

A Testbed for Intelligent Aiding in Adaptive Interfaces

Jack L. Edwards

AI Management and Development Corporation
206 Keewatin Avenue
Toronto, Ontario, Canada M4P 1Z8
jle@interlog.com

Keith C. Hendy

Defence and Civil Institute of Environmental Medicine
1133 Sheppard Ave., West
North York, Ontario M3M 3B9
Keith_Hendy@dciem.dnd.ca

Abstract

Goal and plan, recognition and generation are fundamental to truly intelligent, adaptive aiding in software applications. The design and development of such capabilities is at the heart of an intelligence testbed being built for the Canadian, Defence and Civil Institute of Environmental Medicine (DCIEM). The purpose of the testbed is to examine and generalize principles of intelligent aiding incorporated into a now mature software tool for workspace layout design called, LOCATE. Generalization is being done with a view to applying those principles to other development efforts at DCIEM and beyond.

Background and Problem

An intelligence testbed is being developed for the Defence and Civil Institute of Environmental Medicine (DCIEM) in Toronto, Ontario, Canada. The purpose of the testbed is to examine and generalize principles of intelligent aiding that have been incorporated into a now mature software tool for workspace layout design. Generalization is being done with a view to applying those principles to other development efforts at DCIEM.

The problem is one of how to identify and extend principles of intelligent aiding across a range of software applications in ways that allow systems to adapt both to the task context and to the users who are performing those tasks. The problem was formulated originally in the context of the development of LOCATE, a specialized Computer-Aided Design (CAD) tool for creating workspace layout designs (Hendy 1984, 1989).

LOCATE provides both a GUI for creating designs and an underlying program for evaluating the communication efficiency among various design configurations. Direct access is provided to the World Wide Web and to world wide design information, with opportunities for designers to share ideas and solve problems.

Although it has been possible to implement clever ways for systems to adapt to their users, this does not represent true adaptiveness but, more often reflects the system designer's ability to anticipate some of the contingencies under which adaptation should occur. The result is the appearance of intelligent adaptation, which ceases to be

effective when users venture into areas the designer could not anticipate or for which it was not clear what provisions should be made.

To have true intelligence and adaptiveness, a system must "understand": 1) what users are doing; 2) the knowledge and attributes of those users; and, 3) its own features and capabilities that may be helpful to users in a variety of circumstances. To extend that understanding, it must have the ability to reason about information contained in its various knowledge sources and be able to use the results of that reasoning to provide effective help.

Work on the LOCATE layout design tool has provided a context in which important steps have been taken to support the adaptiveness of the system, through intelligent aiding to users as they create and analyze their designs.

Fundamental to work on LOCATE and to more recent work on DCIEM's intelligence testbed is the ability:

- to track user interface actions and infer associated goals;
- to interpret those actions and goals within a framework of separable models for task, user, dialogue and system (Edwards 1990; Edwards and Hendy 1992; Edwards, Hendy and Scott 2000; Edwards and Mason 1988);
- to use those interpretations to aid users, especially new users, in learning the software and performing the various tasks for which it was designed.

Recent work is extending the identification of goals and partial plans into a coherent framework for plan recognition and generation. LOCATE attempts to maintain a common plan for itself and a user, as it infers what likely plan is underway. In contrast, when help is to be offered to a user, LOCATE generates a plan for how that help is to be provided.

Adaptiveness emerges as the system interacts with a particular user and builds representations for what that user knows, including: information provided directly by the user; information contained in the help it offers to the user; and, information about the current goals and plans being pursued.

LOCATE's Models

Actions and goals related to the current task are placed into a **Task Model** and displayed in a Task Model Window. As task goals are satisfied or abandoned, they are moved into an Action and Goal History that provides an audit trail of what has been done.

LOCATE maintains a **User Model**, displaying information about the user in its User Model Window. Information in that window reflects LOCATE's beliefs about who the user is, some of his or her preferences and what the user currently knows about LOCATE's features.

LOCATE tracks user activities, in part to determine whether the user is employing the most efficient way of performing some task. If she is not, LOCATE provides advice about how she can make more efficient use of the software.

Finally, LOCATE maintains a **System (Self) Model** as a way of representing its own features and capabilities and how they might help users perform their tasks.

When a user wants help, she can choose "Smart Help" from the Help Menu. A window is displayed that allows her to inquire how some activity can be performed, e.g., "how to print." LOCATE searches its knowledge base and returns a set of possibilities in the form of goals related to that activity and plans for how to achieve each. For example, the goals returned for "how to print" include:

- how to print a design;
- how to print a cost function;
- how to print a color cost display.

The user chooses the appropriate goal and LOCATE displays a plan for accomplishing it in another portion of the window.

At the same time help is being provided, LOCATE's System Model is updated and information is displayed in a System Model Window. That information includes the user's query and any conclusions about how LOCATE has determined it can help the user perform the requested activity.

In future, LOCATE will offer to execute part or all of a plan for achieving a user's goal. By examining the user's history, LOCATE will be able to determine if a user always, or most often, asks it to execute a given plan, and, if so, will suggest that it be allowed to perform that as a default function.

Feedback for Adaptation

LOCATE acquires information (feedback) about the current task and user in three ways:

- from information provided directly to the system by the user;
- from help and other information provided to the user by the system;
- by tracking user actions at the interface and interpreting them in terms of the user's ongoing task.

These three sources of information are the ways LOCATE "understands" the knowledge and capabilities a user possesses and what the user is doing at the moment. That knowledge is used to adapt LOCATE's help to the current context of user knowledge and action.

LOCATE initiates help under a variety of circumstances including, for example, when it detects what are called, "non-responsive" actions.

Non-responsive actions are ones that produce no functional response by the system other than perhaps an audio signal to indicate the fact.

