CHAPTER ??

SUCCESSFUL TECHNOLOGY MUST ENABLE PEOPLE TO UTILIZE EXISTING COGNITIVE SKILLS

Muneo Kitajima

National Institute of Bioscience and Human-Technology
kitajima@nibh.go.jp

INTRODUCTION

Many of our activities are purposeful because we interact with our environment to achieve specific goals. What we actually do at a given moment, however, is determined not only by the goals and the environment but also by the knowledge utilized to comprehend the situation. To select what to do, we integrate these sources of information: goals, information from the environment, and knowledge relevant to the current situation. This is especially true when we must discover the actions necessary to accomplish goals in unfamiliar situations.

In this chapter, text comprehension is regarded as one of the fundamental cognitive skills that could be applied to deal with these situations. Text comprehension is a highly automated collection of cognitive processes that make use of massive amounts of knowledge stored in long-term memory. Readers activate knowledge from long-term memory relevant to the current reading goal and integrate this knowledge with the current goal and representation of text. Conflict among activated knowledge elements may exist which necessitates an integration process to arbitrate this conflict within an appropriate time frame.

The goals of reading are diverse—from collecting information from technical documentation and solving word problems to guessing who a criminal might be in a detective story—but people still apply a universal, and fundamental primary text comprehension skill to each comprehension activity. This chapter suggests yet another goal-directed activity where the text comprehension skill is employed—interacting with graphical user interfaces (GUI). In this activity, people read task descriptions and interact with computer applications to achieve their tasks. Objects on the screen replace the text of ordinary reading. This chapter presents a comprehension-based model for these processes, called the **LI**nked model of **C**omprehension-based **A**ction planning and

Instruction taking (**LICAI**).[1] This model is based on the construction–integration theory, a well-established cognitive model of text comprehension, developed by Kintsch (1988).

Text comprehension is probably one of the most fundamental skills people employ when they interact with their environment in goal-directed activities. However, when this skill is applied to different domains such as human–computer interaction, it might not work properly if the environment is not designed to facilitate its use. Therefore, I suggest that the LICAI model be used to identify cognitive problems that might occur when people engage in activities where they take instructions to perform tasks. A LICAI simulation is presented to demonstrate this point.

This chapter begins by describing an example situation identifying two fundamental cognitive processes indispensable to generating comprehension-based, goal-directed interactive activities. One is the goal formation process, and the other is the goal–action mapping process. Experimental results supporting the goal formation process are then described. The next section describes the LICAI model, which assumes that these activities are controlled by the comprehension processes; it is followed by a simple simulation of taking realistic instructions to perform tasks. Results are summarized in light of factors to be considered in designing instructions and interface displays that conform to the comprehension processes.

ACCOMPLISHING GOALS ON UNFAMILIAR INTERACTIVE DEVICES

Coordinating Goals and Actions

Imagine a situation where one is faced with an unfamiliar interactive device to accomplish a certain goal. He or she may or may not successfully discover appropriate actions. What would determine the results? What kinds of cognitive processes would work? How would they work? This section addresses these issues by introducing one of my experiences in Germany. The episode was as follows:

> I arrived at the airport in Stuttgart, Germany, at about 9 p.m. I decided to go to my hotel by train. I had no trouble finding the train station by following well-designed signage. Then I had to buy a ticket from Stuttgart airport station to the station where my hotel was located: Neckartor. I stood facing a ticketing machine. As this was my first visit to Germany, I had never used this machine. Even worse, I could not read German. I could not find anyone to ask how to operate it. For a few minutes, I observed the machine and the board next to it. I thought I understood what was expected. I read a three-digit code off the board representing the destination and entered the code from a ten key pad on the panel. The machine then showed the fare. I inserted the necessary coins in the slot. The machine issued the ticket. I took it.

---

[1] When LICAI is pronounced [li kai], the pronunciation represents a two-kanji Japanese word, 理解, meaning ‘comprehension.’

I discovered a successful sequence of actions by myself. However, the sequence of actions was completely different from the one I normally use in my home country, Japan. The question is: How did I perform this task successfully without the knowledge of the sequence of correct operations?

What follows describes a rather informal analysis of the above episode to show the basic ingredients that support goal-directed activities in novel situations. The analysis is based on Norman's (1986) action theory framework, which consists of goals, the stage of evaluation, the stage of execution, and the environment. In addition, the analysis assumes that I transferred the knowledge I normally employ in Japan to the situation I encountered in Germany. Thus the analysis maps terms used to explain my ticket-buying activity in Japan to similar activity in Germany. Table 1 summarizes the results of the analysis.

Table 1. A comparison of the sequences of actions involved in the ticket-buying activity.

