary knowledge would be included in the network by the sampling process, F*goal* and F*exp* were set equal and manipulated over a range from 1 to 16, W*arg* from 1 to 4, and the other parameters were set as above. The results showed that the larger W*arg* becomes, the faster the integration phase converges, and the model consistently activates the correct plan element with the magnification factors (F*goal* and F*exp*) set to 16. The model will not activate the correct plan for small values of the magnification factors.

An Attentional Mechanism

The magnification factors had to be set to 16 before the correct plan would reliably get the highest activation and be executed. With the magnification factor equal to 16 and an overlap weight of 4, the link strength equals 64 for propositions in the network whose arguments overlap with the arguments of the goal and expectation propositions.

There are two reasons for this behavior. The magnification factor is a kind of "attention" parameter. During the construction phase, the large value of the magnification factor causes the model to preferentially sample propositions in long-term memory that overlap with the goal and expectation. In the step shown in Figure 1, the model will prefer to retrieve knowledge from long-term memory that elaborates the representation of the graph title. These elaborations are the critical propositions that provide links, a path, from the goal and expectation to the correct plan element.

During the integration phase, the large values of the link strengths between goal and expectation and the rest of the network, 64, cause the goal and expectation to be powerful sources of activation. Thus, paths that overlap arguments that appear in the goal and expectation will be highly activated. However, it is exactly one of these paths that leads to the correct plan element.

Errors

Studies of skilled users in a number of domains have shown that they have fairly high error rates. One of the earliest was Card, Moran, and Newell's [3] study of text editing, in which they observed about a 10% error rate. They hypothesized that experts were willing to trade off speed for accuracy because error correction was a routine skill and in most circumstances was not very costly.

There are numerous failure modes in the construction-integration model that can occur during the comprehension process. In this paper, we focus on the processes that cause errors due to action slips by expert users [14]. The model can make errors even when it is provided with correct goals, expectations, display state representations, all possible actions, and general knowledge. The stochastic memory retrieval process describe earlier can fail to sample critical pieces of general knowledge during the construction phase for small values of the sampling parameter. During action selection, the model uses the goal, expectation, and contents of the display to sample associatively related knowledge from long-term memory. Large values of the sampling parameter make it almost certain that all relevant knowledge will be included in the network. This memory retrieval process is part of Kintsch's [8] original model of text comprehension.

Sampling failures cause the model to build an incorrect representation during the construction phase. A skilled user fails to retrieve relevant information from long-term memory. As a result, critical information is missing from the network. The correct action may not be executable because its prerequisites are missing from the network, or the wrong action receives the highest activation because of the incomplete representation. We have assumed that the size of the sampling parameter is determined by a speed-accuracy trade off process.

Figure 4 shows the results of the third set of experiments that were conducted using the second task. Simulation runs

were carried for each of the following set of values of the sampling parameter (4, 6, 8, 12, and 14). Figure 4 plots the probability of successfully completing the label editing task as a function of the sampling parameter. It is particularly interesting to note that the model is capable of performing in a region, 90 to 95% correct, that is characteristic of expert behavior.

There are other error modes. The model may build a correct representation but the parameters describing the activation process are wrong, leading to an incorrect action receiving the highest activation. This occurs when the magnification factors were set to small values.

Many current models [1, 2] of skilled performance do not provide well-motivated explanations of errors. These models use collections of production rules to generate a task's goal structure in working memory and the action sequences to perform the task. However, there are no mechanisms in these models that would cause working memory failures or incorrect rule execution accounting for errors.

The simulation results from the second task shown in Figure 4 demonstrate that the comprehension-based model reported in this paper can make occasional errors and still be able to perform complex tasks. Errors are caused by the failure to use necessary general knowledge stored in long-term memory. The important point to emphasize is that the model does not guess. Errors are due to misunderstandings of the current display state leading to selection of the wrong action.

Display-Based HCI

This section describes how the model is display-based in the sense used by Larkin and Simon [11] and describes the results of the fourth set of experiments that simulated plotting and editing the graph shown in Figure 1.

