TIMM: The Task-Interface Model Mapper

Jacob Eisenstein
Section on Medical Informatics
Stanford University
Stanford CA 94305
jacob1@leland.stanford.edu

Angel Puerta
Section on Medical Informatics
Stanford University
Stanford CA 94305
puerta@smi.stanford.edu

ABSTRACT
We introduce TIMM, an interface-generation tool which bridges the gap between the user-task model and the concrete interface. As an interface-development tool with knowledge of the user-task tree, TIMM provides decision support to the application developer. TIMM formalizes many decisions which would otherwise be made in an ad hoc fashion by various members of the application development team, allowing the interface developer to act with a broader perspective on the interface in mind. Our model-based approach allows global changes to be made during or after the interface design process which would otherwise require tedious step-by-step work. TIMM allows the interface developer to establish three kinds of relations between the user-task model and presentation elements: prioritized links between task objects and presentation elements, dialog-specific styles, and global interface strategies.

Keywords
Model-based interface development, user-interface development tools, task-based design

INTRODUCTION
A number of commercially marketed applications allow interface designers to lay out presentation elements on a canvas to create an interface. These applications, while intuitive and flexible, generally lack a larger perspective on the interface designer's plans; they have no knowledge of a user-task model. On the other hand, tools have been developed to elicit a task model from the user, so as to aid the interface developer by providing a more formalized description of what the user wants to accomplish. In most instances, however, these task-based tools do not include facilities that allow designers to relate these user-task models to actual presentation elements on an interface.

In this paper, we introduce TIMM—the Task-Interface Model Mapper—a tool designed to link these two sides of the interface development process. TIMM aims to provide a middle layer of abstraction between highly abstract user-task models and more concrete presentation elements, such as windows and widgets. Interface designers use TIMM to interactively map classes of task objects to types of widgets, and to partition task steps among windows in an interface. Designers can also use TIMM to establish general design styles and preferences for the interface. By studying how users interact with TIMM, we can (1) explore what are the important relationships between user-task models and interfaces, and (2) implement effective interaction techniques that allow interface designers to manipulate those relationships effectively during interface design.

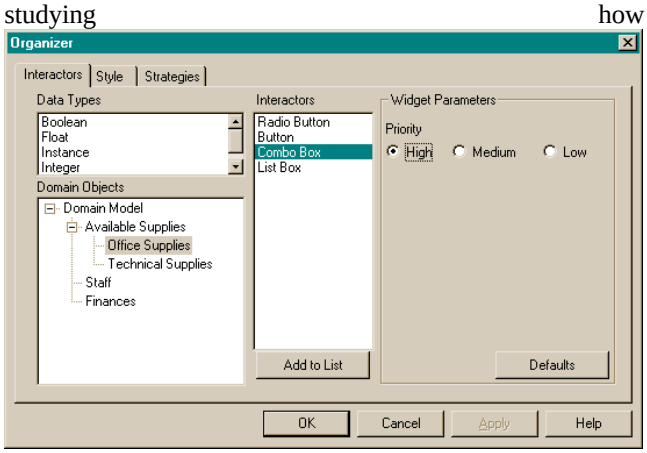

Fig 1. The interactors page of TIMM

TIMM
TIMM is a property sheet with three pages: Interactors, Styles, and Strategies. These pages are independent, self-contained bridges between the user-task model and the interface. TIMM is integrated into of a larger, model-based interface development environment called MOBI-D [1].

The Interactors page (see figure 1) allows the interface developer to establish relations between task objects and interactors—sliders, radio buttons, and the like. Each relation has an attribute called “priority,” which can be set to “high”, “medium”, or “low.” Priority reflects how likely the user is to represent a task object with a particular interactor. For example, a task object representing time might be related to a radio button with priority “low”, while it might be related to a spinner with priority “medium.” A dialog design tool can use the information

gathered on this page to create a list of interactors for each task object, ordered by priority. The Interactors page allows the interface developer to select all the interactors at once, with the task tree in view. We feel this is an advantage when compared to tools that require the interface designer to select interactors while in the process of doing dialog layout, and without a structured connection to a user-task model.

The interface designer also has the option of establishing relationships between interactors and types of task objects, such as integers or strings, allowing quicker and more general decisions to be made. Future usability studies should yield a knowledge base of type-interactor relationships to serve as a default.

