
Towards Multi-Level Dialogue Refinement for User Interfaces

Alexander Behring

TU Darmstadt
Informatik, FG Telecooperation
Hochschulstr. 10
64289 Darmstadt
behring@tk.informatik.tu-
darmstadt.de

Andreas Petter

TU Darmstadt
Informatik, FG Telecooperation
Hochschulstr. 10
64289 Darmstadt
a_petter@tk.informatik.tu-
darmstadt.de

Felix Flentge

TU Darmstadt
Informatik, FG Telecooperation
Hochschulstr. 10
64289 Darmstadt
felix@tk.informatik.tu-
darmstadt.de

Max Mühlhäuser

TU Darmstadt
Informatik, FG Telecooperation
Hochschulstr. 10
64289 Darmstadt
max@tk.informatik.tu-
darmstadt.de

Abstract

In this paper, we present our observations regarding modeling multimodal, context-sensitive user interfaces. Based on these observations, we argue to use the “Dialogue Refinement” approach to conjointly describe behavior and layout of user interfaces.

Keywords

UI Models, Refining User Interfaces, Dialogue, User Interface Description Language (UIDL)

ACM Classification Keywords

D.2.2 [Software Engineering]: Design Tools and Techniques---User interfaces; H.1.m [Models and Principles]: Miscellaneous; H.5.2 [Information Interfaces and Presentation (e.g., HCI)]: User Interfaces---Theory and methods

Introduction

With the increasing number of interaction devices, and their everyday-presence, research to facilitate the development of ubiquitous computing applications is gaining more and more interest. Support for building User Interfaces (UIs) for such applications will have to address the diversity of interaction *modalities*. This comprises the multitude of devices that are available for interaction, as well as their corresponding

Interaction Modality: the interaction style, strategy and device used.

interaction strategies and style. As interaction devices are increasingly portable, context dependence of UIs is another important aspect.

In this paper, we focus on these two challenges for UI description languages (*UIDL*): the increasing number of possible modalities and the adaption to various contexts.

Current approaches

Tools supporting development of UIs have been thoroughly analyzed in [1]. The authors conclude that conventional GUI approaches are ill-suited for upcoming UIDL challenges. They see model-based approaches as a promising alternative.

AUI – Abstract User Interface (abstract presentation)

CUI – Concrete User Interface (concrete presentation)

Frameworks for declarative model-based UIs and development environments exist, such as [3] with a focus on GUIs and [4] for plastic (highly adaptive) UIs. They share an understanding of a set of core models: conceptual model, task or dialogue model, abstract presentation (*AUI*) and concrete presentation (*CUI*) model.

In more recent approaches, these models are used to develop multimodal UIs. Teresa [2] is based on ConcurTaskTrees (*CTT*), a task description language. *CTT* models are first transformed into *AUI* models and then further into *CUI* models. Platform specific UIs can be produced by filtering a “master” task model in which the elements are annotated with supported platforms. Further extensions to *CTT* allow context-dependent sub task models.

UsiXML [5], using an extended version of *CTT*, focuses on the integration of a great variety of different target

platforms. It therefore supports a great set of interactor types at *CUI* level. While the Teresa tool and models are centered on the task model, UsiXML focuses on the *AUI* and *CUI* models.

Observations

In the following, we describe observations made with current UIDLs and draw conclusions from these. We then propose an approach called “Dialogue Refinement” to address the issues that have been identified.

Levels of Refinement

In recent approaches, commonly two levels of UI Refinement exist: an *AUI* model is refined to a *CUI* model. Context-dependence of these UI models is mainly achieved via filtering annotated models or via context-dependent task-models.

Model Driven Architecture (*MDA*) proposes a similar approach: to refine abstract to more concrete models, but with no restrictions on the number of refinement steps [6]. Using this idea, an UI for a specific situation can be refined via multiple levels. Hereby, a *situation* especially includes, besides other context information, the modality used and the target platform.

Allowing an arbitrary number of refinement levels, helps the developer to “provide the information at the right level of abstraction”, as advocated in [6]. In the light of increasingly complex challenges when modeling UIs, this gets more and more important (e.g., as depicted in Figure 1).



We conclude to allow an arbitrary number of refinement levels for UI development.

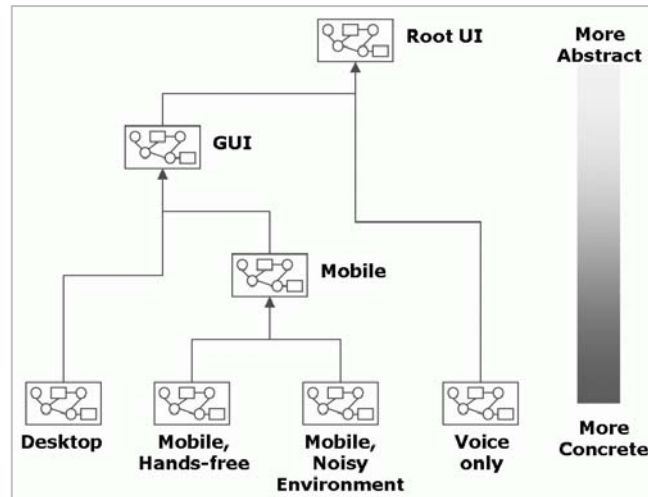


Figure 1: An exemplary tree for AUI Refinement. The abstract “Root UI” is refined stepwise over multiple levels for different situations.

