Chapter 18

INTERACTION TEMPLATES FOR CONSTRUCTING USER INTERFACES FROM TASK MODELS

David Paquette and Kevin Schneider
Department of Computer Science, University of Saskatchewan,
Saskatoon, SK S7N 5A9 (Canada)
E-mail: dnp972@mail.usask.ca, kas@cs.usask.ca

Abstract Task modelling is well suited to identifying user goals and identifying the activities a user performs to achieve these goals. Some task model tools provide simulation capabilities and/or aid in the construction of concrete user interfaces. When it is desirable for the simulated or constructed interface to be realistic, the task model must be specified in considerable detail. Unfortunately this is usually quite onerous for medium to large size systems, for context-dependent user interfaces, and for highly interactive user interfaces. This paper introduces 'Interaction Templates': pre-defined components that can be plugged into a task model to provide concrete dialogue and presentation. Interaction Templates define complex, context sensitive interaction that is to be incorporated into the target user interface and can be used when simulating a task model. Interaction Templates are bound to the task model using an explicit data model. We demonstrate the applicability of Interaction Templates with a case study..

Keywords: Methods and languages, Model-based interface design, Task modelling, Templates, User interface design and specification.

1. INTRODUCTION

Our research applies task modelling approaches to the design of interactive systems. Through a case study we investigated benefits and deficiencies of using task modelling for the design of an interactive system for the propose of tracking soil samples. In particular we are interested in applying ConcurTaskTrees [4] and tool support for simulating a user interface. Other task modelling approaches include UAN [2], SUIDT [1] and U-Tel [8].

Our case study shows that task modelling, although useful for modelling an interactive system with a user, becomes tedious when specifying a task model in sufficient detail to benefit from simulation tool support.

We propose the use of Interaction Templates, pre-defined components that can be plugged into a task model to provide concrete dialogue and presentation. Interaction Templates define complex, context sensitive interaction that is to be incorporated into the target user interface and can be used when simulating a task model. Interaction Templates are bound to the task model using an explicit data model.

CTTE [3,4,5] is a tool for building and evaluating ConcurTaskTrees and was used in our case study. It features a visual task model builder and a task model simulator, which is used to validate task models.

The visual task model builder aids in creating models. It is used to create and define tasks as well as define relationships between those tasks. It includes features that help with layout as well as features that help to ensure the mathematical correctness of a model.

The simulator can be useful in validating task models with users, as well as evaluating usability at a very early stage. With CTTE's Task Model Simulator tasks can be selected from a list of available actions. Any one of the actions can be performed by double clicking it. When an action is performed, the list of enabled actions is updated. This allows users to test the system design to ensure that it allows them to reach certain goals and does not allow actions to be performed when they should not be.

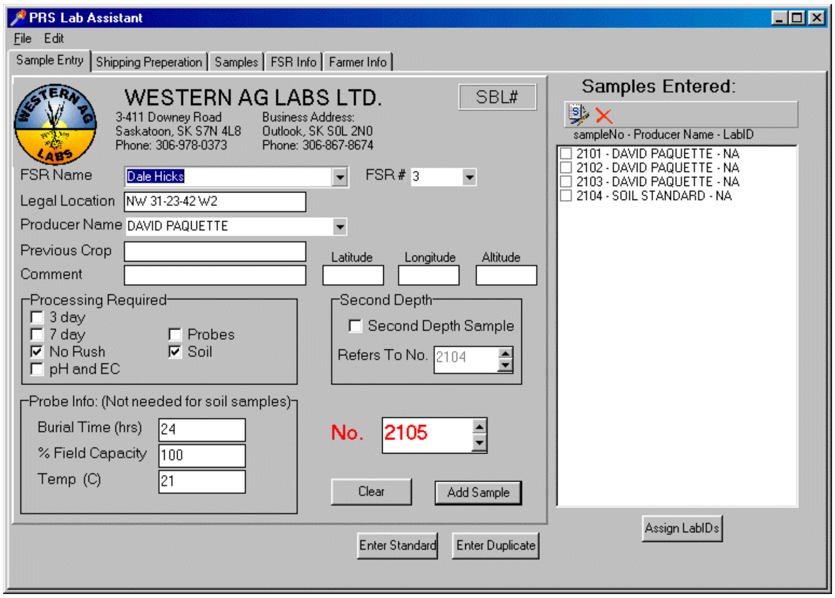
A drawback of this simulator is that it does not allow for true simulation of concurrent tasks. It treats concurrent tasks the same as choice tasks.

