Automatic Interface Generation and Future User Interface Tools

Jeffrey Nichols and Andrew Faulring
Human-Computer Interaction Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213 USA
{ jeffreyn, faulring }@cs.cmu.edu

INTRODUCTION

Automatic interface generation has been used to separate the user interface from the application logic. Although this approach was not successful for general application interfaces, we believe it can be a valuable tool in certain situations. For example, the author of an intelligent agent might want to focus on the intelligence aspect and delegate the details of the user interface to another piece of software. Automatic interface generation also allows specific information about the user and the current situation to be incorporated into the design of the user interface. For example, a user interface could be displayed on whatever device the user has available or the interface could use familiar elements that the user has seen recently.

There has been a rich history of research in the area of automatic generation, mainly in the model-based user interface community. Model-based systems attempt to formally describe the tasks, data, and users that an application will have, and then use these formal models to guide the generation of the user interface. Some systems automatically design the user interface, such as UIDE [2] and Jade [8], and others provide design assistance to a human, such as Trident [9] and Mobi-D [6]. Despite a lot of research, model-based user interface tools have not become common, in part because building models is an abstract process and better results are often achievable by a human designer in less time [3]. There have been successes in limited domains such as dialog box design [2, 8] and remote controls [4], and we believe that model-based techniques show promise provided that the scope of the generated interfaces is kept manageable.

Our research is applying model-based concepts and automatic interface generation in two projects. The Personal Universal Controller (PUC) [4] project is applying model-based techniques to automatically generate remote control interfaces for all of the computerized appliances in the environment and is exploring how the generated interfaces can be automatically customized to both the control device and the user. The Rich Human-Agent Interaction (RHAI, pronounced “ray”) project [1] is building a system that allows intelligent agents to communicate with the user through a dialog manager and a general framework for modeling questions. The dialog manager decides the best time and place to ask the question, and presents the question through an interface that is automatically generated from the description of the question.

PERSONAL UNIVERSAL CONTROLLER

The PUC system allows users to use the handheld devices they are already carrying, such as PDAs and mobile phones, to remotely control home and office appliances, such as televisions, microwave ovens, and copy machines. When a user requests an interface to control an appliance, the user’s device downloads a functional model from the appliance and uses that model to automatically generate an interface. To support the system, we have designed a usable, concise XML-based language for modeling the appliance’s functions. We have also developed rules for generating user interfaces from our language on several different devices, including PocketPCs, Microsoft Smartphones, and desktop computers.

We are currently working on two new features of our system that build interfaces that are customized to the user’s situation. One of the new features will ensure that new user interfaces are consistent with previous interfaces for a similar appliance. This means that the interface for a VCR that you have never used before will look similar to the interface that you already use on your home VCR. The other feature will allow a PUC device to create a single unified interface for a set of connected appliances, such as a home theater or presentation room. The unified interface will be organized by task instead of by appliance and may include automatically generated cross-platform macros. For example, the home theater might have a “Play DVD” macro that turns on the TV and DVD player, turns off the VCR, sets the appropriate input source, and starts playing the DVD.

RICH HUMAN-AGENT INTERACTION

The RHAI project is exploring how to build user interfaces for intelligent agents that help users with complex, routine tasks. The agents that we work with can, for example, schedule multi-person meetings, update project websites, and find and reserve conference rooms. Our agents can interpret the user’s e-mail, at which point they may take some action, such as negotiating possible times for a meeting or determining a set of potential changes to a web page. Before the agent makes any permanent changes, it can ask the user to confirm that its proposed action is correct. We have found that agents often propose the correct action, but they may suggest incorrect or undesirable parameters. During confirmation, the user may change any of the parameters or suggest a different action. The agent receives feedback from the user's response so that it may learn how to improve its future performance.

The model-based aspect of this project is in the communication between intelligent agents and the user interface. We are designing a high-level XML-based language that allows an agent to express its questions and proposed actions without mentioning any user interface details. The language will include things that might affect the appearance of the user interface, such as data types, confidence values and origin information (e.g. the natural language of the original e-mail, stored agent knowledge, etc.) for parameters of proposed actions. One of the interesting challenges of this language is specifying the relationships between different proposed action parameters. If a user changes the value of a parameter, then an alternate value for another parameter may be more appropriate. In some cases it will reasonable to express these relationships in the language, but in other cases the agent must be kept in the loop so that it can modify the parameters directly.