A number of actions qualify as non-responsive actions including: clicking on a ruler in the LOCATE Design Window; clicking on a dimmed button or menu item; double-clicking on a communication link between two workstations; double-clicking a workstation's rotation arrow; and, attempting to resize a Source/Receiver (S/R) Node.

A number of possible implications to these non-responsive actions give direction to the kind of adaptive aiding a system might offer. Two such implications are that:

- the user has a clear goal in mind and is guessing at the interface action that will achieve that goal;
- the user is responding to the presence of an unfamiliar object by using standard action methods on the object to see their effects.

In both cases, LOCATE first informs the user that his action produces no task-related response by the system. Next, it allows the user to identify the goal he had in mind when he performed that action, by displaying a selectable set of possibilities. Finally, it presents a plan for achieving that goal if it and its plan recipe are stored in LOCATE's plan library.

Other goals presented as possibilities for the user's intended goal serve as opportunities for incidental learning, if the user chooses to select them and examine their associated plans.

As a consequence of providing help of this kind, LOCATE is able to infer that the user now knows:

- that this action produces no task-related response by the system;
- how to achieve his intended goal, i.e., the goal he had in mind when he performed the non-responsive action.

Adaptive aspects of this kind of aiding follow from LOCATE's ability to determine whether a particular user received help of this kind on a prior occasion.

After providing help about a specific non-responsive action, if the user performs that action again at some later time, LOCATE will assume he knows it will not produce any task-related response. Under that circumstance, a standard audio beep should be a sufficient reminder.

If the user performs the same action on several subsequent occasions, LOCATE will see it as a "contradictory action," that is, an action that contradicts

LOCATE's beliefs about what the user should know. In that case, help is offered with the goal of clarifying the apparent contradiction.

Feedback Effects on System Performance

Feedback obtained directly from the user is currently quite limited in LOCATE and has minimal effects on system performance. Such information is obtained when the user tells LOCATE who he is, what preferences he has with respect to features of the software and whether he wishes LOCATE to track his activities and offer help.

The decision to provide help to a user, the specific help to be provided and how that help is realized in the interface, all have minimal effects on system performance. High priority help, resulting in the display of an Alert, is the most intrusive kind of help and appears most frequently with new users who are learning the software.

When used, it influences overall system performance because of the interruption of the flow of user activity. The trade-off for this decrement in overall performance is that such help is believed to result in more effective use of LOCATE in the long run and in fewer errors in task performance.

Moderate priority help results in unobtrusive displays of graphical indicators ("light bulbs") that some help is in the offing. Low priority help is only offered when its priority changes through circumstances that add cumulatively to that priority. Neither of these types of help currently impact system performance to any significant degree.

Project Status

Development has progressed to the point where LOCATE is a mature tool for workspace layout design and analysis (see for example, Hendy et al. 2000). Intelligent aiding has been a focus from the beginning and examples have been incorporated within LOCATE as work on its fundamental features has progressed.

The intelligence testbed work has a more recent history. Key to these efforts is building adaptation systems through intelligent aiding, which includes, in part, investigating how such aiding can be decomposed from the software it supports and used with other applications.

Goal identification, plan recognition and plan generation are playing important roles in the testbed work. Investigating and implementing goals and plans along with other information-processing ideas, and with ideas from theories such as Perceptual Control Theory (Powers 1973; Powers and Robertson 1990) form the heart of the testbed work.

Acknowledgments. The testbed work described in this report is being conducted by Artificial Intelligence Management and Development Corporation, under contract to the Canadian Department of National Defence (Contract No. W7711-9-7564). Previous work on the

testbed and on the design and development of the LOCATE Workspace Layout Tool was conducted under several prior contracts.

References

- Broadbent, G. (1988). *Design in architecture*. London: David Fulton Publishers.
- Edwards, J. L. (1990). Proposed architecture for SDBMS-2. Report prepared for the Defence and Civil Institute of Environmental Medicine (DCIEM) by Artificial Intelligence Management and Development Corporation. Toronto, Ontario, Canada.
- Edwards, J. L., & Hendy, K. (1992). Development and validation of user models in an air traffic control simulation. Paper presented at the Second International Workshop on User Modeling, International Conference and Research Center for Computer Science (IBFI), Dagstuhl Castle, Germany, August 10-13, 1992.
- Edwards, J. L., Hendy, K., & Scott (2000). A Testbed for Adaptive Aiding Using the LOCATE Workspace Layout Tool. Paper submitted for a Special Issue on Deployed Systems to the Journal of User Modeling and User Adapted Interaction.
- Edwards, J. L., & Mason, J. A. (1988). Toward intelligent dialogue with ISIS. *International Journal of Man-Machine Studies*, 28, 309-342.
- Hendy, K. C. (1984). 'Locate': A program for computer-aided workspace layout. Master's Thesis, Department of Electrical Engineering, Monash University, Clayton, Victoria, Australia.
- Hendy, K. C. (1989). A Model for Human-Machine-Human Interaction in Workspace Layout Problems. *Human Factors*, 31(5), 593-610.
- Hendy, K. C., Edwards, J. L., Beevis, D., & Hamburger, T. (2000). Analyzing advanced concepts for operations room layouts. *Proceedings of the IEA/HFES 2000 Congress. San Diego, CA, July 30-August 4, 2000.*
- Powers, W. T. (1973). *Behavior: The control of perception*. New York, NY.: Aldine De Gruyter.
- Powers, W.T., & Robertson, R. J. (Eds.) (1990). *Introduction to modern psychology: The control-theory view*. Gravel Switch, KY.: The Control Systems Group Inc..
- Simon, H. (1981). *The sciences of the artificial*. 2nd edition. Cambridge, MA: MIT Press.