A tool similar to CTTE provides a simulator that has the ability to simulate concurrent tasks [6]. The simulation is made up of a window that is filled with smaller windows containing play and stop buttons. Each of the smaller windows represents a task from the task model. These tasks can either be active or inactive. A user can activate a task by pressing the play button, or deactivate a task by pressing the stop button. Activating or deactivating a task in the simulator has the same effect as starting or stopping a task in the system. Activities can either be enabled or disabled. This is shown in the simulator by enabling or disabling the play and stop buttons. When a task is activated, all tasks that are enabled or disabled by that task are updated. By playing through different scenarios, it is easy to validate your task model with users. One of the problems of this simulator is that it would get difficult to sort through the many tasks involved in a large system.

In the next section we describe a case study in which we applied task modelling to an existing interactive system.

2. LAB ASSISTANT CASE STUDY

Lab Assistant is an information system for tracking soil samples through a soil–processing lab in Saskatoon, Saskatchewan. The samples are provided to the lab by producers and university researchers. The system is data intensive, interacting with a database containing information about farmers, geographic locations, soil samples and quality control data. The Lab Assistant is used to enter the data, verify the data, and create a variety of reports for the users. The system was built using Borland Delphi 4 and runs on Microsoft Windows operating systems. Fig. 1 is a screen shot of the Lab Assistant soil sample data entry form.



WESTERN AG LABS LTD.
3-411 Downey Road
Saskatoon, SK S7N 4L8
Phone: 306-978-0373

Business Address:
Outlook, SK S0L 2N0
Phone: 306-867-8674

FSR Name	Dale Hicks
FSR #	3
Legal Location	NW 31-23-42 W2
Producer Name	DAVID PAQUETTE
Previous Crop	
Comment	
Latitude	
Longitude	
Altitude	

Processing Required
☐ 3 day
☐ 7 day
☑ No Rush
☐ pH and EC
☐ Probes
☑ Soil

Second Depth
☐ Second Depth Sample
Refers To No. 2104

Probe Info: (Not needed for soil samples)

Burial Time (hrs)	24
% Field Capacity	100
Temp (C)	21

No. 2105

Samples Entered:
sampleNo - Producer Name - LabID
☐ 2101 - DAVID PAQUETTE - NA
☐ 2102 - DAVID PAQUETTE - NA
☐ 2103 - DAVID PAQUETTE - NA
☐ 2104 - SOIL STANDARD - NA

Figure 1. Lab Assistant Soil Sample Data Entry Form.

2.1 Detailed Task Model

The ConcurTaskTree environment (CTTE) was used to model the user interactions with the Lab Assistant. Each window was considered to be an abstract task named after the main task performed in that window. The abstract tasks were modelled in detail to describe the user's interaction. Widgets such as text fields and buttons were modelled as Interaction Tasks. Relationships between tasks showed the effects each widget had on other objects in the system. Fig. 2 shows a small section of the task model that was developed. The model gave a precise definition of the system and allowed for an accurate simulation using CTTE.

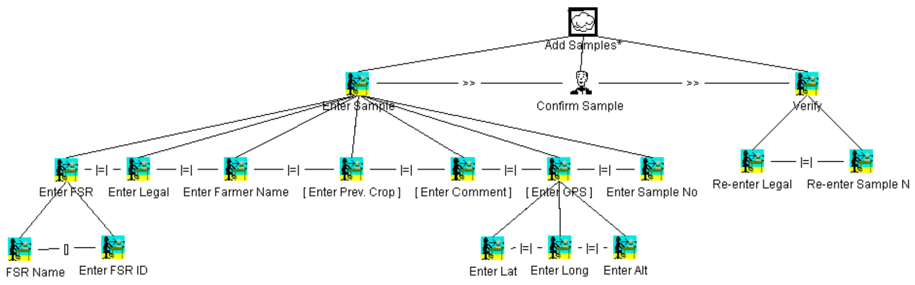


Figure 2. Small section of detailed ConcurTaskTree of Lab Assistant.

Although the task model provided an accurate and precise description of the Lab Assistant, some difficulties were encountered. The process of generating a detailed task model for an information system the size of the Lab Assistant was a long and tedious process. Every interaction between the user and the system was modelled to enable a reasonable simulation of the user interface. The model quickly became too large to be easily understood. The section shown in Fig. 2 only accounted for approximately 10% of the entire model. A model of this size is difficult to view in its entirety and it was difficult to get a good overview of the system from the task model.