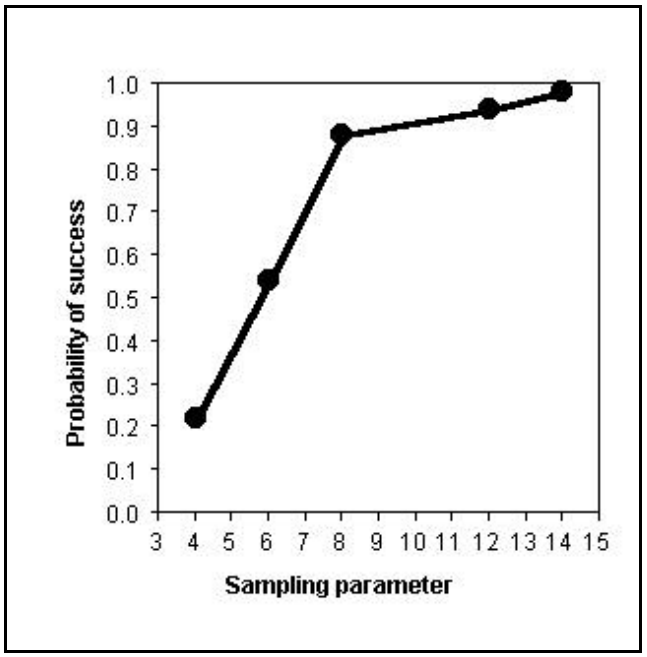

Figure 4. The probability of success plotted as a function of N*sample.*

Howes and Payne [5], Larkin and Simon [11], Larkin [10], and Kitajima [9] have all proposed models that made extensive use of information about intermediate states of a task contained in the environment or a display. Howes and Payne [5] took a grammatical approach to model roles of display in order to make analyses on the consistency of display-based interfaces. Larkin and Simon [11] were concerned with providing a principled account of why various kinds of visual displays could often dramatically facilitate problem solving processes. Larkin and Simon's [11] argument was that display-based problem solving enables individuals to substitute powerful perceptual operations for difficult and unreliably performed cognitive operations.

Classical information-processing models usually assume that users generate and maintain in working memory a complex goal structure that enables them to execute a sequence of actions necessary to perform a task [1, 2]. Larkin [10] and Kitajima [9] show that the state of the display can partially substitute for a complex goal structure stored in working memory. Both the display and the knowledge necessary to interpret to it substitute for a complex, potentially fragile, and difficult to maintain goal structure in working memory.

The Interactions of Goals, Expectations, and the Display

Table 2 presents the goals and expectations incorporated into the simulation that generated the sequence of actions to perform the task shown in Figure 1. The model has a few

Table 2. Goals, expectations and correct action steps for the third task.

G1	**to draw line graph**
E11	to see entering into line graph environment
	step 1: *Move Pointer to "Graph"*
	step 2: *Hold Down Mouse Button*
	step 3: *Move Pointer to "Line"*
	step 4: *Release Mouse Button*
E12	to see that "Serial Position" is selected as X axis
	step 5: *Move Pointer to "Serial Position" in "Horizontal (X) Axis" Scroll Window*
	step 6: *Click Mouse Button*
E13	to see that "Observed" is selected as Y axis
	step 7: *Move Pointer to "Observed" in "Horizontal (Y) Axis" Scroll Window*
	step 8: *Click Mouse Button*
E14	System draw line graph
	step 9: *Move Pointer to "New Plot"*
	step 10: *Click Mouse Button*
G2	**to edit graph title**
E21	edit graph title
	step 11: *Move Pointer to "Title"*
	step 12: *Double Click "Title"*

general subgoals. Expectations provide most of the knowledge that organizes the sequence of actions that performs the task. As we described earlier, the large value of the magnification factor for links between the expectation and display causes the model to focus its "attention" on the part of the complex display relevant to the current sequence of steps. By attention, we mean that the model preferentially retrieves information from general knowledge stored in long-term memory linked to the arguments of the expectation. The model also strongly activates paths from the expectation to plan elements with overlapping arguments in their name fields.