In Japan			*In Germany*		
State of the environment	*Goal*	*Action*	*State of the environment*	*Goal*	*Action*
Railway map, Fare table	Communicate fare (G1)	Read fare from table (A1)	Railway map, Table of destination	Communicate destination (G2)	Read code from board (A1)
Ticket machine, Slots for coins		Insert coins (A2)	Ticket machine, Numeric keypad		Press buttons (A3)
Ticket machine, Buttons	Communicate destination (G2)	Press button (A3)	Ticket machine, Slots for coins	Communicate fare (G1)	Insert coins (A2)
Ticket Machine, Change, Ticket	Obtain ticket (G0)	Take ticket and change (A4)	Ticket machine, Change, Ticket	Obtain ticket (G0)	Take ticket and change (A4)

Goals

In the analysis, two common subgoals were identified that are decompositions of the top level goal, "Obtain a ticket to the destination (G0)":

Subgoals:

G1. Communicate the fare.
G2. Communicate the destination.

In Japan, I obtain the information about fare first, then insert necessary coins and communicate the destination by pressing an appropriate button. Thus, G1 applied first, followed by G2 and G0. By contrast, in Germany, the order of the subgoals was reversed. What I had to do first was to obtain the code for the destination. Then I had to enter the 3-digit code, say '345,' into the ticket machine, then the machine displayed the fare, and I inserted coins in the slot on the machine. What has to be emphasized here, however, is that the subgoals I know from experiences in Japan were successful in the completely novel situation, even if the order of application was changed.

Actions

Similarly, four common actions were identified:

Actions:

A1. Read X from Y.
A2. Insert coins.
A3. Press buttons.
A4. Take the ticket and change.

In Japan, the sequence of action–goal pairs were A1 followed by A2 for G1, then A3 for G2. In Germany, A1 followed by A3 for G2, then A2 for G1. In both countries, A4 was performed for accomplishing the top level goal, G0. The first actions A1 in both countries were apparently the same, but the purposes were different; in Japan, "Read the *fare* from the *table* (A1) in order to communicate the fare (G1)", whereas in Germany, "Read the *code* for the destination from the *board* (A1) in order to communicate the destination (G2)." A2 and A3 were performed to accomplish the goals G1 and G2, respectively. Because the order of the goals were reversed, the order of these actions were reversed.

Necessary Processes for Mapping a Top Level Goal on Actions

The analysis shows the pieces—the subgoals and the actions—that were successfully reorganized to achieve the top level goal in an unfamiliar situation. There are three important processes:

The first process is to generate correct subgoals for a top level goal and to make these available during the task. In the example, G0 was decomposed into G1 and G2, and they were maintained during the interaction [*the goal formation process*].

The second is to select a correct subgoal from the available subgoals. The selection was done by integrating information from the environment with the available subgoals. In Japan, the presence of the fare table was critical when the subgoal "communicate the

fare (G1)" was selected, whereas in Germany, the presence of the table of destination was critical when the subgoal "communicate the destination (G2)" was selected [*the goal selection process*].

The third is to map the selected subgoal onto a correct sequence of actions. Again, the selection of an action is done by integrating pieces of information such as information from the environment (e.g., the appearance of the ticket machine, the fare table, highlighting of buttons, etc.), its elaborations by using knowledge from long-term memory (e.g., a button can be pressed for communication), and the current subgoal [*the goal–action mapping process*].

GENERATION OF GOALS FROM INSTRUCTIONS

In this episode, the correct actions would have never been discovered unless the correct set of workable subgoals had been generated. Having the knowledge to transform the top level goal into the subgoals was crucial. Such transformation is a well-known process that has been studied in detail in the context of word problem solving (Kintsch, 1988; Kintsch and Greeno, 1985). The original problem statement, for example, "Joe had 8 marbles. Then he gave 5 marbles to Tom. How many marbles does Joe have now?" must be transformed into an abstract form that is useful for arithmetic calculation before actual calculation is performed. Even if one has excellent calculation skills, they are useless if a correct transformation has not been achieved.

This section introduces a laboratory experiment conducted by Terwilliger and Polson (1996) that shows how people really transform their original goal given as task instructions into workable subgoals in the context of human–computer interaction. Terwilliger and Polson measured the time it took experienced Macintosh users, who had never used a graphing application, to interact with two forms of the variable selection dialog box. They found clear evidence that people make such transformations.

In the experiment, the user first read a single sentence of instructions, then created a graph from pre-existing data by pulling down a menu, releasing on a menu item, and then assigning variables to axes in a dialog box. The variables in the data to be graphed were "absences" and "month."

Two versions of instructions were considered:

Instructions:

- XY instructions

  "Create a graph with month on the X-axis and absences on the Y-axis."

- FN instructions

  "Create a graph of absences as a function of month."