2.2 Abstract Task Model

In an attempt to overcome the difficulties encountered in building the detailed task model, a second, less detailed task model was created. Here, the system was modelled at the abstract task level. Each window was again as an abstraction task named after the main task performed in that window. However, the specifics of how those tasks were performed inside the window were not considered. Task such as entering specific fields when adding samples were omitted. The abstract task model corresponding to Fig. 2 is shown in Fig. 3. This model was both quicker and easier to build. Because it was much smaller, it also gave a better overview of the system than the previous model.

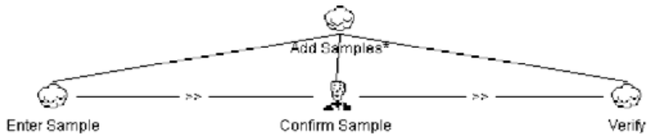


Figure 3. Small section of abstract ConcurTaskTree of Lab Assistant.

Modelling a system at this level of abstraction results in a loss of some of the functionality that CTTE offers. In particular, because some of the detail is missing, CTTE can no longer provide a suitable simulation of the task model. A simulation of entering a sample no longer consists of entering a few required fields and some optional ones. The simulation only consists of entering an abstract sample object, the details of which are unknown. Another problem is that valuable information about the system was lost. It is no longer known what data is verified. In order to fully understand what is occurring in this model, one must already have knowledge of the system.

2.3 User Feedback

In order to get user feedback on the validity of the two task models, the main user involved in the development of the Lab Assistant was introduced to the concept of task models using CTTE. The user is a business computer user with limited technical background, however, she interacts with the Lab Assistant system every day.

First, the user was given an overview of the CTT notation, and taken through an example of modelling a small portion of the Lab Assistant. Together, the task of adding a batch of samples to the database was modelled. The user did not have any difficulties in following the construction of the model, and was able to point out some inaccuracies as the model was built. The user was able to correct a task operator in the model. User feedback allowed for a more accurate model of the Lab Assistant to be built. After analysing the sample task model, the user expressed some concern as to how large the model would be if the entire system were modelled. The small task model built in this example had already covered the entire screen.

Once the small model was built, the user was introduced to the task model simulator. The user was excited by the possibilities of the simulator: the idea of simulating a software system before it was built seemed very useful to her. She commented that if this tool had been available when designing the Lab Assistant, some design flaws would likely have been identified much earlier in the design process. In the current version of the Lab Assistant, the flow of data does not match the flow of samples through the lab. To work around this problem, users must change the way soil is analysed to match the assumptions that were made when the Lab Assistant was built. There are no current plans to fix this problem, as the cost is expected be significant. If this problem could have been identified in a simulation with users at an early stage in design, the cost would have been minimal.

Next the user was shown the detailed task model and the simplified task model that had been previously developed. The user was overwhelmed by the size of the first model. Since the model was too large to view in its entirety, she found it difficult to see the ‘big picture’ of the Lab Assistant from the model. She was impressed by the functionality available in the simulator as it allowed for a very accurate simulation of the Lab Assistant. The user found the second task model easier to understand. She found that because it lacked detail and was smaller, it gave a better overview of the Lab Assistant.

She did not, however, find the simulation as valuable in this model as it had been in the previous model. Overall the user felt that CTTE would be a useful tool in the analysis and design of Lab Assistant. In particular, she felt that the task model simulator integrated in CTTE would be a very valuable tool in validating proposed designs and catching major problems early in the design process.

2.4 Case Study Summary

CTTE provides a good mechanism for modelling information systems. Task trees provide an excellent overview of a system and offer a powerful simulation capability. Task trees are valuable as a common language for both designers and users. As was seen in the task modelling session with the user, it appears that it is beneficial to actively involve users in the creation of task models using CTTE, and that users are able to understand task models. Once the task models have been built, the simulation capabilities provided in tools such as CTTE can be extremely useful in validating proposed designs with users. A few problems arose when modelling information systems using CTTE. These problems can be summarised as being a problem of scale. Building accurate task models for medium to large systems is a long and tedious process. The models quickly grow to be too large to understand. This can be partially solved by modelling information systems at a more abstract task level, unfortunately with the side effect that simulations become less valuable and validating designs more difficult. The next section tries to address these shortfalls, with a proposal to extend task models with Interaction Templates.

