From Models to Interactive Systems Tool Support and XIML

Peter Forbrig, Anke Dittmar, Daniel Reichart, Daniel Sinnig
Software Engineering Group D
epartment of Computer Science,University of Rostock,
D-18051 Rostock, Germany
[pforbrig|ad|dreichart|dsinnig]@informatik.uni-rostock.de

ABSTRACT
This paper proposes a model-based approach for developing interactive applications. In particular a tool for deriving the navigational structure of the UI from task, object, user and device models is introduced. The editor is based on the XIML technology and allows simulations considering temporal relations between task and design decisions for the navigation dialogue.

KEYWORDS
Model-based Design, Navigation Dialogue, Abstract Prototype, XIML

INTRODUCTION
Rapid development of user interfaces, as performed in the course of prototyping, helps developers to understand the functionality and facilitates the participation of users.

Interactive system development that takes into account the work of end users has to comprise some representation of this work. In order to develop software based on user tasks and objects, several frameworks have been introduced. Our approach is characterized by figure 1. It demonstrates that the dialogue model and the application model have to be based on the same analysis specification, which consists of mutual related models of tasks, users, business objects and devices.

Software development is considered as a sequence of transformations mainly controlled by patterns. We already developed a tool for transformations controlled by design patterns [9]. It is our goal to develop similar tools for the user interface design. This paper is focused on a tool supporting the transformation from task models to dialogue models which is intended to be supported by patterns later on.

LEAVE BLANK THE LAST 2.5 cm (1") OF THE LEFT COLUMN ON THE FIRST PAGE FOR THE COPYRIGHT NOTICE.

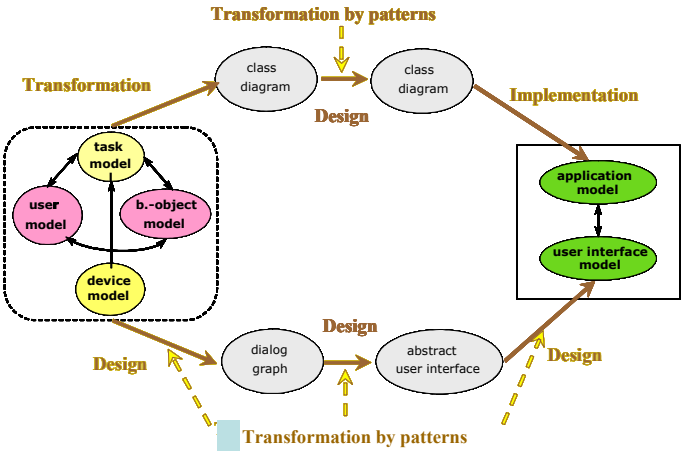

Figure 1: Model-based software development process

TaO PRINCIPLES AND XIML FRAMEWORK
The XIML (eXtensible Interface Markup Language) [16] is a framework for specifying models for interactive systems. It allows the description of tasks, objects, users and devices as well as the description of user interfaces.

From our point of view it was fascinating to represent our ideas of models by XIML specifications. This was possible without problems because XIML allows introducing relations between different model elements. One can introduce relations between classes of the task model and classes of the domain model. (Here domain model is once again used as business-object model in our terminology.) For instance the "task_has_artefact" relation is a binary relation between a task and an object. In the same way tools can be attached to tasks.

It was also possible to allow more general temporal relations for tasks [21] than usually used. Relations between tasks at different levels of hierarchy are possible.

At first a tool was developed to read an XIML file, to present different views for tasks, users and objects and to allow an animation according to scenarios.

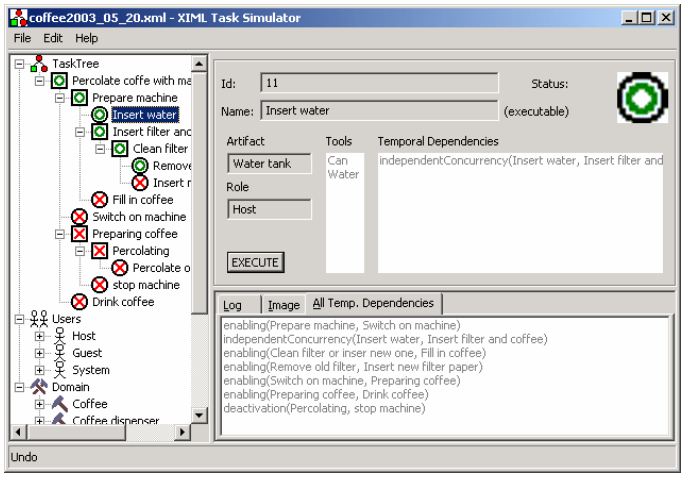

Figure 2: Start situation of animating a XIML model for percolating coffee

Basic tasks (in rectangular boxes) can be executed from the view point of tasks, users and domain objects. The attached artefacts, the tools, the role, and the involved temporal relations can be seen on the right hand side.