Similarly, two versions of dialog box for the assignment of axes were devised:

Dialog boxes:

- XY dialog box

The left selection list was labeled “X Axis:” and the right selection list was labeled “Y Axis:”.

- FN dialog box

  The left list was labeled “Plot:” and the right list was labeled “As a Function of:”.

Different subjects were exposed to all combinations of instructions and dialog box type. Sixteen subjects were drawn from the introductory psychology subject pool at the University of Colorado and received course credit for their participation in the experiment. Four versions of the system were created, one for each combination of “XY” or “FN” instructions with an “XY” or “FN” dialog box. Subjects read the instructions on one sheet of a workbook, then switched to a different sheet to perform the necessary actions. The total time to create the graph was recorded automatically for each subject. The average times for each condition are shown in Table 2. An ANOVA with two between-subjects variables revealed that, on average, the task took significantly longer when the dialog box had the “FN” labels than when it had the “XY” labels. There were no other significant effects or interactions. A set of planned comparisons revealed that the average times for the two versions of dialog box were significantly different for each version of the instructions, but that the times for the two versions of the instructions were not significantly different for either version of the dialog box.

Table 2. Average time in seconds to create graph by instructions and dialog box type. (adapted from Terwilliger and Polson, 1996)

	Instructions Type		
Dialog Box Type	*XY*	*FN*	*Average*
XY	M = 80.31 SE = 10.52	M = 78.59 SE = 6.43	M = 79.45 SE = 5.71
FN	M = 117.58 SE = 11.89	M = 113.55 SE = 8.45	M = 115.57 SE = 6.80
Average	M = 98.95 SE = 10.18	M = 96.07 SE = 8.23	M = 97.51 SE = 6.34

In this experiment, the subjects would have transformed “FN” instructions into the form comparable with the representations for “XY” instructions when they finished reading it. In my episode in Germany, the internally generated top level goal, G0, would have been transformed into the subgoals, G1 and G2. In both cases, the representations of goals were different from the original ones. People would use goals represented in very specialized forms suitable for selecting actions on the interfaces. These goals would

have been generated by decomposing and/or transforming their original goals, which can be either internal, like 'obtain ticket,' or external, like 'XY' or 'FN' instructions.

A COMPREHENSION-BASED MODEL OF GOAL FORMATION AND GOAL–ACTION MAPPING: LICAI

Above, I outlined the three processes indispensable for discovering correct actions when interacting with novel interfaces: the goal formation, goal selection, and goal–action mapping processes. Goal formation through transformation has been evidenced by the laboratory experiment described in the previous section. Goal–action mapping has been studied extensively in the field of human–computer interaction (e.g., Hutchins, Hollan, & Norman, 1986; Payne, Squibb, & Howes, 1990).

This section describes a model that integrates these three processes. The basic idea is that each of these processes can be modeled as a comprehension process, characterized as highly automated cognitive processes that uses massive amounts of knowledge stored in long-term memory. The model is called **LI**nked model of **C**omprehension-based **A**ction planning and **I**nstruction taking (**LICAI**), developed by Kitajima and Polson (1996, 1997), which deals with situations where people take instructions and map their understanding onto actions on interfaces. It is currently being applied to model people's interaction within office automation and flight automation environments.

The cognitive processes specified in LICAI are implemented using the construction–integration architecture developed by Kintsch (1988). Thus, I start by describing this architecture and then explain the LICAI model.

Construction–Integration Architecture for Comprehension Process

The construction–integration architecture is symbolic-connectionist and has been applied successfully to model cognitive processes such as text comprehension (Kintsch, 1988), word problem solving (Kintsch, 1988), and action planning (Mannes & Kintsch, 1991; Kitajima & Polson, 1995). The construction–integration architecture assumes that parsers map surface representations into propositional semantic network representations (text in Kintsch, 1988, and visual displays in Kitajima and Polson, 1995), and a bottom-up, weakly constrained, rule-based process generate alternatives from the semantic representations. The rules are not context sensitive; therefore, the alternatives represented in the network may not be consistent with the current context.

The construction phase generates a network of propositions that contain a representation of the input (text or visual display), alternative meanings and interpretations of the input, and possible alternative actions. This network also incorporates the knowledge necessary to select among the alternatives. This knowledge includes goals, information retrieved from long-term memory, and information carried over from previous construction–integration cycles. A fundamental linking mechanism assumed by the construction–integration theory is the argument overlap mechanism: when two nodes share symbols, they are connected.

