Toward Multi-disciplinary Model-Based (Re)Design of Sustainable User Interfaces

Jan Van den Bergh, Mieke Haesen, Kris Luyten,
Sofie Notelaers, and Karin Coninx

Hasselt University - tUL - IBBT, Expertise Centre for Digital Media,
Wetenschapspark 2, 3590 Diepenbeek, Belgium
{ jan.vandenbergh,mieke.haesen,kris.luyten,sofie.notelaers,karin.coninx}@uhasselt.be

Abstract. This paper reports on our experience in using the MuiCSer process framework for the redesign of the user interface for operating an industrial digital printing system. MuiCSer is created to support the user-centered interface design of new *and* legacy systems by a multi-disciplinary team. The process framework is created to enhance increased flexibility, usability and sustainability of the designed user interfaces. Resulting user interfaces are decoupled from the application logic, but still help to maintain consistency with the available functionality even when this changes over time. This report focuses on the usage of the task model during the analysis of the current user interface, the creation of user interface prototypes at various fidelity levels and the still ongoing realization of a flexible user interface management system to support future changes of the deployed user interfaces.

1 Introduction

In contrast with traditional design efforts, our challenge is to create a new design that can evolve together with the rest of the application: a sustainable user interface. The purpose is to reduce the cost of further improvements to the software that will be applied after the design has finished. It is very likely that a complex industrial application such as the one we are targeting, will be updated regularly (as evidenced by the history of the redesigned software). Because of the extreme complexity of such an application, the efforts of updating application logic as well as the related user interface have a high cost.

Furthermore, in terms of user interface complexity, we are dealing with a user interface that surpasses the complexity of most user interfaces a regular user has to deal with. The user interface of such a high-end digital printing system can easily contain hundreds of different windows (tabs are counted as a separate window) although this amount can vary depending on the needs of the specific user. It is easy to loose track of both the overview and the important details when only traditional design methodologies (e.g. card sorting and paper prototypes) are used to steer the redesign.

A model-based approach allows capturing these inter- and intra-window relationships and link window sequences with tasks that need to be supported using

e.g. a task model. It is however equally important to be able to trace the individual components in the final user interface back to the tasks they support. When dealing with the redesign of such complex user interfaces in combination with new as well as changing requirements, a fully automated transformational approach is not an option. To support these goals we used a new software engineering process framework, *Multi-disciplinary user-Centered Software engineering (MuiCSer)*, that is visible for the customer, has some degree of agility and allows more easy updates once a design is deployed as the final user interface [1]. This approach is based on our experience with model-based approaches, user-centered user interface design and software engineering processes to provide a solution that supports user-centered interface redesign for complex evolving systems.

MuiCSer does not stop at first delivery of the system and thus the necessary tools and runtime environment need to be provided to support further evolution of the system. Therefore we ensured that existing commercially supported tools can be used to further develop the system, while custom tools are provided to keep track of the model relations. In this paper, we discuss our first results of this approach to redesign the user interface of a complex industrial application into a sustainable user interface.

2 MuiCSer

The usability of a system can be improved by using a User-Centered Design approach as described in ISO 13407 [2]. When a redesign concerns the user interface of a complex system, it is also important not to loose track of the functionality offered by the application logic. Model-based approaches can help us to preserve the link between the user interface and the application logic. We use the MuiCSer process framework [1] to provide a smooth integration between user interface design and software development and to support the involvement of a multi-disciplinary team.

The MuiCSer process, shown in Fig. 1, starts with a user and task analysis to learn about the tasks end users carry out with the existing interface. Other behavioral aspects of the end-user operating the interface are also captured by observations and contextual inquiries. Narrative reports, personas [5] and scenarios typically result from a user and task analysis, while a sensible and correct redesign of a system requires more structured models about the user requirements and the application logic. Existing manuals are also examined to complete the task analysis; these provide a clear overview of what functionality is available and how the supported functionality is communicated to the end-users. It also allows us to filter out the most important workflow patterns supported by the system.

The results of the analysis, obtained in the first stage, are used to progress towards system interaction models and presentation models. We label this stage the structured interaction analysis stage. Models support a combination of user requirements and functional requirements in order to keep track of the application logic during the design and the development of the user interface. These models contribute to low-fidelity prototypes which evolve into high-fidelity prototypes and the final user interface. Both low- and high-fidelity prototypes are often created by designers, thus

tool support is required that checks for consistency with other models, such as the task model while creating the prototypes.

MuiCSer supports the iterative development of systems, including several evaluations, verifications and validations of the artifacts. For the development of complex systems we propose to use a central repository that keeps pace with changes of several artifacts and maintains and labels relationships between artifacts.