3. INTERACTION TEMPLATES

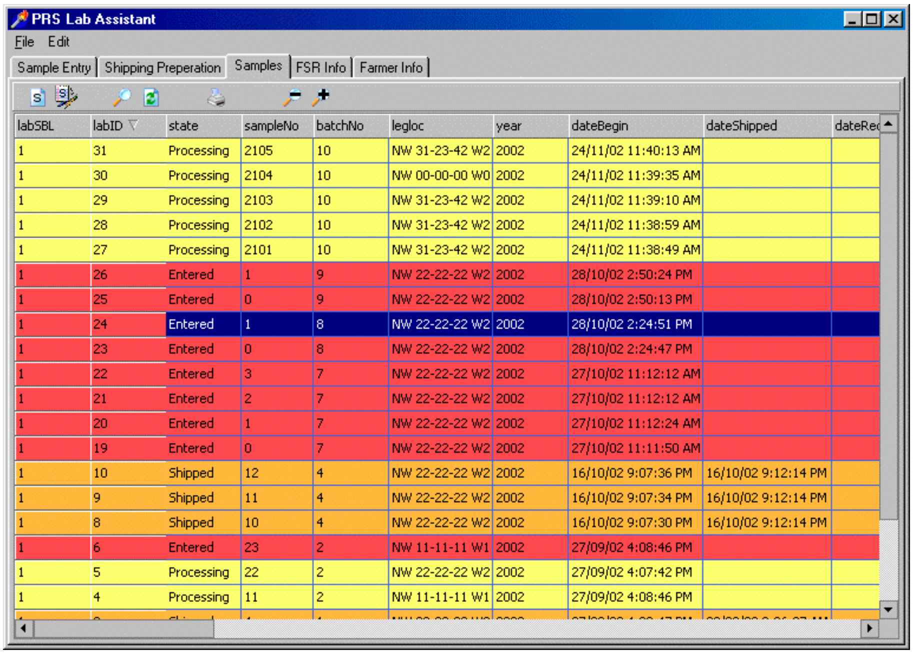
Interaction Templates were developed to attempt to find a middle ground between the two models (detailed and abstract) discussed in the Lab Assistant case study. Interaction Templates are intended to help us to build models quickly, and allow for detailed simulation while maintaining a useful system overview.

While building task models for information systems, there are sub trees that repeat throughout the model with only slight variations. These sub trees are associated with common interface interactions found in information systems. Interaction Templates model these common interface interactions. They include a detailed task model, an execution path (i.e., dialog), and a presentation component. Inserting and customising Interaction Templates reduces the need to model the same interaction repeatedly in a system, and thus, greatly reduces the time spent modelling information systems [7]. As well, Interaction Templates can be designed and tested to ensure their usability in accomplishing a task and can be designed to be 'plastic' [9] and thus adapt to different contexts [7].

Two common interface interactions, data table interaction and print dialog interaction, were chosen as examples of Interaction Templates. These examples illustrate how Interaction Templates attempt to solve some of the problems encountered in the Lab Assistant case study. The next two sections describe these Interaction Templates examples. To help illustrate an Interaction Template we provide a snapshot of its presentation for a specific context and its task tree.

3.1 Example 1: Data Table Interaction Template

Information systems typically deal with large amounts of data. This data is often visualised and interacted with by means of a data table component. The data table Interaction Template models the particular data table interaction found in the data tables of the Lab Assistant. With slight modifications, this Interaction Template could be adapted to model different data table components. The screen snap shot of the data table Interaction Template is shown in Fig. 4.