The integration phase selects an alternative by integrating information represented in the network generated during the construction phase. Integration can be thought of as a

constraint satisfaction process. The network of interconnected propositions defines a collection of constraints that are satisfied by the selected alternative. Integration is performed in a spreading activation process. The nodes in the network can be further divided into sources of activation, targets of activation, and links between sources and targets. Goals and the representation of the current context (i.e., text or visual display) are typical sources, and the targets are alternatives. The linking information comes from long-term memory and other sources, and the spreading activation process is controlled by the pattern of links in the network. When the integration phase terminates, the most highly activated alternative represents the result of the construction–integration cycle that satisfies the constraints. Because propositions in the network are linked by shared arguments, the pattern of argument overlap plays a key role in the results of the integration phase.

The LICAI Model

Overview

Figure 1 schematically describes the LICAI model. The three main processes are expressed by various construction–integration cycles along with the kinds of information that constitute networks at various moments.

The *goal formation process* is modeled as a problem-model construction cycle, which is a strategic form of the basic text-comprehension process that generates representations specialized for interacting with devices; that is, the goals that control the solution of a task described in instructions.

If the text contains descriptions of multiple goals, LICAI assumes that they are stored in episodic memory during the goal formation process. When the user finishes reading the instructions and attempts to perform one or more steps of the task, LICAI uses the current application display as retrieval cues to select a single goal from the episodic memory. This *goal selection process* is modeled as a memory retrieval cycle, a variation of the basic construction–integration cycle developed by Kintsch and Welsch (1991) as a model of cued recall.

The *goal–action mapping process* takes the goal retrieved from the episodic memory and attempts to generate one or more actions to satisfy the goal on the interface display. This process is a generalization of the model of skilled performance using a computer with a graphical user interface developed by Kitajima and Polson (1995), modeled as a pair of an attention cycle and an action planning cycle.

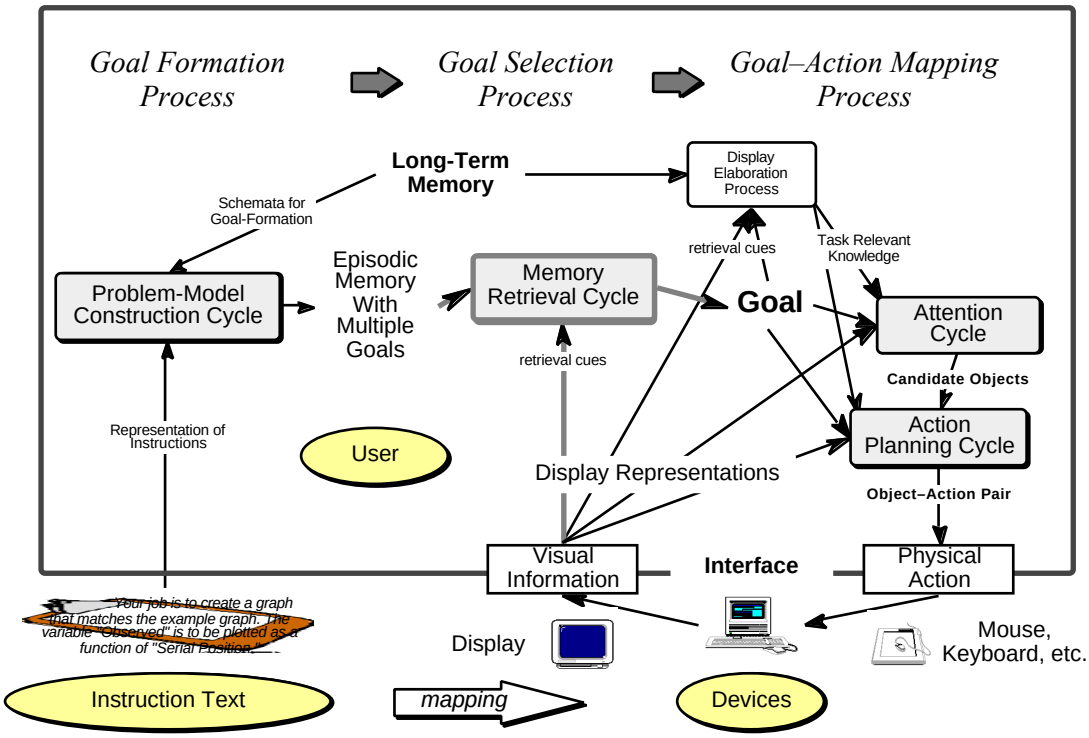

Figure 1. The LICAI model (Kitajima and Polson, 1997).

Goal Formation Process (Represented with thin lines in Figure 1)

When reading instructions, a user attempts to extract goals that should be accomplished on an interface. LICAI assumes that this process is analogous to solving word problems. Instructions are processed by executing a single construction–integration cycle for each sentence. In the construction phase, LICAI generates a network that includes semantic representations of a sentence as well as elaborations that translate the semantic representation into goals. In the integration phase, LICAI selects a single meaning for the sentence and links this representation with the memory representation of earlier parts of the text. Thus, after reading the entire text, the memory contents represent the result of instruction comprehension.

