Adaptive Dialogue Management and UIDL-based Interactive Applications

Frank Honold, Mark Poguntke, Felix Schüssel, Michael Weber
Institute of Media Informatics
Ulm University
89081 Ulm, Germany
{frank.honold, mark.poguntke, felix.schuessel, michael.weber}@uni-ulm.de

ABSTRACT
Different approaches exist to describe user interfaces for interactive applications and services in a model-based way using user interface description languages (UIDLs). These descriptions can be device and platform independent and allow adaptivity to the context of use, although this adaptivity has to be predefined. In this paper we motivate the use of UIDLs in a broader view: As the basis for building adaptive, flexible and reliable systems using adaptive dialog management. The goal is to provide the user with the right service and the right interface at the right time. We present different requirements for adaptive models with UIDLs and discuss future work to achieve this vision.

Categories and Subject Descriptors
H.5.2 [Information Interfaces and Presentation]: User Interfaces

General Terms
Theory

Keywords
adaptivity, dialogue modeling, multimodality, fission, fusion, uml, uidl, requirements

1. INTRODUCTION
In the vision of ubiquitous computing different services do not only work separately for strictly defined use cases. The user may have a goal that requires the combination of different services. This may not be planned in advance by the individual service providers. Also, different users may prefer different modalities and may approach their goal in various ways. An adaptive interface for different devices and services to the user is desired.

With UIDLs abstract definitions of user interfaces can be achieved to allow a more flexible adaptation to different devices and modalities. However, the adaptation of a user-system dialogue is only possible if all context dependencies and the respective adaptations were defined in advance.

We present requirements to achieve a flexible basis for adaptive dialogue management with flexible UIs based on existing approaches for UIDLs. Major challenges are exemplified with a scenario for the combination of different services to achieve a user’s goal.

2. USER INTERFACE MODELING
Model-based methods are wide-spread for the development process within a software project. The Unified Modeling Language (UML) provides a standard notation that has proven its applicability for modeling different aspects of software in many research and industry projects. For modeling user interfaces (UI) no de facto standard exist. However, model-based user interface development holds the advantages of reusability, readability and easier adaptability of UIs amongst others [14]. Several approaches provide notations that are independent of any concrete implementation and can be used during the process of user interface development. Reported approaches allow user interface designers to specify interaction on a very high-level UIDL without any details on particular input and output devices or platform information. Examples for UIDLs include UsiXML [19], TERESA XML [15] and UIML [17]. These are special notations and require new modeling tools or plug-ins to existing tools to work with them.

Several researchers motivate the use of UML for user interface modeling [7, 6, 5]. De Melo analyzed different user interface modeling approaches and defined decision criteria. UML was identified as most flexible and extendable approach with best tool support [6]. Our approach in [12] builds the basis for further considerations concerning adaptivity. We use the Cameleon Reference Framework [3] and start with a task model created with UML. Figure 1 illustrates the required transformation steps to derive the final UI. Based on pre-defined search criteria UML structures representing specific interaction schemas are then found and transformed to an abstract UI model described in XML. This is used to derive a concrete executable UI model. We illustrated this with the model of a radio application and the derivation of different XAML-based concrete user interfaces [12].

3. SCENARIO
We introduce the need for adaptive dialogue management with the following scenario:
The user is listening to a song played in a radio application on their smartphone. The song seems familiar and the user wants to get some info on the artist. The user wants to get more information on tour dates of this artist. Browsing the tour calendar it shows a concert in the user’s home city in a few months. Finally, the user decides to buy a ticket for this concert.

For all these different tasks a number of services exist the user would have to use separately to finally get the concert ticket. An intelligent system could use adaptive dialogue management to assist the user in succeeding their goal by automatically offering the different services when needed. This can be facilitated by the use of UIDLs providing abstract interfaces to the services. Figure 2 illustrates this within our given scenario.

Furthermore these interfaces could even be adapted to available devices and modalities in a given context. In the following, adaptive dialogue management is introduced. Then, requirements for adaptive dialog management for UIDL-based interactive applications are derived.

4. ADAPTIVE DIALOGUE MANAGEMENT

Continually available systems like smartphones or in-car infotainment systems are ubiquitous in nowadays society and their individual success is to a great part based on user-friendly interface design and flexible integration of diverse applications. As these systems are usually already configurable in their look and feel and may be adapted individually, the next step in this area is adaptivity at runtime to the current state of the user (e.g. the emotion, the situation, his expertise) and their surroundings. Adaptivity should not only be restricted to a single application, we call this *intra-application adaptivity*, but should span all applications available which we call *inter-application adaptivity*.

Such systems would not only combine the information gathered by multi-sensor networks, but could also extend the interaction possibilities for the user to all devices capable of interaction. This allows for a more natural way of interaction and does not constrain the user to a predefined limited set of modalities. The basis for such kind of systems is an adaptive multimodal dialogue management that is able to manage multiple applications (or parts of a single application) as well as different devices for input and output. For this purpose the underlying dialogue model needs to be at a high level of abstraction, so that dialogues can be invoked independently of their concrete user interface, depending only on their purpose for the user.

A lot of work has already been done on this topic regarding multimodal dialogue system architectures [4, 13, 9] and complete systems like in [20, 11]. We propose a goal based dialog management in the style of spoken language dialog systems (SLDS) that can be easily modeled using a graphical tool [1].