Each expectation in Table 2 describes the final result of a sequence of steps, which means that there is *not* a new goal-expectation pair for each step. The successive selection of the most eligible plan element at each display state associated with an expectation, that is, the sequencing of steps, is controlled by the condition field of the plan elements, whose truth value is sensitively affected by the step-by-step changes in the display state.

Another critical fact of our model is that it makes extensive use of information about intermediate states of a task contained in the display. The construction-integration cycle

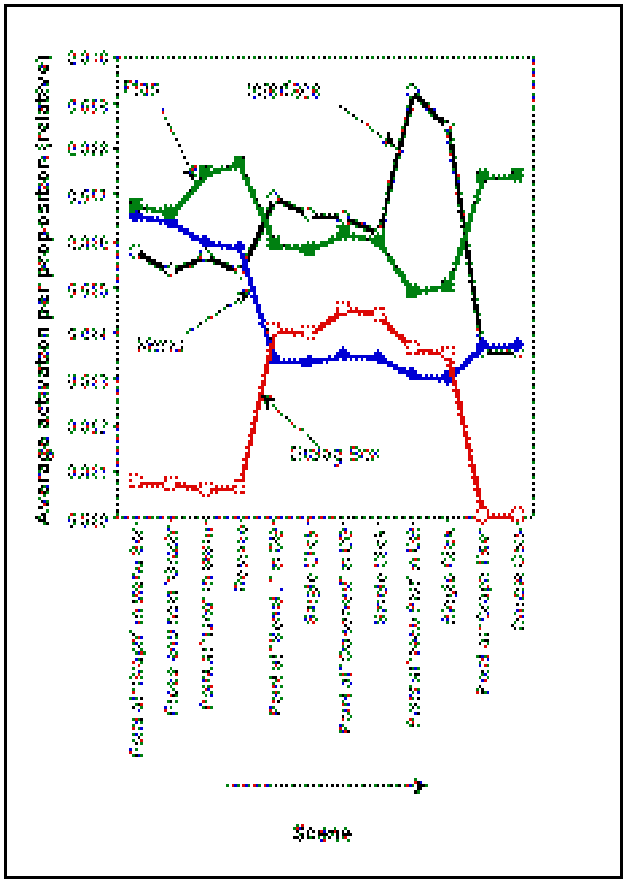

Figure 5. Knowledge use in the integration process.

enables the model to bring all relevant knowledge to bear on the problem of selecting the next correct action. The importance of a given knowledge domain in selecting the action is shown by the amount of activation of propositions in that domain.

Figure 5 shows activation values per proposition in each domain. These values are calculated by dividing the activation values collected by each domain by the number of propositions in the domain. The domains shown in Figure 5 changed their values very sensitive to the content of the display and the current expectation. From the figure, it is clearly seen what part of knowledge was doing work in the network. The other domains had relatively constant activation values throughout the task, including mouse, icon, graph, editor, and spread sheet. Their averages were around 0.002 which was remarkably smaller than those values shown in Figure 5.

SUMMARY

This paper has provided a detailed account of a computational model of the skilled use of a graphical user interface. It is highly flexible and is capable of making mistakes, hallmarks of skilled behavior.

Our most important contribution is the explanation of errors made by expert users who have complete, well-learned knowledge of how to perform tasks. The model can make errors because it must compute each correct action. It does not have a verbatim representation of the correct action sequence which is always successfully retrieved from memory. In our current models, the simulation is provided with the correct sequence of goals and expectations required to perform a task. The representation of the current situation is constructed from information in the goal, expectation, and the display. It is then augmented by information retrieved from long-term memory by the probabilistic sampling process. The information retrieved from long-term memory is critical to the interpretation of the information contained on the screen.