LICAI incorporates three kinds of schema that work during the construction phase. A schema is a knowledge structure that takes a semantic representation of instructions as input and generates one or more specialized propositions defined by a predicate and slots with strong constraints on the admissible arguments. *Global instruction reading schemata* represent the top-level strategy used by a reader to process text that describes a given task. All verbs with the implicit subject YOU are mapped into a text base proposition of the form DO [YOU, verb, object].

Task-domain schemata elaborate DO propositions and generate a more complete description of a task. For example, the original "FN" instructions,

DO [YOU, PLOT, AS-A-FUNCTION-OF [ABSENCE, MONTH]]

will be converted into two propositions:

DO [YOU, PLOT, ON [MONTH, X-AXIS]], and
DO [YOU, PLOT, ON [ABSENCE, Y-AXIS]].

Task-goal formation schemata transform DO propositions into propositions that represent goals that control the goal–action mapping processes. For example, the transformed instructions will be converted into the following forms:

PERFORM [PLOT, MONTH, X-AXIS], and
PERFORM [PLOT, ABSENCE, Y-AXIS].

Goal Selection Process (Represented with dotted lines Figure 1)

After reading the instructions, the user tries to select a goal from memory to perform in the current situation. LICAI assumes that a goal is selected from memory by a single construction–integration cycle. In the construction phase, the memory and the current interface display constitute a network. In the integration phase, a goal consistent with the current interface display is selected; that is, the most highly activated goal is selected. Since the sources of activation are the nodes representing the display, and the pattern of links in the network is largely determined by the argument overlap mechanism, a goal that overlaps the currently visible screen objects is likely to be selected. For example, if the representation of the goal includes matching labels on any screen objects, it will be selected.

Goal–Action Mapping Process (Represented with thick lines in Figure 1)

After selecting a goal, the user tries to generate a sequence of one or more actions that will accomplish the selected goal. This process is a generalization of the model of skilled, display-based action planning developed by Kitajima and Polson (1995), which involves two construction–integration cycles: the attention cycle and the action planning cycle.

As semantic knowledge of words is required to comprehend texts, knowledge about objects on the screen is also indispensable for successful interaction with display-based interfaces. The initial display representations contain only limited information about the identity of each object and its appearance, including visual attributes (e.g., color, highlighting). The poor display representations are augmented by retrieving relevant knowledge from long-term memory. This display elaboration process is simulated by a random memory sampling process: the retrieval cues are the selected goal and the propositions representing the current display. The elaboration process is stochastic and is taken from Kintsch (1988) where Raaijmakers and Shiffrin's (1981) model was used to describe the retrieval process. The probability that each cue retrieves particular information in a single memory retrieval process is proportional to the strength of the link between them. The model carries out multiple memory retrieval in a single elaboration process. A parameter, *the elaboration parameter*, controls the number of times each argument in the display and goal representations is used as retrieval cue. Kitajima and Polson (1995) discussed in detail the predictions and implications that follow from this stochastic elaboration process.

The retrieved information elaborates the display representation, providing information about interrelationships between display objects, relationships between the

goal and display objects, and other attributes of display objects. For example, if Object23 is the screen object in the X-Axis selection list labeled by Month, then the following items are stored in long-term memory about Object23 and can be retrieved by the display elaboration process:

- Object23 has-label Month
- Object23 is-a-member-of Line-Graph-Dialog-Box
- Object23 can-be-pointed-at
- Object23 can-be-selected

The elaborated display representation is the model's evaluation of the current display in the context defined by the goal. This corresponds to the stage of evaluation of Norman's (1986) action theory framework.

In the goal–action mapping process, the model first limits its attention to three screen objects (out of ~100 objects displayed on the screen) by applying an attention cycle. These screen objects are candidates for the next action to be operated upon. During the construction phase, a network is generated that consists of nodes representing the goals, the screen objects and their elaborations, and candidate object nodes of the form 'Screen-Object-X is-attended.' Any screen objects are potential candidates. During the integration phase, the conflict is to be resolved. The sources of activation are the goals and the screen objects. The targets are the candidate object nodes. When the spreading activation process terminates, the model selects the three most highly activated candidate object nodes. These nodes represent screen objects to be attended to during the action planning cycle.

The result of the integration process is dominated by two factors: the strengths of links from the representation of the goals, which is parametrized by a parameter, *the attention parameter*, and the number of propositions that are necessary to bridge the goals and the candidate objects.