In our scenario a computer system assists a user in succeeding his goal. The envisioned system owns a model of the state of the world as it is and a model of how it should be. The system’s AI planner owns the ability to identify and structure abstract tasks from a given service repository in that way, that it generates a planned sequence of certain tasks ordered by causal links to achieve the aforementioned goal (cf. [2]). These tasks are then passed over to the dialogue management (DM) as abstract dialogue acts, which need to be realized via a suitable user interface. This could be done with the use of UIDLs. The DM identifies all services which can realize the given task. Each service provides its own UIDL description. So in co-operation with the interaction management (IM) the DM decides for one service to be integrated. The service gets integrated into the current dialogue sequence and the IM is responsible for the UI-refinement including fission and fusion. The IM looks up all available interfaces, or even compositions of such interfaces to offer the right interface at the right time (cf. Fig. 3).

5. MODALITY ARBITRATION

For dynamic use in ubiquitous environments novel systems will co-operate with capable devices and make use of their different input and output components. Starting from
abstract UI descriptions the interaction management (IM) shall derive an appropriate UI by reasoning about unimodal or multimodal output and the corresponding concrete UIs. Throughout the process of modality arbitration the IM has to get knowledge about the later user interface’s interaction concepts, the interaction interfaces. The IM analyzes the UI description, and explores all capable components which support the needed interaction for input and output. The IM’s fission component is capable for output organization. If not yet modeled, a concrete UI will be derived from the abstract UI description within the fission process, as motivated by [3, 7]. The fusion component analyses the occurring interaction input and checks, if it could be mapped to the UI’s given interaction interface. If possible, the interaction gets assigned and passed back via the dialogue management to the linked service or application.

6. REQUIREMENTS

As motivated above future systems shall compose the user interface at runtime in a very flexible way. The adaptiveness of the single UI components as well as their combination is influenced by the user model, the device and components model, the surrounding model, the task model, the available widgets, and the information which shall be communicated.

To reason on a concrete user interface rises requirements for a UIDL-concept, which offers the possibility to describe the UI on different abstraction levels. The (device- and modality-independent) abstract level for the dialogue management and the concrete description to realize the UI. To identify suitable pre-described widgets, each widget has to provide information about its purpose, its needs for and effects of a possible applicability. To seamlessly interact with a realized UI via different interaction metaphors (speech, touch, gesture, etc.), the UI description must reveal a description of its interaction interface.

The next issue addresses multi-tasking and interaction mapping support. In our scenario we described the situation where the user listens to the music and asks for the artist. The dialogue manager handles the w-words\(^1\) interaction, if the current application can not do this; here: “Who is the artist playing on the radio?” So the system identifies a new goal, the planner integrates it, because the system retrieved a suitable application. Next, the dialogue manager integrates the service at runtime and the interaction management is responsible to communicate the music database’s UI towards the user, and so on (see Fig. 2).

There may be temporarily at least two concurrent applications running in parallel. Due to the UIs described interaction interfaces and the fusion’s interaction mapping (cf. sec 5) different input components can be assigned to different applications if they support the same interaction for a user input. This goes along with some of the needs for interaction modelling named in [8, 16].

We summarize the requirements for future UIDLs for a better applicability in dialogue management and modality arbitration as follows:

- A UIDL-concept shall allow to describe user interfaces on different levels of abstraction (including concrete and abstract layout or arranging descriptions)
- There shall be a way to describe a UI in a device- and modality-independent way
- The UIDL-concept shall allow to describe the purpose of a modeled UI
- It shall be possible to describe a UI’s needs for applicability as well as its resulting effects
- UIDLs shall allow to describe interaction interfaces for seamless interaction integration with different devices and components
- UIDLs for core-interaction have to provide binding mechanisms to link the UI with the functional core
- To support interaction, event mechanisms shall be describable
- The UIDL shall allow to describe reusable components for different purposes
- Information – expressed or gathered via a user interface – shall be describable or referenceable by the expressiveness of a UIDL. To allow to model an abstract UI for abstract information and a concrete UI for concrete information.

It is unlikely that a single UIDL will evolve for all kind of applications, since there are different fields of application. According to [18, 18] there are differences between UIs for “command/control oriented interfaces, where the user initiates all action” and an interface which “is more modeled after communication, where the context of the interaction has a significant impact on what, when, and how information is communicated.” Thus a system can be designed using an interaction based or an information based approach.

The requirements for multimodal interaction modeling (cf. [8]) are different to the one of user interface modeling. We

\(^1\) who, what, when, where, why, which, whom, ...
understand UML as a modeling language which offers possibilities to model both. In [5] we focus on interaction modeling, whereas in [12] we focus on user interface modeling, both utilizing UML.

7. CONCLUSION AND FURTHER WORK

We illustrated important challenges with future adaptive dialogues in combination with adaptive user interfaces. The approach to add adaptivity to an overall user-system dialogue with different services described with UIDLs implies different requirements. The derived requirements have to be an essential part of future user interface descriptions and can be fulfilled by different types of UIDLs. We highlight the need to describe the UI on different levels of abstraction, device- and modality-independent, and the addition of different goals a user interface, respectively the underlying application, can fulfill. The work on adaptive dialogue management for abstract user interfaces is currently ongoing. UIDLs have to meet different requirements whether they shall support an interaction based or an information based approach. The requirements shall be used as basis for further discussions and push future research to achieve adaptive, flexible and reliable systems – so called Companion Systems.

8. ACKNOWLEDGMENTS

This work is originated in the Transregional Collaborative Research Centre SFB/TRR 62 "Companion-Technology for Cognitive Technical Systems" funded by the German Research Foundation (DFG).

9. REFERENCES


