Context Models for Adaptive Dialogs and Multimodal Interaction

Frank Honold, Felix Schüssel, Michael Weber
Ulm University
Institute of Media Informatics
89069 Ulm, Germany
{frank.honold, felix.schuessel, michael.weber}@uni-ulm.de

Florian Nothdurft, Gregor Bertrand, Wolfgang Minker
Ulm University
Institute of Communications Engineering
89069 Ulm, Germany
{florian.nothdurft, gregor.bertrand, wolfgang.minker}@uni-ulm.de

Abstract—This article presents a context adaptive approach for multimodal interaction for the use in cognitive technical systems, so called companion systems. A system architecture is presented and we clarify where context awareness occurs on different levels with a layered context model. The focus is on the topics of dialog management, multimodal fusion, and multimodal fission, as the main participants in interaction. An implemented prototype is presented, yielding some concrete instances of the described context models and the adaption to them.

I. INTRODUCTION

When envisioning future technical systems there is a common ground for the main characteristics needed: Future technical systems shall be continually available, cooperative and reliable assistants which provide their functionality in a completely individualized way [1]. Users should trust these systems and experience them as competent and empathic partners. The three main abilities to realize this vision are:

Recognition and Cognition: To possess the ability to recognize the surroundings and the user’s situation (Recognition). Build a cognitive model of the user to make inferences on the user’s intentions and emotional state (Cognition).

Planning and Decision Making: To possess the ability to plan a sequence of steps to realize the user’s intention (Planning) depending on the surroundings, the user model and the situation he is currently in (Decision Making).

Interaction and Communication: To possess the ability to help the user realize the intention in the most individualized way possible by adapting the interaction during runtime to his capabilities, preferences, characteristics and taking into account both the situation and the disposition of the user as well as monitoring the interaction and the user’s reactions in order to learn from it.

Considering and integrating these abilities for cognitive technical systems results in what we call a Companion System, due to the fact that accompanying the user is one of the key factors in this new distributed computing paradigm. Based on the evolution chain of distributed computing paradigms by [2], there are several additional criteria which qualify a technical system to be a companion. According to [3], the evolution chain can be extended by intention-awareness, artificial intelligence planning and adaption by learning to characterize Companion Systems (see figure 1).

Fig. 1. The evolution chain towards Companion Systems [3]

An abstract architecture of such a companion system is depicted in figure 2 and consists of the following functional parts:

Fig. 2. Architectural overview of Companion Systems

The environment recognition consists of several sensors and sensory fusion systems that capture the surroundings and the user’s situation and provide their results to the knowledge base. Within the knowledge base, inferences on the user’s intentions and emotional state are made and the overall world state is maintained. This information is accessible by all other parts of the companion system. The application core provides the mere functionality, that the system is capable of, for example an online shopping system or a calendar application. The task planning module is an artificial hybrid planning system [4], which provides a solution for a given problem. This problem usually solved by an individual arrangement of different tasks a user has to accomplish. The sequence of these abstract tasks is then forwarded to the dialog management, which decomposes each of them into concrete and individual dialog goals. This decomposition is done in an adaptive way, so that task-independent events like misunderstandings or additional question and answer dialogs can properly be handled at runtime. The dialog management passes each communicable dialog fragment as communicative acts to the multimodal fusion module. Within this module, the available modalities are assigned in a context aware manner and output device components are finally utilized for the presentation to the user. When the user performs an input, it is captured by input device components and transferred to the multimodal fusion module,
which generates an abstract input resulting from all observed modalities using an adaptive fusion process. The input is then passed back to the dialog manager for dialog goal evaluation and if necessary, the task planning module is notified of a success or failure of the current task.

This article focuses on a companion system’s ability of Interaction and Communication. Thus, we discuss adaptiveness to context information of the dialog management as well as the multimodal fission and fusion modules that play the major part during the interaction with the user. We discuss related work and present a layered adaption framework where context awareness occurs on different levels. We name the different levels and describe which context models can be used for adaption on each distinct level. We present a concept for context modeling and processing within the different levels of our approach. Finally, we describe an implemented prototype of an assistive system, showing some concrete examples of used context models.

II. RELATED WORK ON CONTEXT-AWARE COMPANIONS

Context awareness is described by Dey and Abowd as follows: “A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user’s task” [5]. The modeling and use of context information in the field of informatics dates to the early 1960s [6]. Many different approaches exist dealing with “how to model context” [2]. The different approaches are caused by diverse requirements of context aware systems and the systems themselves were originated to suit different purposes. In the remainder of this article we therefore focus on the needs related to the domain of companion systems and take a look on two related projects: SmartKom [7], and MuDiS [8] from the research cluster CoTeSys1.