The second construction–integration cycle is an action planning cycle. As preparation for constructing a network, the candidate objects carried over from the preceding cycle are combined with any possible actions to form object–action pairs of alternatives. The model considers all possible actions on each candidate object. The Kitajima and Polson (1995) model incorporates 18 possible actions.[2] Examples would include 'single-click Object23,' 'move Object23,' and the like.

During the construction phase, the model generates a network that includes the goals, the screen objects and their elaborations, and representations of all possible actions on each candidate object. During the integration phase, the sources are the goals and the screen objects, and the targets are the nodes representing the combinations of

[2]Representations of actions define different functions of single physical actions in many different contexts. For simulating a graph drawing task, the model defines eighteen cognitive actions on six physical actions; Move-Mouse-Cursor, Single-Click, Double-Click, Hold-Mouse-Button-Down, Release-Mouse-Button, and Type.

object–actions. The pattern of activation is determined by the same factors for the attention cycle. At the end of the integration phase, the model selects the most highly activated object–action pair whose preconditions are satisfied as the next action to be executed. The action representations include conditions to be satisfied for their execution. The conditions are matched against the elaborated display representations. Some conditions are satisfied by the current screen, others by information that was retrieved from long-term memory in the elaboration process. For example, the model cannot select an action to double click a document icon for editing unless the icon is currently pointed at by the mouse cursor and the information is available that the icon can be double clicked. Observe that if information about a necessary condition is missing from an elaborated display representation, the model cannot perform that action on the *incorrectly* described object.

Failure in Goal–Action Mappings

In a set of simulation experiments reported in Kitajima and Polson (1995), it was found that the goal–action mapping can fail due to the following three reasons.

The first is that the attention cycle can fail to include the correct object on the list of candidate objects. The second is that the action planning cycle can fail in which the correct object–action pair cannot become the highest activated among executable ones. In the model's terms, these kinds of errors are ascribed to both or either of low values of the attention parameter, and /or missing bridging knowledge that had to be retrieved from long-term memory.

The third reason is that the elaboration process fails to incorporate all of the conditions for the correct action in the elaborated display representation. Low values of the elaboration parameter cause this error. Parameter values in the range of 12 to 20 caused the model to simulate error rates in the range of 10% to 20% (Kitajima and Polson, 1995).

The first and the third reasons are internal to the model, whereas the second reason, missing bridging knowledge, would be controlled externally; for example, we can reduce the possibility of failure by carefully designing instructions and interfaces. However, note that this insight comes only from the nature of the goal–action mapping process where the goal has already been selected. The goal–action mapping process does not know whether the selected goal is the correct one or the wrong one.

The next section focuses on LICAI's simulation of the whole processes to show the second reason can also result in failure to select correct goals. Thus, possibility of failure increases dramatically if the whole process are considered. The final section provides a summary of these analyses.

SIMULATION OF TAKING AND CARRYING OUT LONG INSTRUCTIONS

This section applies the LICAI model to the experimental situation studied by Franzke (1994, 1995). In her experiment, the participants were given instructions as a HyperCard stack. Tasks that participants were given were to create a graph specified in the instructions and make several edits on the default graph. This section reports the

results of LICAI's simulation and identifies potential cognitive problems that the participants would have faced. A comparison of Franzke's results with LICAI's simulation can be found in Kitajima and Polson (1997).

Simulation of Performance

Following Franzke's (1994, 1995) experiments, let's assume that the participants have read the following instructions and memorized them. Then they tried to map the results of comprehension onto actions.

> *Instructions*
>
> In this experiment you are going to learn a new Macintosh application, Cricket Graph, by exploration. The task you are going to perform will be presented to you as a series of exercises. The data you are going to plot is contained in a Cricket Graph document, "Example Data." Your overall goal is to create a new graph that matches the example graph shown in the instructions. Your first exercise is to plot the variable "Number of Accidents" as a function of the variable "Month." After you have created a new graph, you will modify it so that it more closely matches the example given in your instructions.

Goal Formation

The LICAI model reads the above sentences and extracts any potentially useful goals by transforming the textual representation with the help of comprehension schemata for goal formation. For example, by reading the first sentence, LICAI elaborates it to generate a goal 'perform "Learn Cricket-Graph."' Before reading the second sentence, the goal is stored in episodic memory with a memory strength that reflected the degree of the consistency with the other elements in the current working memory. After completing the entire instructions, the following nine subgoals would be generated and stored in the episodic memory:

- perform "Learn Cricket-Graph"
- perform "Perform Task"
- perform "Plot Data"
- perform "Create Graph"
- perform "Plot Number-of-Accidents As-a-Function-of Month"
- perform "Put Month on X-Axis"
- perform "Put Number-of-Accidents on Y-Axis"
- perform "undefined-action on a document labeled Example-Data"
- perform "Modify Graph"

Goal Selection

After reading the instructions, nine subgoals are stored in episodic memory. In the course of task performance, when the display shown in Figure 2 is provided, LICAI retrieves a subgoal that is consistent with the current display. Note that the overlapping arguments in the representation of subgoals and screen objects are critical determinants of this selection. The labels on screen objects are part of their representations, and they

can be linked to the subgoals. In this case, three goals shown below are likely to be selected:

- perform “Plot Number-of-Accidents As-a-Function-of Month”
- perform “Put Month on X-Axis”
- perform “Put Number-of-Accidents on Y-Axis”

Depending on how these goals were encoded when they were originally generated during the goal formation process, the result of goal selection might differ. Let’s assume that ‘perform “Put Month on X-Axis”’ be selected.