If critical information is not sampled during retrieval, the simulation can make an error. The sampling parameter determines the probabilities that the necessary knowledge will be retrieved from long-term memory. However, the likelihood of an error is dependent upon the details of the current situation. If the correct action can be selected based on knowledge of the goal, expectation and information in the display, missing information from long-term memory will have no effect. If, however, the correct action is dependent upon the information retrieved from long-term memory, sampling failures will lead to errors. In this way, the error is not a random response; the model is not guessing. The kinds of actions the model will choose are strongly constrained by the current goal, expectation, and state of the display.

The mechanisms mediating skilled performance contained in this model are display-based in Larkin's [10] sense. The results of the fourth set of simulation experiments show

how a model is able to compute the action sequence necessary to plot and edit the graph shown in Figure 1. Expectations described in terms of expected changes in the display guide execution of the task and focus the model's "attention" on the part of the display relevant to the current task. The model preferentially samples information from long-term memory providing a detailed elaboration of the relevant parts of the display. Utilization of information in the various knowledge domains varies as a function of step in this complex task. These variations are jointly determined by changes in the expectation and the state of the display. The step-by-step changes in display state are used for sequencing the actions to complete a task.

ACKNOWLEDGMENTS

We wish to thank Clayton Lewis, Walter Kintsch, David Kieras, and Stephanie Doane for their contribution to this research program. We thank John Rieman, Adrienne Lee, Peter Foltz, and Dannielle McNamar for their comments on earlier version of this paper. Muneo Kitajima was a visitor at the Institute of Cognitive Science, University of Colorado during our collaboration. Peter Polson's participation in this research was supported in part by NSF Grant IRI 87-22792 and by Army Research Institute Contract MDA903-89-K-0025.

REFERENCES

1. Anderson, J. R. Skill acquisition: Compilation of weak-method solutions. *Psychological Review*, 94 (1987), 192-211.

2. Bovair, A. S., Kieras, D. E., and Polson, P. G. The acquisition and performance of text-editing skill: a cognitive complexity analysis. *Human Computer Interaction*, 5, 1 (1990), 1-48.

3. Card, S. K., Moran, T. P., and Newell, A. *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates, Hillsdale, NJ., 1983.

4. Doane, S. M., Kintsch, W., and Polson, P. G. Modeling UNIX command production: What experts must know. ICS Technical Report #90-1. Institute of Cognitive Science, University of Colorado, Boulder CO., 1990.

5. Howes, A., and Payne, S. J. Display-based competence: towards user models for menu-driven interfaces. *Int. J. of Man-Machine Studies*, 33 (1990), 637-655.

6. Kieras, D. E. Towards a Practical GOMS Model Methodology for User Interface Design. In M. Helander (Ed.) *The Handbook of Human-Computer Interaction*. Amsterdam, NV: North-Holland , 1988.

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

8. Kintsch, W. The role of knowledge in discourse comprehension: A construction-integration model. Psychological Review, 95 (1988), 163-182.

9. Kitajima, M. A formal representation system for the human- computer interaction process. *Int. J. Man-Machine Studies*, 30 (1989), 669-696.

10. Larkin, J. H. Display-based problem solving. In D. Klahr and K. Kotovsky (Eds.). *Complex Information Processing: The Impact of Herbert A. Simon*. Lawrence Erlbaum Assoc, Hillsdale, New Jersey, 1989, 319-342.

11. Larkin, J.H., and Simon, H.A. Why a diagram is (sometimes) worth 10,000 words. *Cognitive Science*, 11 (1987), 65-100.

12. Mannes, S. M., and Kintsch, W. Routine computing tasks: Planning as Understanding. *Cognitive Science*, 15 (1991), 305-342.

13. Mayes, J.T., Draper, S.W., McGregor, M.A., and Oatley, K. Information flow in a user interface: the effect of experience and context on the recall of MacWrite screens. In *People and Computer IV*, D.M. Jones and R. Einder, Eds., Cambridge University Press, Cambridge, UK., 1988.