The screen shot of figure 3 portrays a situation after filling water into the tank. There is only one basic task that can be performed. Due to the temporal relations this is "Remove old filter". This can be checked by having a look at the temporal relations displayed on the screen under "All Temp. Dependencies".

This first experiment demonstrated the opportunities of XIML and pushed the idea of using task models for requirements analysis and design. In the next section we will demonstrate how the design of navigation dialogues can be supported.

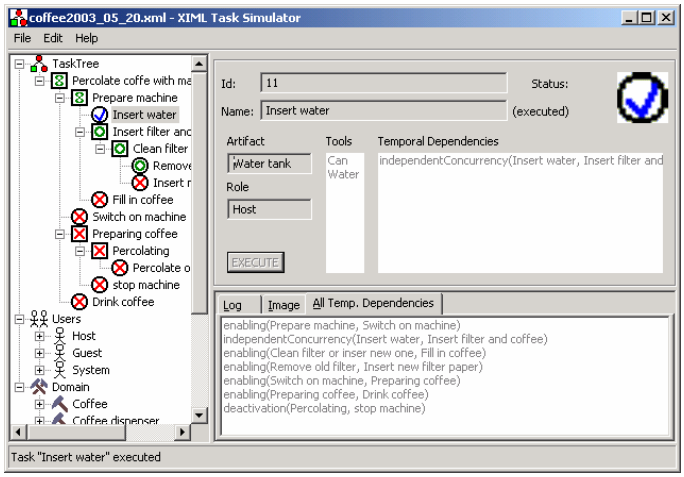

Figure 3: Situation during animation

Figure 4 presents the screen shot of an animation of a task model for a simple electronic shop. It displays the hierarchical structure and the temporal relations of tasks. This task model is the basis for designing a navigation dialogue later on in figure 6.

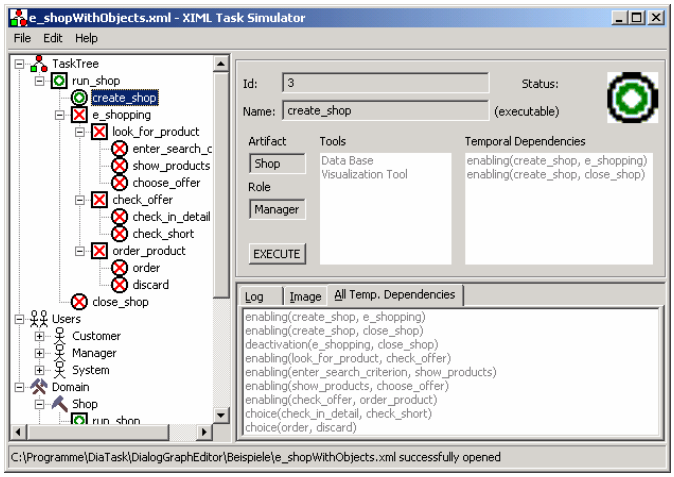

Figure 4: Start situation of animating a XIML model of an electronic shop.

DESIGNING DIALOGUES

There are different strategies to design the dialogue model. Janus [2] uses information mainly from the object model, but most approaches are based on tasks. Teresa [3] follows an idea of grouping tasks based on preconditions, which allows an automatic generation of dialogue models. Using our method of explicitly designing a dialogue graph

an alternative strategy by designing a very abstract user interface can be employed. The software developer has to decide which tasks are grouped together in one view and how the transition from one view to another one is specified. Views can simply be considered as a group of elements.

A dialogue graph consists of dialogue views and transitions. There are five types of dialogue views and two types of transition views, which are presented by Figure 8.

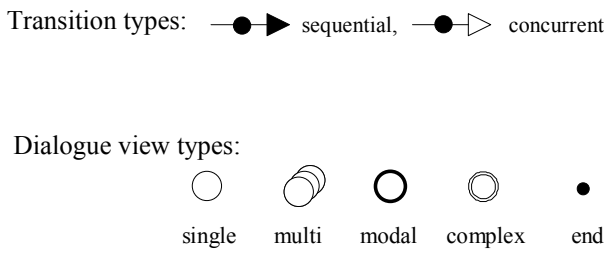

Figure 5: Elements of a dialogue graph

In contrast to sequential transition a concurrent transition means that both views are still visible. Based on these assumptions the idea emerged to develop an editor that allows manipulating such graphs by attaching tasks to transactions and to views. (If a task is attached to a view no transition takes place by executing this task).

DIALOGUE GRAPH EDITOR

This editor is able to read and write a file specifying task, user and object models as XIML specification. A representation of dialogue graphs was developed as well.