Action Planning

Given the selected goal, ‘perform “Put Month on X-Axis”’, the LICAI model elaborates the display shown by Figure 2 by using knowledge stored in long-term memory. For example, part of the following knowledge would be incorporated: knowledge about scrolling lists, titles of the list, items of the list, such as, ‘the scrolling list items are selectable,’ ‘the scrolling list titles are not selectable,’ etc. , and others concerned with the GUI basics. A network elaborated by these pieces of knowledge is linked and integrated. The correct screen object, ‘Month,’ in the scrolling list labeled by ‘Horizontal (X) Axis’ would be selected as the object to be acted on, and the sequence of actions, first point at Month, then single-click, would be selected. All these selections are done by comprehension of display on the basis of the given subgoal and the use of lots of knowledge retrieved from long-term memory.

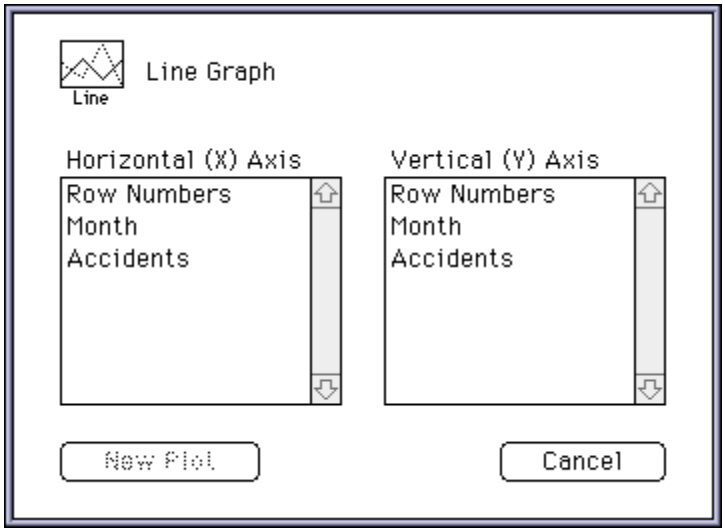

Figure 2. Dialog box that is presented after comprehending the instructions.