labSBL	labID	state	sampleNo	batchNo	legloc	year	dateBegin	dateShipped	dateRe
1	31	Processing	2105	10	NW 31-23-42 W2	2002	24/11/02 11:40:13 AM		
1	30	Processing	2104	10	NW 00-00-00 W0	2002	24/11/02 11:39:35 AM		
1	29	Processing	2103	10	NW 31-23-42 W2	2002	24/11/02 11:39:10 AM		
1	28	Processing	2102	10	NW 31-23-42 W2	2002	24/11/02 11:38:59 AM		
1	27	Processing	2101	10	NW 31-23-42 W2	2002	24/11/02 11:38:49 AM		
1	26	Entered	1	9	NW 22-22-22 W2	2002	28/10/02 2:50:24 PM		
1	25	Entered	0	9	NW 22-22-22 W2	2002	28/10/02 2:50:13 PM		
1	24	Entered	1	8	NW 22-22-22 W2	2002	28/10/02 2:24:51 PM		
1	23	Entered	0	8	NW 22-22-22 W2	2002	28/10/02 2:24:47 PM		
1	22	Entered	3	7	NW 22-22-22 W2	2002	27/10/02 11:12:12 AM		
1	21	Entered	2	7	NW 22-22-22 W2	2002	27/10/02 11:12:12 AM		
1	20	Entered	1	7	NW 22-22-22 W2	2002	27/10/02 11:12:24 AM		
1	19	Entered	0	7	NW 22-22-22 W2	2002	27/10/02 11:11:50 AM		
1	10	Shipped	12	4	NW 22-22-22 W2	2002	16/10/02 9:07:36 PM	16/10/02 9:12:14 PM	
1	9	Shipped	11	4	NW 22-22-22 W2	2002	16/10/02 9:07:34 PM	16/10/02 9:12:14 PM	
1	8	Shipped	10	4	NW 22-22-22 W2	2002	16/10/02 9:07:30 PM	16/10/02 9:12:14 PM	
1	6	Entered	23	2	NW 11-11-11 W1	2002	27/09/02 4:08:46 PM		
1	5	Processing	22	2	NW 22-22-22 W2	2002	27/09/02 4:07:42 PM		
1	4	Processing	11	2	NW 11-11-11 W1	2002	27/09/02 4:08:46 PM		

Figure 4. Data Table Interaction Template plugged into the Lab Assistant software.

In the abstract view (Fig. 5), with all sub trees hidden, the data table Interaction Template contains a description of the data the table will be showing. The data is actually expressed with XML but for simplicity we have shown just the key data fields in our figures. Given this abstract view and field information of the data table Interaction Template, the details of the interaction are known without viewing the larger detailed view shown in Fig. 6. Because data tables are common interface interactions in information systems, most users and designers do not need to view the detailed model to understand the interactions that are being described.

This is the feature of the Interaction Template that allows for detailed simulation while maintaining reasonable model size. Hiding the details below the abstraction table interaction task helps to condense the overall size of the model.

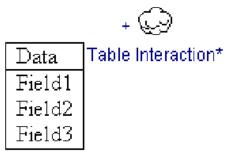


Figure 5. Abstract Task Tree of the Interaction Template for a data table.

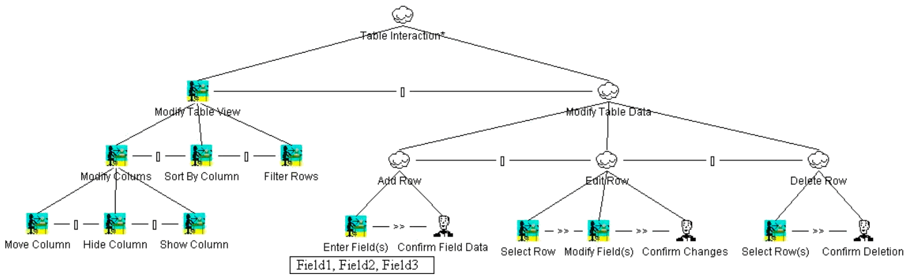


Figure 6. Detailed Task Tree of the Interaction Template for a data table.

While it may not be useful to see the details of a table interaction when trying to view an entire model, these details do become necessary when using the task model simulator. Given the description of the data, it would be possible for a tool to customise the details of the template. With such a tool, the Enter Field(s) task in Fig. 6 would be broken down into Enter Field1, Enter Field2, and Enter Field3.

The Move Column, Hide Column, Show Column, and Sort By Column tasks could be expanded in a similar fashion. With this Interaction Template, adding a detailed model of a table interaction would be as simple as selecting the Interaction Template and defining the data that will be displayed. With proper tool support, a lot of the tedious work would be completed automatically.