With this editor it is possible to develop different dialogue graphs for the same task model. Practically all models can be stored together into one single XIML file.

On the left hand side of figure 6, the task model is visible. It is the task model of figure 3. For this model a dialogue graph was designed, which is presented on the right hand side. One node is characterized as starting point. This characterization is represented by a traffic light.

By selecting a task (e.g. create shop) and a transition (e.g. sequential) a transition can be specified by drawing a line between to nodes. In this way a task is attached to a transition.

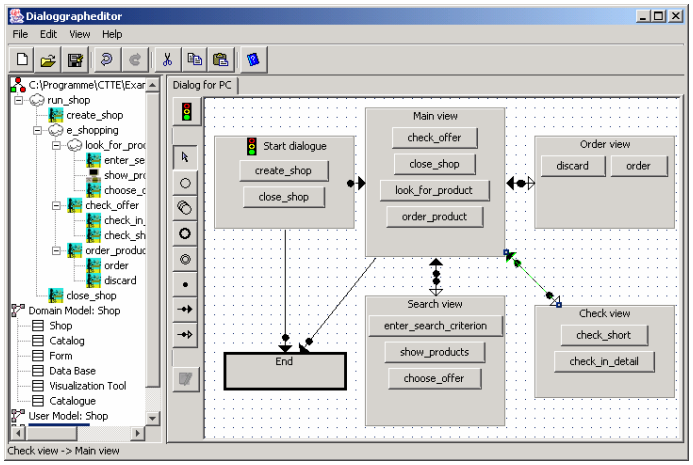

Figure 6: `Dialogue graph editor`

Clicking on the traffic light can animate the dialogue graph. A new window appears containing all visible views. Figure 7 demonstrates a special situation during the animation. Two views ("Main view" and "Search view") are visible. At this moment the view "Search view" is active but only the task "enter_serach_criterion" can be executed. Thus, the animation does not only consider specification from the dialogue model itself but it interprets temporal relations from task model as well. By clicking on the "Main view" button this view will become active and all active task are presented by active buttons.

Figure 7: Animated dialogue graph editor

In the following we will look at an example of a dialog graph for a mailing system containing a multiple view.

In figure 8 it is specified that there is a concurrent transition and a sequential transitions between the views "Window Mail List" and "Window Mail". Assuming view "Window Mail List" is active. By selecting a mail (Select and open mail) a concurrent transition is executed. A view "Window Mail" appears and view "Window Mail List" stays visible. A sequential transition (e.g. initiated by "Close mail") results in a disappearing of the view which is the origin of the transition (e.g. "Window Mail").

"Window Mail" is a multiple view. That means that the concurrent transitions to this view have to be object-based. For each object (in our example for each mail) an own view is dynamically created.

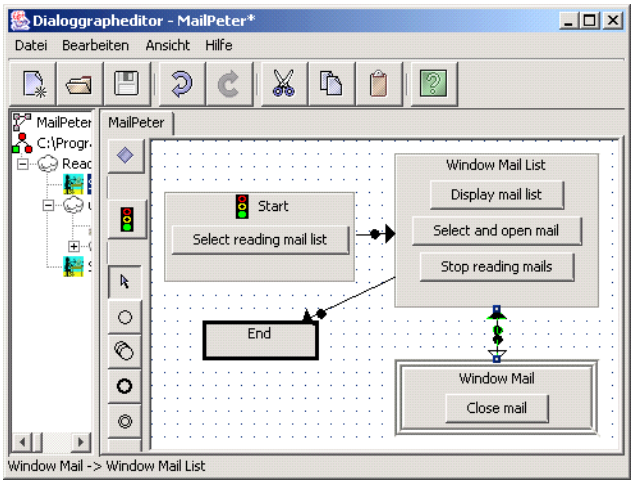

Figure 8: Dialog graph for a mailing system

The specification of figure 8 can be animated as well. Figure 9 displays a screenshot taken after performing twice "Select and open mail".

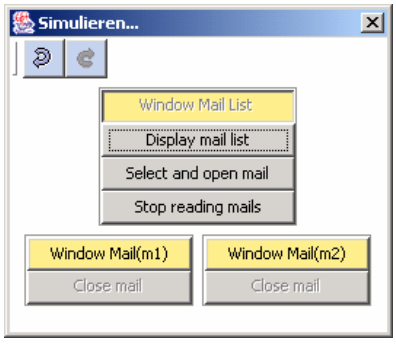

Figure 9: Example of multiple views

In figure 8 there is no task model visible for the dialog graph. Indeed there is no task model attached to the dialog graph.

At the moment task models are not able to specify the all necessary behaviour. If "Select and open mail" is specified as iterative task, reading of one mail has to be finished before the next mail can be read. Recursion might help to solve the problem but the definition of a kind of "instance iteration" operator seems to be more usable.