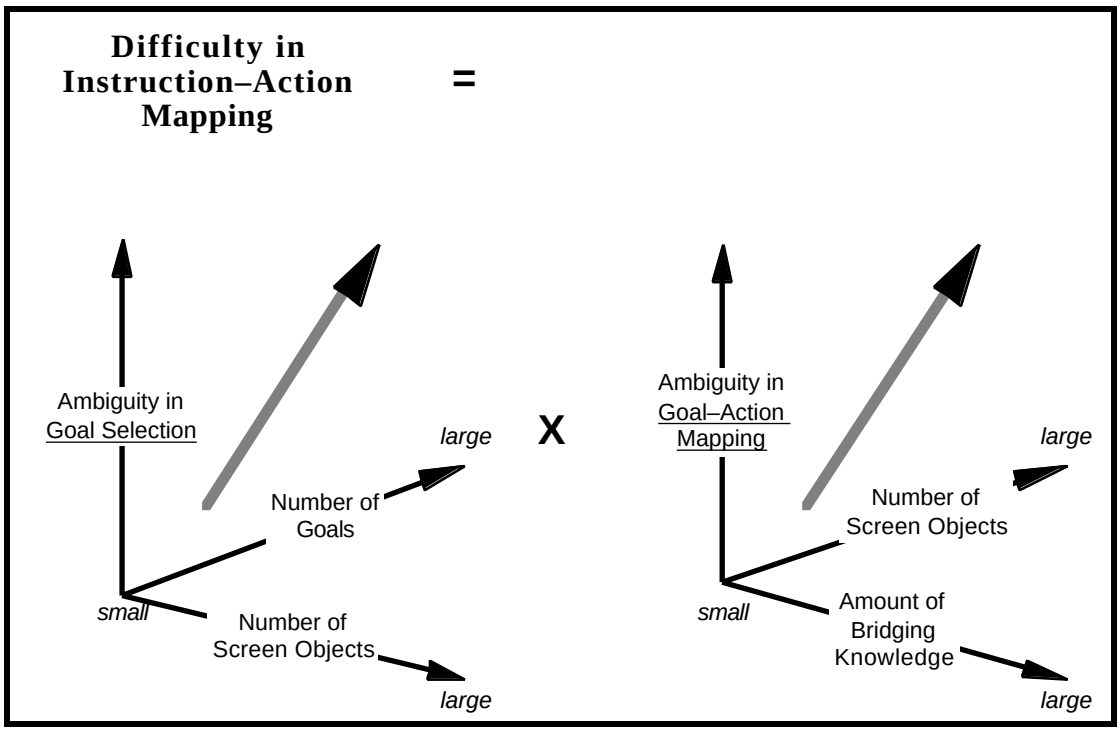

Figure 3. LICAI's predictions on difficulties in mapping instructions onto actions.

POTENTIAL DIFFICULTIES IN MAPPING INSTRUCTIONS ON ACTIONS

The LICAI model describes the underlying mechanism that controls users' instruction mapping onto interface actions. A series of construction–integration cycles depicts the various component processes executed. Mapping from instructions to an action is successful if the goal formation process generates the correct goal, if the goal selection process selects that goal, and if the goal–action mapping process generates the correct action sequence. We predict that this process will be more difficult with longer instructions, more screen objects, and/or an increase in possible actions.

Figure 3 schematically illustrates potential difficulties in mapping instructions on actions that the LICAI model predicts. In the simulation described in the previous section, nine subgoals were extracted from the original instructions. When reading the instructions, the model does not know which subgoals will be relevant or irrelevant, nor the order of their accomplishment. This ambiguity must be resolved by an interface display to be provided. The comprehension process for disambiguating the multiple subgoal problem tends to get difficult as the number of subgoals increases and the number of screen objects that have to arbitrate the problem increases (see the left part of the figure). Likewise, in the goal–action mapping process, the attention cycle tends to fail as the amount of knowledge that is necessary to bridge the goal and the correct object gets larger. This would become worse as the number of screen objects increases (see the right part of the figure).

Understanding how people comprehend instructions and how they map the understandings on interface displays helps us to identify locations where potential

problems might occur, and the strategies to eliminate, or reduce, such problems. I conclude this chapter by suggesting ways to avoid introducing unnecessary difficulties by controlling external sources (i.e., instructions, interface displays):

- Design instruction materials so that the correct goal is generated.
- Design interface displays so that the correct goal is retrieved from the episodic memory generated during the instruction taking process.
- Design interface display so that the correct screen object overlaps with the correct goal.
- Design interface display so that the condition for the correct action is retrieved from long-term memory during the display elaboration process.

REFERENCES

Franzke, M. (1994). *Exploration, acquisition, and retention of skill with display-based systems.* Unpublished doctoral dissertation, Department of Psychology, University of Colorado, Boulder.

Franzke, M. (1995). Turning research into practice: Characteristics of display-based interaction. In *Proceedings of human factors in computing systems CHI '95* (pp. 421–428). New York: ACM.

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

Kintsch, W., and Welsch, D. M. (1991). The construction–integartion model: A framework for studying memory for text. In W. E. Hockley and S. Lewandowsky (Eds.), *Relating theory and data: Essays on human memory* (pp. 367–385). Hillsdale, NJ: Erlbaum.

Kitajima, M., and Polson, P. G. (1995). A comprehension-based model of correct performance and errors in skilled, display-based human–computer interaction. *International Journal of Human–Computer Systems,* **43**, 65–99.

Kitajima, M., and Polson, P. G. (1996). A comprehension-based model of exploration. In *Proceedings of human factors in computing systems CHI '96* (pp. 324–331). New York: ACM.

Kitajima, M., and Polson, P. G. (1997). A comprehension-based model of exploration. *Human–Computer Interaction: Special Issue on Cognitive Architectures in HCI,* **12**, (4).

Hutchins, E. L., Hollan, J. D., and Norman, D. A. (1986). Direct manipulation interfaces. In D. A. Norman and S. W. Draper (Eds.), *User centered system design* (pp. 87-124). Hillsdale, NJ: Erlbaum.

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

Payne, S.J., Squibb, H.R., and Howes, A. (1990). The nature of device models: The yoked state hypothesis and some experiments with text editors. *Human–Computer Interaction*, **5**, 415–444.

Raaijmakers, J. G., and Shiffrin, R. M. (1981). Search of associative memory. *Psychological Review,* **88**, 93–134.

Terwilliger, R. B., and Polson, P. G. (1996). Task elaboration or label following: an empirical study of representation in human-computer interaction. In *Conference companion of human factors in computing systems CHI '96* (pp. 201–202). New York: ACM.