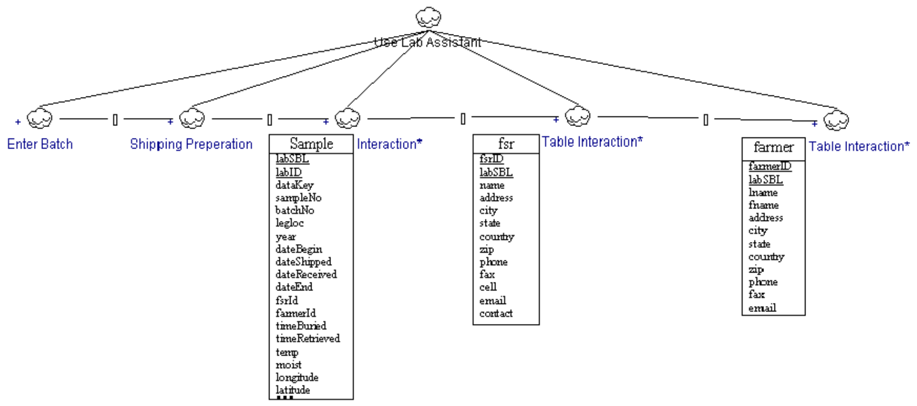


Figure 7. Abstract view of Lab Assistant Task Tree.

The detailed view shows exactly how a user interacts with a data table. The interaction is divided into two subtasks: modify table view and modify table data. When modifying the table view, a user can move, hide and show columns, as well as sort and filter rows. When modifying table data, a user can add, edit and delete rows. With this parameterised Interaction Template, detailed ConcurTaskTrees that are customised for a specific use could be built quickly.

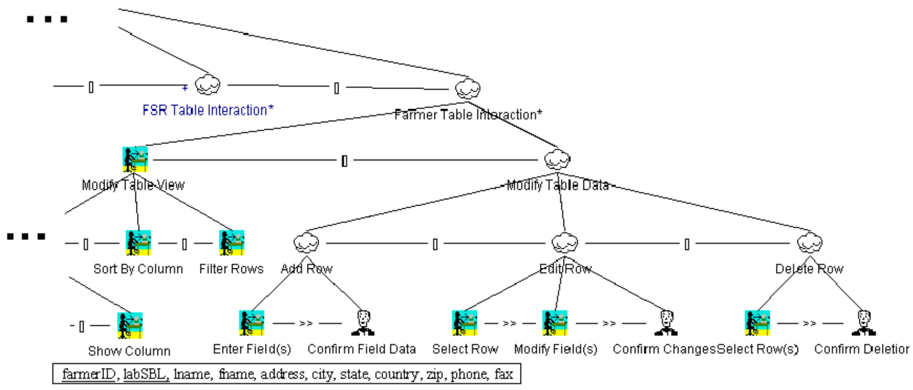


Figure 8. Lab Assistant Task Tree with the Data Table Interaction Template plugged in.

Fig. 7 shows how the data table Interaction Template was used in the Lab Assistant. The abstract view gave a good overview of the system, and helped to keep the model at a manageable size. In Fig. 8, the *fsr* (field service representative) and *farmer* tables fit perfectly with the template. No manual changes to the template were necessary to model these portions of the Lab Assistant. The *sample* table, however, did not match the template exactly. In the *sample* table, it is not possible to add a sample directly to the table. In order to add a sample, the user must go to the abstract task of entering a batch. The *sample* table also does not allow rows to be deleted. Sample deletion is handled in a separate software system called the Lab Manager. The Lab Manager was not modelled in this case study.

Fig. 9 shows how the template was inserted, and easily modified to reflect the differences found in the *sample* table. Under the abstract task *Modify Table Data*, *Add Row* was replaced with the abstract task *Enter Batch*. The abstract task *Delete Row* was simply removed. Although the original template did not match the sample table perfectly, the effort required to manually customise it was minimal.

