AN ADAPTABLE USER INTERFACE FOR TOLERATING NAIVE USERS' ERROR BEHAVIOR: A PROTOTYPE DESIGN AND AN EMPIRICAL STUDY

Max M. North and Sarah M. North

Human-Computer Interaction Group
Clark Atlanta University, Atlanta, Georgia 30314, (404) 880-8242
max@cc.gatech.edu, snorth@cau.auc.edu

ABSTRACT

This short paper briefly discusses a hybrid adaptive interface, called AUI, for tolerating naive user's error behavior. The paper demonstrates how the fusion of different adaptive user interface techniques provide a powerful hybrid interface. The prototype AUI adapts itself to each individual user by observing their actions and receiving direct feedback. The AUI requires minimal background knowledge at first, but gradually this knowledge becomes more sophisticated and more reliable. The AUI collaborates with the user in achieving the user's goals. An application of AUI to Unix operating system commands is investigated. The preliminary results of the pilot studies suggest subjects who utilize the AUI perform at a higher level ($t=6.85$, $df=78$, $p<0.05$) than their counterparts who do not utilize the AUI.

AUI DESCRIPTION

The fusion of different adaptive interface techniques (i.e., adaptive scheduling and pattern recognition, learning agents, etc.) can provide a powerful hybrid adaptive interface for many domain-specific problems such as tolerating the users' error. Specifically, the AUI observes all of the users' error behavior and attempts to resolve the errors. When the AUI cannot predicate the intended action, it initiates an interactive dialogue with the user. The knowledge bases of the AUI are modified based on the direct feedback or recommendation from the user. Specifically, the AUI can recognize and classify types of user's error behavior such as perceptual, cognitive, and motor errors. By analyzing the classification of error types committed by users, AUI develops strategies to minimize the occurrence and effects of errors.

Although the AUI approach seems closely related to the characteristics of the learning interface agent, there are major differences between the two approaches: (i) the architecture of the AUI is the product of the fusion of several adaptive interface techniques, (ii) the AUI has a forgetting algorithm that discards unimportant and unnecessary knowledge and optimizes itself regularly, (iii) the AUI is based on a sound and empirical theory of the client-centered approach, (iv) the AUI incorporates several small-scale knowledge bases that are constantly regulated and updated to keep the most relevant knowledge rather than keeping it all which causes tremendous computing time, (v) the AUI classifies and understands user's unique error behavior, and (vi) the AUI adapts to individual error behavior.

The AUI utilizes several small scale knowledge bases such as user models, commands, errors, tasks, and adaptive knowledge. The major components of the AUI are: interaction dialogue, front-end parser, pattern matching/learning, and the inference engine. The AUI uses an object-oriented hierarchical approach where the components of a system (subject domain) are grouped into a number of subsystems (topics) based on functional description. Although the AUI architecture seems fairly simple, it provides a powerful and practical interface to assist users with certain types of errors that can improve the user's performance and attitude tremendously.

CONCLUSION

The AUI is an ongoing research project at the Clark Atlanta University by the Human-Computer Interaction Group. The preliminary results of the pilot case studies so far are very encouraging, and users' feedback has been positive. Many of today's Unix systems use graphical interfaces. Two examples are *Motif*™, from the Open Software Foundation (OSF), and *OpenLook*™, created by AT&T™ and endorsed by Unix International. Whether it communicates as text or graphical interface, every interactive system has a command language. Consequently, the adaptable user interface may be applied to many different command language based systems (e.g., GUIs). Finally, this research produces several recommendations and guidelines that may reduce the learning time of Unix commands for novice users and increase the efficiency of the experts.

ACKNOWLEDGEMENT

This research project is sponsored by Boeing Computer Services and partially supported by the U. S. Army Center of Excellence in Information Science under contract number DAAL03-92-6-0377. The views contained in this document are those of the authors and should not be interpreted as representing the official policies of the U. S. Government, either expressed or implied. In addition, this research is aided by Grant-in-Aid of Research from Sigma Xi, The Scientific Research Society, ACM/SIGCHI.