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

15. Payne, S. J. Display-based action at the user interface, *Int. J. of Man-Machine Studies*, 35 (1991), 275-289.

16. Raaijmaker, J. G., and Shiffrin, R. M. Search of associative memory. *Psychological Review*, 88, 1981, 93-134.

17. Selz, O. The laws of cognitive activity, productive and reproductive: A condensed version. In Frijda, N.H. and De Groot, A. *Otto Selz: His Contribution to Psychology*. Mouton Publishers, The Hague, The Netherlands, 1990, 20-75.

18. Wharton, C., and Lewis, C. Soar and construction-integration model: Pressing a button in two cognitive architectures. Technical Report #CU-CS-466-90. Department of Computer Science, University of Colorado, Boulder CO., 1990.

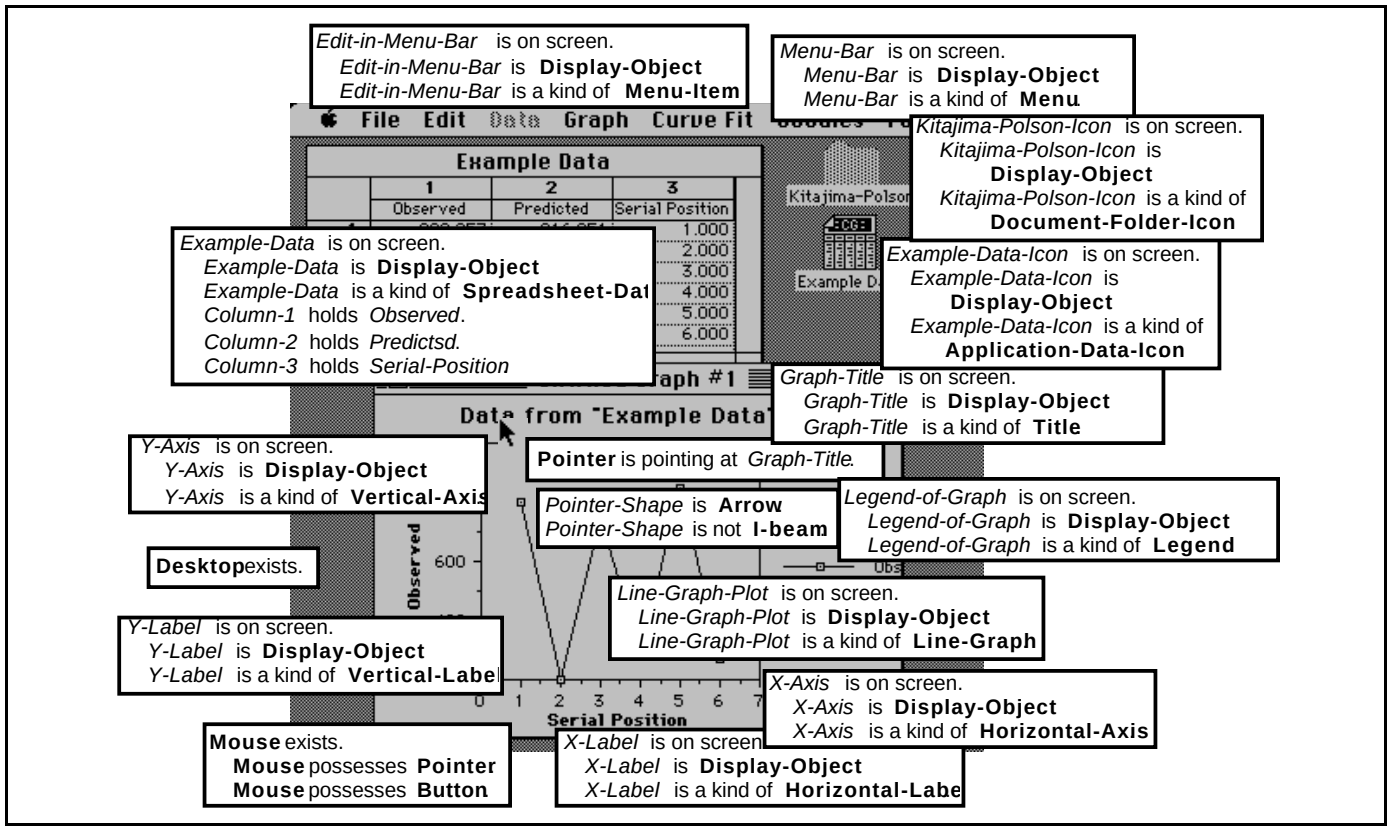

Figure 2. The propositional representation of the display shown in Figure 1 that is input into the model. See text for a description of the propositional notation.