From this model, RHAI automatically generates high-quality user interfaces that allow the user to confirm or change an agent's proposed actions. We use a rule-based approach to generate interfaces from the models. In the future, we are planning to develop methods for integrating the generated interfaces with running applications, because it often makes sense to ask users questions in the context of existing applications with which they are familiar.

GENERAL CHALLENGES FOR UI TOOLS

Both the PUC and RHAI projects suggest some common challenges that must be addressed for future user interface tools that incorporate automatic generation. The most important is finding more domains where automatic generation can be applied successfully. Rather than seek general solutions like the model-based community has in the past, we believe that it is important to explore a large number of specific domains and tailor the automatic generation approach to each domain. Working in a specific domain allows the system to be optimized and may lead to new techniques that are important for that domain. For example, our work in the appliance domain showed that many interfaces had special renderings for similar functional groups, which led to the Smart Templates [5] technique.

A related challenge is to improve modeling languages, which can be made easier to author within a specific domain as we have shown with the PUC's specification language. The number of new XML-based languages, such as XIML, UIML and USIXML, is growing significantly however, and work will be needed to understand the advantages and disadvantages of each language.

Model building techniques could also benefit from more tool support, particularly for making the process less abstract. The UI Pilot [7] is a recent example of integrating model building with an existing design process to address issues such as abstractness.

Another challenge is finding novel ways of using models from multiple sources to create customized user interfaces. The PUC project is investigating this idea through its consistency and multi-appliance work. To ensure consistency, the PUC combines old appliance and presentation models with a new appliance model to generate an interface that is consistent with those the user has seen in the past. When multiple appliances are being used to complete a larger task, the PUC combines many appliance models to generate an aggregate user interface. There is an opportunity for our agent work to use this idea as well, perhaps by combining the presentation model of an existing application with a presentation model supplied by an intelligent agent.

CONCLUSIONS

Our projects are investigating ways that model-based concepts might be applied to future user interface systems. We believe that automatic interface generation is important for supporting interfaces on multiple devices and incorporating situational information into interfaces. Some challenges are finding specific domains where automatic generation is practical, improving modeling techniques, and exploring novel ways of combining models to improve usability.

REFERENCES

1. Faulring, A. and Myers, B.A. "Enabling Rich Human-Agent Interaction for a Calendar Scheduling Agent," in *Submitted to CHI'2005.*
2. Kim, W.C. and Foley, J.D. "Providing High-level Control and Expert Assistance in the User Interface Presentation Design," in *Proceedings of INTERCHI'93.* Amsterdam, The Netherlands: pp. 430-437.
3. Myers, B.A., Hudson, S.E., and Pausch, R., "Past, Present and Future of User Interface Software Tools." *ACM Transactions on Computer Human Interaction*, 2000. **7**(1): pp. 3-28.
4. Nichols, J., Myers, B.A., Higgins, M., Hughes, J., Harris, T.K., Rosenfeld, R., Pignol, M. "Generating Remote Control Interfaces for Complex Appliances," in *UIST 2002.* Paris, France: pp. 161-170.
5. Nichols, J., Myers, B.A., Litwack, K. "Improving Automatic Interface Generation with Smart Templates," in *Intelligent User Interfaces.* 2004. Funchal, Portugal: pp. 286-288.
6. Puerta, A.R., "A Model-Based Interface Development Environment." *IEEE Software*, 1997. **14**(4): pp. 41-47.
7. Puerta, A.R., Micheletti, M., and Mak, A. "The UI Pilot: A Model-Based Tool to Guide Early Interface Design," in *Intelligent User Interfaces.* 2005. San Diego, CA:
8. Vander Zanden, B. and Myers, B.A. "Automatic, Look-and-Feel Independent Dialog Creation for Graphical User Interfaces," in *Proceedings of SIGCHI'90.* Seattle, WA: pp. 27-34.
9. Vanderdonckt, J. "Knowledge-Based Systems for Automated User Interface Generation: the TRIDENT Experience," in *Technical Report RP-95-010.* 1995. Namur: Facultes Universitaires Notre-Dame de la Paix, Institut d' Informatique