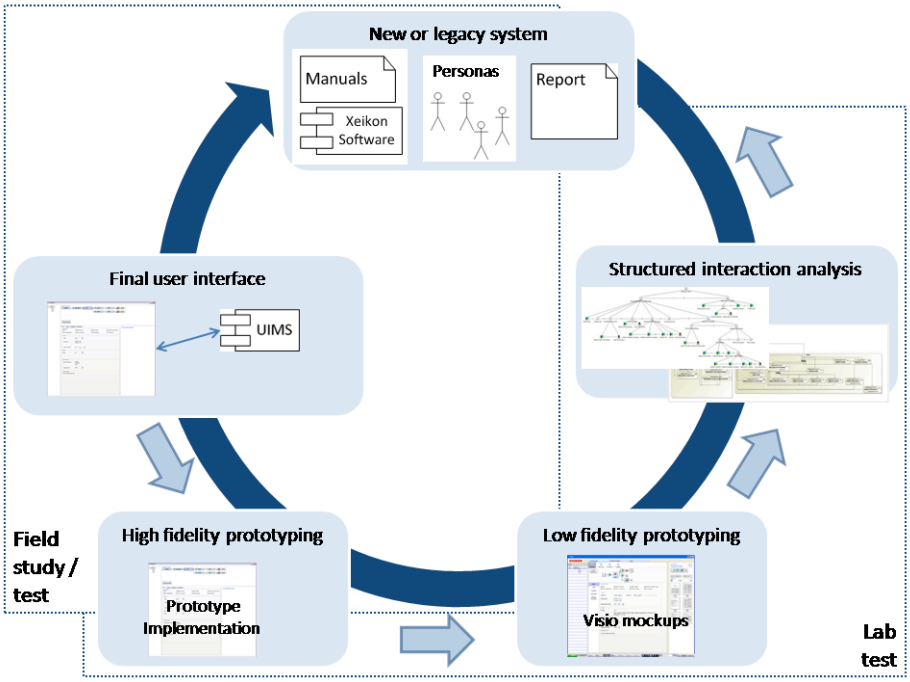

Fig. 1. The MuiCSer framework illustrated using artifacts as they were created in the Warhol project. Note that the visualization of the prototypes and final user interface are intermediate results and do not reflect the actual interfaces.[1]

3 Applying MuiCSer : UI Redesign of a Complex System

The digital printing system being redesigned can be managed using two different applications. Since these applications were extended with new functionality over several years, a redesign of the user interfaces became required to obtain one single coherent user interface and to improve the usability. The redesign cycle started with usability researchers who carried out a user and task analysis. This resulted in a report

[1] The usability researchers opted for mid-fidelity instead of low-fidelity prototypes.

including findings on the observations, a set of personas and a high-level Hierarchical Task Analysis. In this section we describe the path we take to progress from this report toward a more detailed task model, and how this model was and is being used to redesign the user interfaces.

3.1 Creating the Task Model

Based on the report of the usability researchers and the existing software and manuals, detailed task models were created. We use the ConcurTaskTrees (CTT) notation to specify task models with CTTE [3]. CTT supports task hierarchies with different types of tasks, which are related with temporal operators. Since it is unnecessary to detail all the tasks (e.g. extend the task hierarchy), the tasks that are focused on during the first iteration are elaborated. Although the task model contains detailed information of a selection of tasks, a task count of the CTT exceeds 1850 tasks[2]. The leaf tasks in this task model do not always correspond to a single user action. In case of command tasks, a task can be mapped onto a single action. E.g. a task "*Confirm configuration changes*" corresponds to a single action; in this case a button press. Whenever a task involves changing or selecting data, a task might correspond to multiple actions.

3.2 Prototyping

Mid-fidelity prototypes were created by the usability researchers and are based on the results of their user and task analysis, usability guidelines and their experience in the creation of this kind of artifacts. During the creation of these mid-fidelity prototypes, the CTT models were used to determine the completeness and correctness of the prototypes. These prototypes were discussed within the project team and after several reifications of these prototypes they were considered to be stable enough to be turned into high-fidelity prototypes.

The high-fidelity prototypes are created using XAML[3], an XML-based language to describe the user interface for an interactive application. One of the motivations for choosing XAML was based on the rich tool support that is offered, both for designers and developers. These tools enable fast creation of the high-fidelity prototypes using drag-and-drop when possible and custom development for selected parts of the user interface.

To ensure that the created prototypes are consistent and complete with respect to the task model, the XAML describing the user interface controls is enhanced with structured annotations which indicate the corresponding task model. Annotations are inserted in the XAML code that link the parts of the user interface description (subtrees because of the XML-language) with tasks from the task model. Our

[2] These tasks are spread over more than 20 separate CTT files to deal with the complexity and limitations of the tools.
[3] http://msdn2.microsoft.com/en-us/library/ms752059.aspx

approach can be used with other XML-based user interface description languages such as XForms[4], UIML[5] and UsiXML[6].
With these simple links in place, we can build tools that support developers and designers to maintain consistency and correctness between the task model and high-level prototypes. For example, we created a tool that automatically verifies whether the user interface is complete with respect to the tasks from the task model. I.e. all tasks that need to be explicitly presented to the user have a corresponding user interface part in the prototypes. Referring back to Fig. 1, this is one example of information that is maintained throughout the different stages in our process.