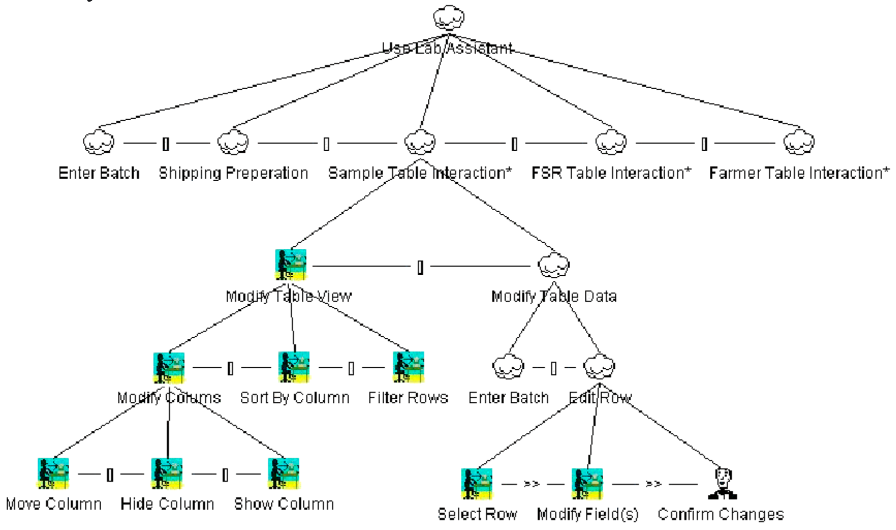


Figure 9. Data Table Interaction modified to fit the sample table.

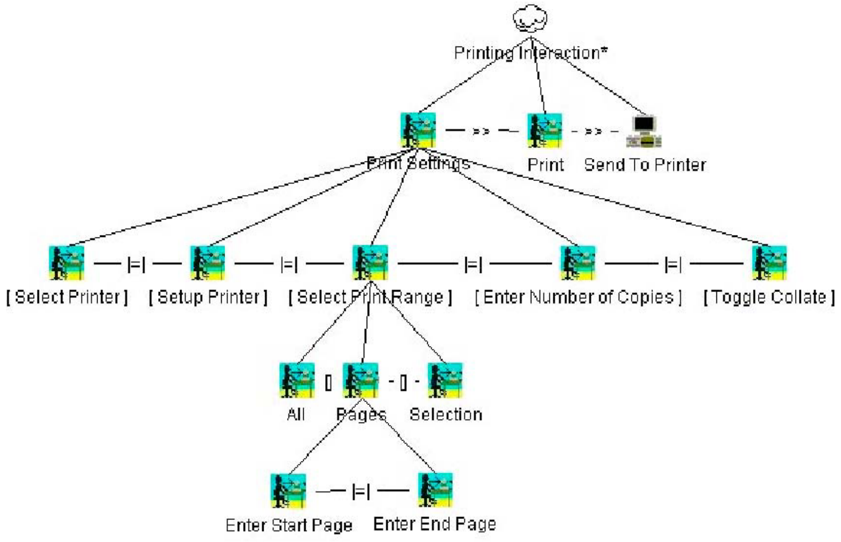
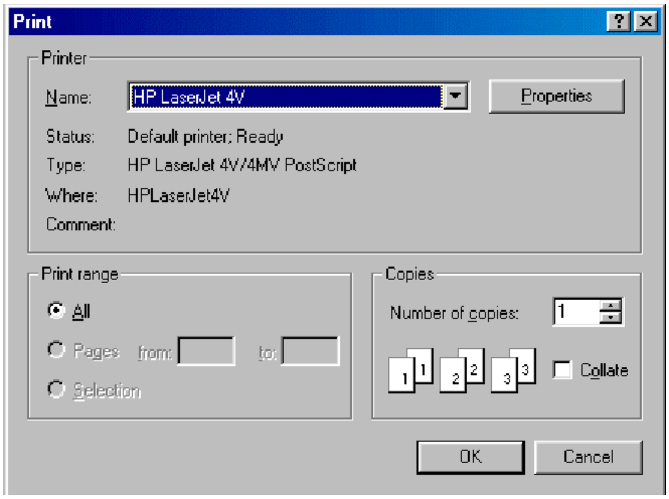


Figure 10. Print Dialog Interaction Template Task Model.



Name: HP LaserJet 4V
Status: Default printer; Ready
Type: HP LaserJet 4V/4MV PostScript
Where: HPLaserJet4V
Comment:

Print range: (● All) ○ Pages from: to: ○ Selection

Number of copies: 1
☐ Collate

Figure 11. Print Dialog Interaction Template for Windows 9x.

3.2 Example 2: Print Dialog Interaction Template

Another interesting interaction that is commonly found in information systems is the print dialog. Since print dialogs are usually built in to the operating system, they are almost always identical in every place they appear in a software system. This makes them an excellent candidate for Interaction Templates. If they are always the same, an entire print dialog interaction can be added in only one step. The only parameter needed for the Print Dialog Interaction Template is an operating system, because print dialogs differ slightly between operating systems. If an Interaction Template was being built and a printer interaction was needed, the Print Dialog Interaction Template could be selected, and an operating system selected. A model of a print dialog matching the selected OS would then be inserted into the task model. A plastic Interaction Template could be used to provide different behaviour for each operating system. An example of the Print Dialog Interaction Template for Windows 9x is shown in Fig. 10. The Windows 9x print dialog box is shown in Fig. 11.