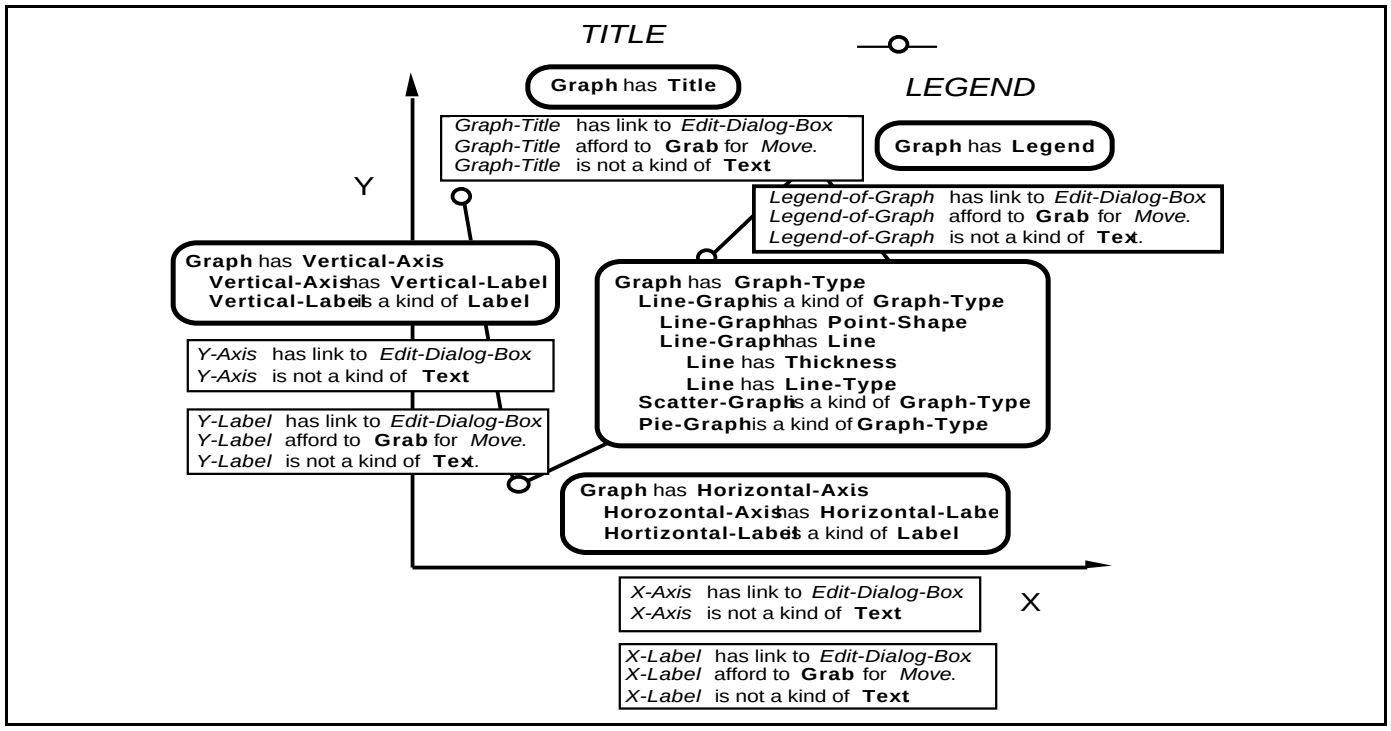

Figure 3. Example of representation in long-term memory concerning graphical domain knowledge.