Reader familiar with CTTE [12] might recognize the similarity of the icons in the example with those provided in CTTE. The dialogue graph editor has an import interface for CTTE models stored in XML format. Such models can be imported and dialogue graphs can be developed as an alternative to the user interface development of Teresa. The task model for the electronic shop was developed with CTTE and imported to our TaO system. The information related to our methodology of artefacts, tools and roles was attached later.

As already mentioned with the dialogue graph editor several dialogue graphs can be designed for one analysis model. All models can be animated and forthcoming users can participate in the design of the user interface.

SUMMARY AND WORK TO BE DONE

It was outlined how the design process of interactive systems can be structured and how it can be supported by tools in its early phases. The metaphor of tasks, artefacts and tools was used to describe models in the analysis phase.

Such models can be the basis for developing dialogues as well. One of such approaches is supported by Teresa within the Cameleon project [3] by computing task sets.

In this paper, an alternative approach was presented, which can be characterized as an interactive design process by hierarchical dialog graphs. Tool support allows a joined animation of the dialog graph specification (abstract prototypes) with the corresponding task specification. The abstract prototype is already animated according to the temporal relations of the task model. Additionally the animated task tree is visualized.

At the moment, specifications of object based transitions are not supported in an optimal way. More precise object information is necessary for this purpose. Patterns will be used within this context as well. First concepts have already been developed. They have to be refined in the future.

REFERENCES

1. UIML Tutorial. http://www.harmonia.com
2. H. Balzert, "From OOA to GUIs: The JANUS System", Journal of Object-Oriented Programming, Febr. 1996, pp. 43-47.
3. Cameleon project, http://giove.cnuce.cnr.it/cameleon/SIGatCHI.html.
4. J. Eisenstein, J. Vanderdonckt and A. Puerta, "Adapting to Mobile Contexts with User-Interface Modeling", Workshop on Mobile Computing Systems and Applications 2000 (Monterey, CA, 7-8 December 2000), IEEE Press
5. J.D. Foley, History, Results and Bibliography of the User Interface Design Environment (UIDE), an Early Model-based Systems for User Interface Design and Implementation, in Proc. of DSV-IS'94, Carrara, 8-10 June 1994, pp. 3-14.
6. P. Forbrig, R. Lämmel and D. Mannhaupt, "Patterns-oriented Development with Rational Rose", Rational Edge, Vol. 1, No. 1, 2001,
7. P. Forbrig and A. Dittmar, "Interfacing Business Object Models and User Models with Action Models", Proc. HCI International 2003, Vol. IV, p. 83-87, Greece
8. P. Forbrig an A. Dittmar, "Bridging the Gap between Scenarios and Formal Models", tProc. HCI International 2003, Vol. I, p. 98-102, Greece
9. E. Gamma, R. Helm, R. Johnson and J. Vlissides, "Design Patterns. Addison-Wesley", 1995
10. P. Johnson, "Human Computer Interaction: Psychology, Task Analysis and Software Engineering", McGRAW HILL BOOK COMPANY, 1992.
11. G. Mori, F. Paternò, C. Santoro, Tool Support for Designing Nomadic Applications, Proceedings of IUI 2003 -Miami, Florida, January 12-15, 2003.
12. F. Paterno, C. Mancini, S. Meniconi, ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models, Proceedings Interact'97, Chapman&Hall , 1997, pp. 362-369.
13. F. Paternò, Model-Based Design and Evaluation of Interactive Applications. Springer, 2000
14. A. Puerta and J. Eisenstein, "XIML: A Common Representation for Interaction Data", Sixth International Conference on Intelligent User Interfaces, IUI 2002.
15. UML: http://www.uml.org
16. XIML, http://www.ximl.org
17. A. Dittmar, "More precise descriptions of temporal relations within task models", DSV-IS 2000, Limerick, June 2000.
18. P. Forbrig, A. Dittmar, D. Reichart, D. Sinnig, User-Centred Design and Abstract Prototypes, Proc. BIR 2003, p. 132 – 145, Berlin, September 2003.
19. D. Sinnig, H. Javahery, P. Forbrig, A. Seffah, The Complicity of Model-Based Approaches and Patterns for UI Engineering, Proc. BIR 2003, p. 120-131, Berlin, September 2003.
20. XUL, http://www.xulplanet.com/ , 2003.
21. A. Dittmar, "More precise descriptions of temporal relations within task models", DSV-IS 2000, Limerick, June 2000.
22. M. Biere, B. Bomsdorf and G. Szwillus, "The Visual Task Model Builder", Proceedings of the CADUI'99, Kluwer Academic Publishers, 1999, pp. 245-256.
23. A. Dix, J. Finlay, G. Abowd and R. Beale, "Human Computer Interaction", Prentice Hall, 1