The SmartKom project is an often cited system which offers multimodal interaction possibilities and adaptive dialog behavior. When interacting with SmartKom a user can interact via a virtual communication assistant using speech, gestures, and mimics. The system itself responds to the user and the digital avatar Smartakus presents the system’s output. This kind of interaction, by means of delegating the virtual assistant, is called situated delegation oriented dialog paradigm (SDDP).

The MuDiS project presents a software architecture for a multimodal dialog system that is extensible and applicable to a wide range of applications. Its multimodality is supposed to not only be restricted to classical input modes like speech and gestures, but shall also integrate novel input channels like emotion recognition and person tracking.

A. Dialog Management

In SmartKom the dialog management (DM) is used to implement the system’s personality (i.e. the avatar Smartakus’ behavior) [9]. The DM is a plan-based reusable component serving 14 different applications within the SmartKom complex. Thus, it delegates the applications’ interaction possibilities via the UI towards the user and triggers functional interactions in turn.

1CoTeSys homepage – http://www.cotesys.org/ [2013-02-06]

In the MuDiS software architecture a dedicated dialog manager is used to keep track of the current state of conversation and infer actions that need to be executed next. For this purpose it is based on a state machine and a knowledge base implementation in Prolog. The state machine consists of fixed, abstract dialog nodes like confirm, ask knowledge base and output, which realize an application independent, albeit simple dialog. The concrete dialog steps are directly generated as results from queries to the knowledge base. Concerning the dialog management, there is no indication of some form of adaptivity to context.

B. Multimodal Fission

The topic of multimodal fission encompasses the two problems of (1) how to segment certain information items and (2) how to communicate these information fragments via different output components.

In SmartKom the user perceives the system’s output in two ways: the guiding avatar Smartakus, and the system’s functional output serving user requests, e.g. a TV schedule or a map [10]. Thereby the functional output is rendered using pre-defined widget containers, which can be adapted to suit given information (e.g. by not using some UI elements of a certain form widget). The main multimodal interaction element is Smartakus. Its multimodal output is based on pre-defined mappings of speech and facial expressions, whereas its cross-modal behavior for deictic references depends on the given layout situation of the functional output. To realize context adaptivity pre-defined XSLT style sheets are used for three different situations (home, public, and mobile), to transform a given abstract system output into the final UI.

The MuDiS system seems to rely on predefined ways of system output (text-to-speech and robot movements have been reported), although the architecture itself could make use of a more sophisticated output generation module as stated by the authors. If this includes some form of adaptive output generation remains unclear.

C. Multimodal Fusion

SmartKom is able to incorporate and fuse speech, gesture and/or facial expressions for interaction [11]. The system performs an early fusion to combine different mimic features (eye, nose, mouth) [7] and a late fusion to combine the different input modalities (speech, gesture and/or facial expressions). The interaction patterns for fusion are directly modeled in algorithms. The system’s modality fusion approach relies on knowledge about the system’s current location as well as the dialog’s discourse model for the resolution of expressions to the situational context.

Within MuDiS, multimodal fusion is done as a late fusion approach, where already characterized events from different input modalities are combined to semantic representations that serve as events for the dialog manager. This approach allows for modality abstraction and can therefore be used with any form of modality, although only speech and simple head movements (nodding and head shaking) have been implemented so far. There seems to be no form of runtime adaptivity to context information.
Systems known for several years as well as current approaches form a solid basis for further developments in the field of companion technology. The engineering foundation is set to build systems which apply context adaptivity in a concise way on different levels. In the remainder we give an outline how dialog management and multimodal interaction can benefit from which context parameters to realize an overall adaptivity in the human computer interaction cycle.

III. CONTEXT AND ADAPTATION LAYERS

In accordance with Coutaz et al. [6] context is more than a simple state description, but rather is part of a process. The involved fragments of gathered information may be interpreted in a different manner each time they are examined. Coutaz et al. motivate this with a printer example. A set of documents shall be printed. There are different printers in a certain location where the user can move around. The challenge is the determination of the correct printer on which each document will be printed. The given example’s essential information depends on knowledge about the user’s meeting schedule. Another example might be a simple thumb up gesture recognition. It may have different meanings: okay or hitchhiking. Again, additional information leads to the correct meaning. To solve such a syntactical classification problem a “holistic” treatment of context is recommend [6].

In Strang and Linnhoff-Popien’s survey on context modeling [2], the authors identify six demands implied by ubiquitous systems. For our approach we assume a knowledge base, available at all times, meeting the six demands on context modeling identified by [2] in the following way:

**Single Point of Information** The information based system is realized via a central knowledge base (KB). The KB maintains the distributed composition and administration of context information.

