Rapid Prototyping of Self-Adaptive Interfaces with the L-CID Model

Angel R. Puerta[1]

Medical Computer Science Group

Knowledge Systems Laboratory

Stanford University

puerta@sumex-aim.stanford.edu

Position Paper Submitted to:

Intelligent Multimedia Interfaces Workshop

AAAI-91

Introduction

While we are by now familiar with the concept of adaptive interfaces, little attention has been given to a much more challenging type of systems: self-adaptive interfaces. This type of interfaces have the capability of improving their performance over time. In the same manner that the adaptation process requires some type of feedback from the user, the self-adaptation process involves obtaining feedback from the adaptation process itself to be able to modify it according to an appropriate measure of success.

The quest for adaptation in interfaces is based on the fact that it is impossible for an interface designer to anticipate the needs and requirements of each possible user of the system. Thus, postponing some interaction decisions until run-time allows the designer to create much more general interfaces. When the pro-

[1] Research conducted while the author was at the University of South Carolina, Columbia, South Carolina.

cess of adaptation is itself flawed or inadequate, however, the interface designer is once again faced with the task of redesigning the interface to correct the deficiencies. On the other hand, if the interface has the capability of modifying its own adaptation method, then the designer is placed at a higher level of abstraction and can produce interfaces considerably more general (useful for a larger number of users), and tailorable (adapting to each user more efficiently as time goes on). Self-adaptation is particularly important in multimedia interfaces where adaptation is complicated by the multiple means of communication available.

The author takes the view of self-adaptive interfaces being knowledge-based and of self-adaptation as being performed through the use of machine learning techniques. Earlier work [7] discussed the impact of machine learning capabilities in the design and operation of the interface, and taxonomized the possible learning objectives of a user interface. Admittedly, one serious roadblock in incorporating machine learning into user interfaces is that there is no solid background on what learning techniques or paradigms are more effective for a particular user interface.

With this problem in mind, the author developed a model of a self-adaptive interface [8] which gives to interface developers rapid-prototyping capabilities to experiment with different learning techniques and strategies. There are three main requirements for this model. First, it must be mappable easily to other general, established models for intelligent interfaces so results obtained can be applied to interfaces developed from those models. Second, it must not prescribe a learning strategy, a technique, or a learning objective. Finally, it must be flexible enough to accommodate a number of learning strategies and techniques and, at the same time, exhibit a degree of completeness that permits experimentation on a majority of identifiable learning objectives for user interfaces.

Starting from Card's triple agent model of intelligent interfaces [1] and using the blackboard model [4] as an implementation foundation, a blackboard framework for experimentation with self-adaptation was developed. L-CID (Learner for Computer Interface Development) is shown in Figure 1. It is a multilayer model where the lower layer, *Babieca, performs the interface functionality, thus, it is the horse of the model. The upper layer, Tizona, can learn about the lower layer process, detect deficiencies, and make* *appropriate changes to Babieca's knowledge base, therefore, it is the sword of the model. The four major knowledge sources in Babieca correspond to the four interface functions of translation (Discourse Manager), user modeling (Modeling Expert), Task Modeling (Task and Domain Advisor), and help (Intelligent Assistant). The four knowledge sources in Tizona correspond to the four learning processes of training (Training Expert), performing (Performance Element), critiquing (Evaluation Expert), and modifying a knowledge base (Representation Expert). The Performance Element is in fact, in the multilayer view, the lower interface layer, Babieca.*

In the following sections, an example of the use of L-CID to experiment with a learning technique for modality selection is described.

Modality Selection with L-CID

One of the challenges present in building interfaces that can communicate in various modes and media, such as visual and auditory media, or textual and graphical modes, is the need to properly match the information to be presented to the user with the most appropriate channel of presentation. The problem of modality selection and effectiveness of presentation has been studied by psychologists as part of research on human factors [10]. In an intelligent user interface system able to exploit the multiple channels of communications available to humans, the task of modality selection can become very complex. The problem is compounded by the fact that for a specific presentation mode there are a number of techniques which can be used to present the information in that single mode (e.g., pie charts vs. bar charts). A possible course of action is to add heuristics to the system to deal with modality selection decisions [6]. Given the fact that modality selection varies widely between users, and even between tasks for one user, the number of heuristics needed could quickly grow to unmanageable levels.

Consequently, it was determined that modality selection is a process that could benefit by being a learning objective in a self-adaptive system. Defining this learning objective in the context of the L-CID model, yielded the following problem statement:

Given: A set of information types, a set of presentation techniques, and a set of multiple-valued user, computer, environment, and task features.

Find: A complete mapping of each information type to one or more modalities under every valid combination of values of user, computer, environment, and task features.

The learning technique selected to solve this problem assumed that the system would start with no knowledge on how to map an information type to a presentation technique, but knowing, for a particular information type, what presentation techniques were available in the interface. When no knowledge about prior interaction exists, the technique randomly selects one available presentation technique. Random choices that prove to be correct (as determined by the Evaluation Expert knowledge source) are used in future occurrences while incorrect choices are discarded for future selections. After a certain period of time, the system should have a complete mapping of information types to presentation techniques. This learning technique, albeit naive, serves to illustrate the use of L-CID in quickly determining that technique’s deficiencies, and making appropriate modifications to develop a more sophisticated technique.