Conjoint Refinement of Behavior and Layout

We observed that behavioral aspects in situation specific UIs are not formulated in a conjoint way together with the UI layout. Task and AUI model can convey behavior information, but behavioral and layout information is not given at the same level of detail and abstraction.

The importance of an integrated formulation can be illustrated by comparing a voice to a direct manipulation (*DM*) graphical modality. The voice modality heavily depends on the behavioral (temporal) aspect, whereas the DM GUI heavily depends on the layout. When trying to integrate the two modalities, both dimensions have to be taken into account at the same level. Otherwise, a gap between DM (layout-

driven) and dialogue (behavior-driven) interfaces opens up.

➔ *We conclude to conjointly refine UI layout and behavior.*

Flexibility of Interactors and Interaction Concepts

To hardcode interactors and interaction concepts into metamodels is common among current approaches. As the metamodel is the central interface, it therewith gets harder to extend the set of supported interactors and interaction concepts for a given approach. Furthermore, Myers et al. [1] state that the currently hard to extend interactor libraries of tools are limiting the developer.

➔ *We conclude not to hardcode interactors and interaction concepts into the metamodel.*

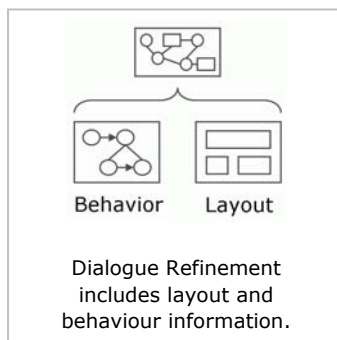
Addressing Platform Specifics

Current approaches like UsiXML and Teresa rather focus on common aspects of UI descriptions for different modalities than addressing the details of a specific modality. In consequence, the developer cannot control the low-level pragmatics of the interactions look and feel. This contradicts the conclusion by Myers et al. [1] that control over such pragmatics is important for the developer.

➔ *We conclude to allow the developer to model platform specific aspects.*

Dialogue Refinement

We propose to address the conclusions described above by using a graph (*Dialogue Refinement Graph*) for UI modeling. UI models are nodes in such a graph and can be refined to other models (nodes). There is no limit



imposed on the depth of the tree. The refinement can be driven by situation changes, which includes changes in platform and modality (i.e. context-dependence).

UI models contain integrated information about layout (e.g., interactors) and behavior (e.g., state charts). Finally, UI interactor types are not hardcoded in the metamodel, but contained in libraries. Additionally, we advocate accompanying type information with ontologies, describing the relations between different types. This can be exploited to support automatic generation of situation specific UIs. Furthermore, we plan to investigate how to describe small variations atop a modeled UI ("*UI Nuances*") that can be reused.

In EMODE [7], we developed the idea of using a multi-level refinement graph, called "*AUI Refinement*" (cf. Figure 1), and integrated it in the UI editor. The node at the top of the tree contains the most abstract UI description, serving as the connection to other models (e.g., application flow). This "*Root AUI*" is refined further for different situations, as depicted in the figure. Further, an interactor ontology and transformations were produced, supporting the developer in refining UIs. Currently, we are extending these results of EMODE with the behavioral aspect, as we discovered its necessity.

Conclusion

In this paper, we introduced the idea of Dialogue Refinement, based on our preliminary work. The presented approach supports refinement over an arbitrary number of abstraction levels, while allowing the developer to still address platform specifics. Finally, it supports classification of interactors with a supporting ontology and conjointly refines UI layout and behavior.

Acknowledgements

We acknowledge funding of this work by the BMBF in context of the SoKNOS and ITEA EMODE projects and thank our colleagues for their valuable contributions.

References

- [1] Myers, B., Hudson, S.E., and Pausch, R. Past, Present, and Future of User Interface Software Tools. *ACM Transactions on Computer-Human Interaction* 7, 1 (2000), 3-28.
- [2] Mori, G., Paternò, F., and Santoro, C. Design and Development of Multidevice User Interfaces through Multiple Logical Descriptions. *IEEE Trans. Softw. Eng.* 30, 8 (2004), 507-520.
- [3] da Silva, P.P. User Interface Declarative Models and Development Environments: A Survey. *Lecture Notes in Computer Science* 1946 (2000), 207-226.
- [4] Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Souchon, N., Bouillon, L., Florins, M., and Vanderdonckt, J. Plasticity of User Interfaces: A Revised Reference Framework. *Proc. TAMODIA '02*, INFOREC Publishing (2002), 127-134.
- [5] Limbourg, Q., Vanderdonckt, J., Michotte, B., Bouillon, L., and Jaquero, V.L. USIXML: A Language Supporting Multi-path Development of User Interfaces. *EHCI/DS-VIS* (2004), 200-220.
- [6] Koch, T., Uhl, A., and Weise, D. Model Driven Architecture. *Interactive Objects Software* (2001). Available online at <ftp://ftp.omg.org/pub/docs/ormsc/02-09-04.pdf>.
- [7] Dargie, W., Strunk, A., Winkler, M., Mrohs, B., Thakar, S., and Enkelmann, W. EMODE – a Model-Based Approach to Develop Adaptive Multimodal Interactive Systems. *Proc. ICSOFT 2007, INSTICC Press* (2007).