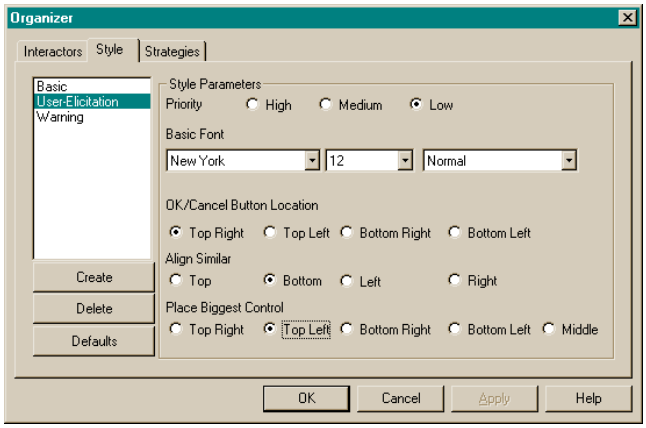

Figure 2. The styles page of TIMM.

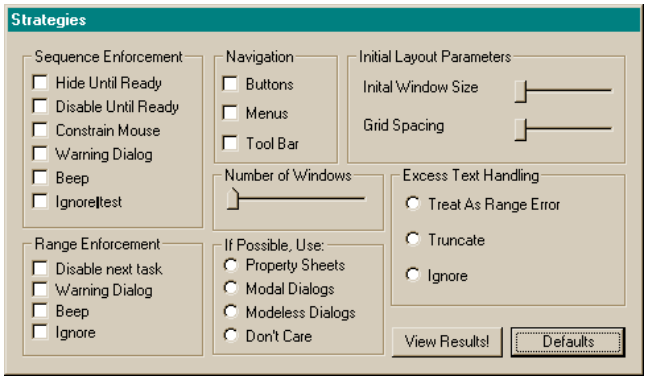

Figure 3. The strategies page of TIMM

The Styles page (see figure 2) is similar to the "styles" dialog made familiar by the modern word processor. The interface designer uses this page to establish a library of styles. These styles can then be applied consistently to a single task or a group of subtasks in a user-task model. Styles currently include the following information: font, font size, font style, location of the "OK" and "Cancel" buttons, alignment of similar controls, and when to put boxes around controls. Usability studies may lead us to include additional information in future versions. Styles can be applied to individual dialogs within the interface. For example, an application might have one style for warning dialogs and another for the help page. Styles also have the priority attribute, which is intended to help the user decide which style to apply first. TIMM includes three default styles. By formalizing style in a single, coherent model, the Styles page offers several advantages when compared with conventional approaches. Broad stylistic changes can be made without going back to change each individual dialog, the possibility of accidentally generating dialogs in a slightly incorrect style is eliminated, and the declarative model allows multiple interface designers to work on the same project and coordinate stylistic decisions.

The Strategies page (see figure 3) allows the interface developer to make very general settings that affect the entire interface. These settings include information regarding the number of windows, methods for navigating within the interface, task sequencing, initial layout parameters, and error handling. Again, usability studies will allow us to establish a knowledge base of default settings, and may lead us to establish different controls on this page. The Strategies page supports the interface developer in making interface-related decisions that might otherwise be left to the application developer, and allows several global decisions to be made at once, rather than in an ad hoc fashion throughout the interface design process.

RELATED WORK

TIMM is closest aligned to the work in model-based interface development [1]. There have been some efforts in this area that attempt to map user tasks to interface elements, notably ADEPT [2]. However, those approaches are usually algorithmic and not interactive and explicit as in TIMM.

CONCLUSIONS

TIMM offers several advantages when compared with traditional approaches to user-interface design. Most importantly, it creates a structured approach to relate user tasks with concrete interface elements. It also formalizes and brings together many conceptually related interface design decisions, which might otherwise have been made at different times by different people. TIMM's model-based approach allows broad decisions to be made and then changed easily and quickly. Its knowledge of the task tree allows TIMM to provide decision support, which can lead to a more efficient and structured interface development process.

ACKNOWLEDGMENTS

This work is supported by DARPA under contract N66001-96-C-8525.

REFERENCES

1. Puerta, A. R. A Model-Based Interface Development Environment. *IEEE Software*, (14) 4, July/August 1997, pp. 40-47.

2. Wilson, S. and Johnson, P. *Beyond Hacking: A*

Model-Based Approach To User Interface Design, in Proc. of *HCI'93*. 1993.