Testing the Learning Technique

The simulation of the learning technique in L-CID was implemented in GBB1 [2], a general-purpose blackboard development system. As a framework, L-CID is not constrained to this specific implementation. To better illustrate the process of learning modality selection, a running example is shown in Figure 2. This particular example involves a target scenario that must be learned by the system. The scenario consists of a single feature for each of the models in the Modeling Expert knowledge source of L-CID: Username (user model), Computer-Type (computer model), and Location (environment model); as well as a single feature for the Task and Domain Advisor knowledge source: task-type (Task Expert). Note that some knowledge sources in L-CID were not deemed relevant for this problem, thus, they were left out of the implementation.The scenario also consists of an *information type and a correct modality value to be assigned for this specific combination of feature values. The correct modality assignment is only known to the Evaluation Expert who uses it to determine whether the actions of the Modality Selector are appropriate. The Modality Selector has knowledge about a set of valid mappings that can be used with* *the specified Information type.*

As seen in the example, in the first action cycle the system has no prior knowledge of which modality must be selected, thus it randomly selects one value from the set of valid mappings. Since this value is determined to be incorrect, the learning system drops it from the set of valid mappings. Once the system randomly selects the correct value, future responses will apply this response directly.

A series of tests were conducted to observe the limitations of this technique [9]. The tests varied the number of target scenarios that the system had to deal with (100-200), the number of features (up to 20), and the number of entries in the valid mappings set (3, 5, and 7). Let us focus on the comparison of two of the results obtained. Figures 3 and 4, show that the learning system had more difficulty in dealing with increases in the number of entries in the valid mappings set than with increases in the number of target scenarios. Therefore, it was concluded that the learning technique would benefit most by reducing the number of times that a random selection must be made from the valid mappings set.

A variation of the technique was devised where the system generalizes the response given by the Modality Selector. Instead of requiring this knowledge source to exactly match the current scenario with a prior scenario for which a correct response was given, the Modality Selector matches the current scenario to any prior scenario that showed a single difference in the feature values with the current scenario. The improvement in the learning technique is showed in Figure 5. Using this result, an extension to the original technique was developed that avoided making random choices as long as at least one feature value from the current scenario could be matched to a prior, correct, scenario.

Discussion

Recent years have seen a marked interest in intelligent interfaces on the part of researchers. However, these interfaces continue to fail to materialize as working systems. It is the author's point of view, that full-blown intelligent interfaces are not viable at this time. Instead, what we will be seeing more and more is the use of artificial intelligence techniques to aid in the completion of very specific tasks in already available interfaces. In addition, the dynamic nature of the human-computer interaction process determines that it is tremendously difficult to anticipate the needs and requirements of a single user performing a variety of tasks, let alone doing so for a group of users. Therefore, more of the interface decision process must be switched to the interface itself in the form of self-adaptation. There are several examples of the use of self-adaptation to aid in specific interface tasks [3,5]. The challenge in self-adaptive interfaces is to find appropriate learning strategies and techniques that produce interfaces that give the user a feeling of control of the interaction process, do not interfere with the completion of the user's task, and present a stable interaction environment to the user.

L-CID allows interface developers the luxury of experimenting with learning techniques for specific tasks without having to make early commitments in their designs that can later on prove costly. This ability should encourage developers to consider artificial-intelligence solutions to some interface problems, and may increase the rate at which these systems are being produced. On the flip side, L-CID is a model rather than a tool, therefore, its use requires a deep understanding of the conceptual model on which it is based. Regardless, given the increased availability of more powerful and versatile interfaces that can communicate in various media, it can intuitively be seen that self-adaptation is a clear candidate to produce the interface of the future: designed for a general group of users, but completely adaptable to the actual users who utilize it.

References

[1] S. K. Card, **Human FActors and Artificial Intelligence, in Intelligent Interfaces Theory, Research, and Design, ed. by P. A. Hancock and M. H. Chignell, North Holland, New York, New York, 1989, pp. 27-41.**

[2] D. D. Corkill, K Q. Gallagher, and K. E. Murray, **GBB: A Generic Blackboard Development System, in Blackboard Systems, ed. by R. Engelmore and T. Morgan, Addison-Wesley, Wokingham, England, 1988, pp. 503-517.**

[3] A. Cypher, **EAGER: Programming Repetitive Tasks by Example, Proceedings of CHI '91, to be released, May 1991.**

[4] R. Engelmore and T. Morgan, **Blackboard Systems, Addison-Wesley, Wokingham, England, 1988.**

[5] R. A. Gargan et al., **Multimodal Response Planning: An Adaptive Rule-Based Approach, Proceedings of CHI 1988, pp. 229-234.**

[6] D. L. Maulsby and I. H. Witten, **Inducing Programs in a Direct-Manipulation Environment, Proceedings of CHI 89, pp. 57-62.**

[7] A. R. Puerta and R. D. Bonnell, **Machine Learning in User Interfaces: Objectives and Strategies, Proceedings of the XV Latin-American Conference on Informatics, Santiago, Chile, July 1989.**

[8] A. R. Puerta and R. D. Bonnell, **A Blackboard Model for a Self-Improving, Intelligent User Interface, Proceedings of the Annual Conference of the International Association of Knowledge Engineers, College PArk, Maryland, June 1989.**

[9] A. R. Puerta, **L-CID: A Blackboard Framework to Experiment with Self-Adaptation in Intelligent Interfaces, Doctoral Dissertation, USCMI Report Number 90-esl-6, Center for Machine Intelligence, University of South Carolina, Columbia, South Carolina, July 1990.**

[10] C. Wickens, D. L. Sandry, and M. Vidulich, **Compatibility and Resource Competition between Modalities of Input, Central Processing, and Output, Human Factors, Volume 25, Number 2, 1983, pp. 227-248.**