4 Toward a runtime system supporting evolution

Because the final user interface will be used by users that have divergent skillsets and that need to perform other tasks, the user interface should be tailored according to the user role. Furthermore, the user interface has to be able to smoothly evolve together with the capabilities of the printing system as well as the changing preferences of the user. One of the challenges is to support this evolution in the systems' user interface and to support changes in design while maintaining consistency and correctness of the user interface. We created a user interface management system (UIMS) that exploits the relationships between the tasks and the user interface descriptions that are created during design. A single XAML-file, stored in the UIMS, describes the contents of a single window (we count each tab as a separate window) and can be linked to multiple tasks. The latter indicates a single window can be used to perform multiple tasks.

The availability of the tasks within a single window as well as the organization and availability of the windows within the user interface of the application is determined by a task model that is associated to a specific user. The concrete visualization of the window structure will be determined by the project team, based on the most appropriate user interface patterns, such as those described by Van Welie [7], and the results of user tests and integrated in the user interface management system.

A user-specific task model will be created in a similar manner as the creation of multi-device user interfaces based on a single task model [4]. One will start from a task model that describes the complete capabilities of the digital printing system that are exposed through the user interface. Tasks that are not relevant for a specific user (or group of users) can be omitted from this task model and will consequently be hidden from the user interface.

Extending the system can then be done by a simple two or three step process; by adding a task to the task model and adding the necessary XAML-file (fragments) and optionally updating the user profiles when the added functionality is not desired for a

[4] http://www.w3.org/MarkUp/Forms/
[5] http://www.oasis-open.org/committees/uiml/
[6] http://www.usixml.org
[7] http://www.welie.com/patterns/index.php

specific user. The system then takes care of the changes in the user interface structure implied by these additions.

Task model relationships specified between the tasks can be used to see the impact of changes in the user interface due to changes in the user profile or offered functionality.

5 Discussion and conclusions

This paper presented our user-centered software engineering process framework, MuiCSer , and showed how it is currently instantiated to redesign a complex user interface into a sustainable user interface. The process makes extensive use of different models, of which the task model is the most prominent during the first stage. The user interface management system, although still very much work in progress, is already used with a selection of models to support consistency and correctness during interface design. The fact that models are reused later in the design process improves the maintenance of these models and eases the evolution of the system since (part of) the design choices are still up-to-date for subsequent iterations.

In our experience this simple yet effective approach to combine several artifacts helps to structure the redesign of a user interface for a complex system, and to support collaboration among different members in a multi-disciplinary team. We were able to communicate clearly to the company's development team how the task model is reflected in the user interface, and how changes in task structure are propagated toward the user interface design. Furthermore, designers were able to create designs with their own tools, in this case Microsoft Expression, and check whether the created designs covered all tasks that need to be supported. This provides us with the necessary means to combine creative activities with the more rigid approach that is typical for model-based user interface design.

Acknowledgements This research was performed in the context of the IWT project Warhol of Punch Graphix in cooperation with usability researchers of IBBT-CUO (KULeuven). The MuiCSer Framework is also based on our experiences in the IWT project AMASS++ (IWT 060051). Part of the research at the Expertise Centre for Digital Media is funded by the ERDF (European Regional Development Fund) and the Flemish Government.

References

1. Mieke Haesen, Kris Luyten, Karin Coninx, Jan Van den Bergh, and Chris Raymaekers. MuiCSer: A Multi-disciplinary User-Centered Software Engineering Process to increase the overal User Experience. In *Proceedings of the 10th International Conference on Enterprise Information Systems*, Barcelona, Spain, June 2008.
2. International Standards Organization. ISO 13407. *Human Centred Design Process* for Interactive Systems. Geneva, Swiss, 1999.

3. Giulio Mori, Fabio Paternò, and Carmen Santoro. CTTE: support for developing and analyzing task models for interactive system design. *IEEE Transactions on Software Engineering*, 28(8):797–813, 2002.
4. Fabio Paternò and Carmen Santoro. One model, many interfaces. In Christophe Kolski and Jean Vanderdonckt, editors, *Computer-Aided Design of User Interfaces III, Proceedings of the Fourth International Conference on Computer-Aided Design of User Interfaces*, volume 3, pages 143–154. Kluwer Academic, 2002.
5. John Pruitt and Tamara Adlin. *The Persona Lifecycle : Keeping People in Mind Throughout Product Design*. Morgan Kaufmann, 2006