4. CONCLUSION

Interaction Templates were developed to solve the problems identified in the Lab Assistant case study. They attempt to solve these problems by taking advantage of common interface interactions found in information systems. The data table interaction and print dialog interaction examples show how

Interaction Templates attempt to solve these problems of scale by condensing the overall model size while maintaining quality of information. In the future we hope to build tool support for building ConcurTaskTrees using Interaction Templates.

REFERENCES

[1] Baron, M. and Girard, P., *SUIDT: A Task Model Based GUI-Builder*, in C. Pribeanu, J. Vanderdonckt (eds.), "Task Models and Diagrams for User Interface Design", Proceedings of the 1st International Workshop on Task Models and Diagrams for User Interface Design TAMODIA'2002 (Bucharest, 18-19 July 2002), INFOREC Printing House, Bucharest, 2002, pp. 64-71.

[2] Brandenburg, J.L., Hartson, H.R., and Hix, D., *Different Languages for Different Development Activities: Behavioral Representations Techniques for User Interface Design*, in B. Myers (ed.), "Languages for Developing User Interfaces", Jones and Bartlett Publishers, Boston, 1992.

[3] Paternò, F., *Model-Based Design and Evaluation of Interactive Applications*, Springer-Verlag, Berlin, 2000.

[4] Paternò, F., *Task Models in Interactive Software Systems*, in S.K. Chang (ed.), "Handbook of Software Engineering and Knowledge Engineering", Vol. 1, World Scientific Publishing Co., River Edge, 2001, accessible at ftp://cs.pitt.edu/chang/handbook/21.pdf

[5] Paternò. F., *Tools for task modelling: Where we are, where we are headed*, in C. Pribeanu, J. Vanderdonckt (eds.), "Task Models and Diagrams for User Interface Design", Proceedings of the 1st International Workshop on Task Models and Diagrams for User Interface Design TAMODIA'2002 (Bucharest, 18-19 July 2002), INFOREC Printing House, Bucharest, 2002, pp. 10–17.

[6] Seffah, A. and Forbrig, P., *Multiple User Interfaces: Towards a Task-Driven and Patterns-Oriented Design Model*, in P. Forbrig, Q. Limbourg, B. Urban, J. Vanderdonckt (eds.), Proceedings of the 9th International Conference on Design, Specification, and Verification of Interactive Systems DSV-IS'2002 (Rostock, 12-14 June 2002), Lecture Notes in Computer Science, Vol. 2545, Springer-Verlag, Berlin, 2002, pp. 118–132.

[7] Souchon, N., Limbourg, Q., and Vanderdonckt, J., *Task Modelling in Multiple Contexts of Use*, in P. Forbrig, Q. Limbourg, B. Urban, J. Vanderdonckt (eds.), Proceedings of 9th International Workshop on Design, Specification and Verification of Interactive Systems DSV-IS 2002 (Rostock, 12-14 June 2002), Lecture Notes in Computer Science, Vol. 2545, Springer-Verlag, Berlin, 2002, pp. 59-73.

[8] Tam, R.C.M., Maulsby, D., and Puerta, A.R., *U-TEL: A Tool for Eliciting User Task Models from Domain Experts*, in Proceedings of 3rd ACM International Conference on Intelligent User Interfaces IUI'98 (San Francisco, 6-9 January 1998), ACM Press, New York, 1998, pp. 77-80.

[9] Thevenin, D. and Coutaz, J., *Plasticity of User Interfaces: Framework and Research Agenda*, in A. Sasse, Ch. Johnson (eds.), Proceedings of 7th IFIP TC.13 Internationa Conference on Human-Computer Interaction Interact'99 (Edinburgh, 30 August- 3 September 1999), IOS Press Publ., Amsterdam, 1999, pp. 110–117.

[10] van Welie, M., van der Veer, G.C., and Eliens, A., *Patterns as Tools for User Interface Design*, in J. Vanderdonckt, Ch. Farenc (eds.), Proceedings of International Workshop on Tools for Working with Guidelines TFWWG'2000 (Biarritz, 7-8 October 2000), Springer-Verlag, London, 2000, pp. 313-324.