**Valid Information** The KB ensures the validation of distributed and/or logically combined context information.

**Quality of Information** Each information provided by the KB is annotated with an additional value expressing the information’s plausibility in our probabilistic approach.

**Closed World Assumption** The KB’s available information may be derived from different sources. After an information fusion process we assume the KB’s information as valid and unambiguous.

**Level of Formality** The communicated contextual facts in our approach rely on a scheme-based definition of context information. Each participant in our scenario shall be able to understand the information.

**Applicability to Existing Environments** In our current approach the KB is not connected to other existing open ontologies, open knowledge bases or web services, but it would be possible to do so.

Different aspects of context knowledge are used in the diverse layers of our layered adaptation framework. The utilization of context knowledge on different levels leads to a better overall adaption as described in the remainder of this section. Figure 3 shows the layered adaptation framework for multimodal systems.

The information, which builds the context in our approach, is composed of eight models (cf. gray boxes in fig. 3). Each model represents knowledge which can be utilized for adaptation on different levels in a human computer interaction cycle. The models are:

- **Application Model** The application model describes the interaction possibilities offered by the application domain, e.g. the play and pause methods or the volume setting known from diverse media applications.

- **Task Model** User’s tasks (current ones as well as intended ones) are modeled in a hierarchical task model.

- **Environment Model** The environment model is used to describe context information e.g. the level of noise or lighting condition.

- **User Model** This model’s information describes users properties like location, preferences, handicaps, name and so on. In addition we advise to use this model to provide user specific information about each user’s current action or working domain to support the information mapping in fusion and fission.

- **Dialog Model** The dialog model describes the tasks the user has to accomplish in co-operation with the system. It contains abstract tasks as well as their dissection into dialog acts.

- **Information Model** This model provides information representing communicable content. It represents the attributes of information objects as well as their relationships to other information objects. Due to its modality independent modeling information may be provided in a redundant way (e.g. picture plus textual description).

- **Presentation Model** This model holds information on the current presentation of the system, i.e. information about what is presented where and how. The presentation model

---

2 also known as domain model or concepts model
3 also known as surroundings model
is updated as soon as the fission module finished on deciding which information is to be presented in what kind of way.

**Component Model** This model is used to describe information about the available input and output components provided by different devices, e.g. screen size and resolution. In addition this model describes which UI concepts can be used by which component.

In the remainder of this section we describe the different layers which are involved in the human computer interaction cycle and how they are affected by these models.

### A. Dialog Management

The purpose of the dialog management is to control the human-machine dialog to fulfill a task the user has to accomplish (e.g. to wire up a home theater system). From the dialog manager’s point of view the AI planning module provides an abstract description of dedicated action steps, which are needed in order to successfully manage an envisioned task. However, this abstract description of needed steps is not detailed enough for a human-computer dialog, because the dialog has to instruct the user how to accomplish each single task step in a personally tailored and adapted way. The dialog should be adaptive during runtime and react properly to events like misunderstandings or interposed questions. It shall also adapt to the user’s emotions or the user’s expertise level if such kind of context knowledge is present.

The dialog manager decomposes each provided plan step one by one during runtime into a hierarchical dialog structure which consists of so-called dialog goals (Node D is decomposed in fig. 4). The nodes in such a structure represent more detailed descriptions of its initializing plan step but are not directly executable. The leaves in this structure (a, b, c, d in fig. 4) are modality-independent executable dialog goals, which can be communicated towards the user by passing them as dialog output (leaf a in our example) to the fission manager for further processing. The main part of the output is the so-called dialog act.

The adaptivity requirements impose a high demand on the dialog model in size and require the dialog control to be efficient in deciding which adaptive part of the dialog should be chosen for each dialog step depending on contextual and situational information.

The dialog management receives user input via the fusion module, interacts with the application, keeps track of recent dialogs in a dialog history, chooses the best dialog strategy and decides on the best response to the user. The most important factor, influencing dialog strategies, is the initiative. In a system-initiative dialog the user is guided by the system from one dialog step to another. In user-initiated dialogs the user can pose any question or request and the system tries to answer. In the mixed-initiative dialog strategy the system initiates the interaction. However, the user can interfere by barging in and trying to lead the system to his desired directions. The main characteristic is that the initiative will shift back to the system and vice versa during the ongoing dialog.

Deciding on the best response to the user is on the one hand selecting the best type of response but on the other hand selecting the best content for this response. The selection of appropriate content depends e.g. on the current knowledge of the user concerning the response. Every user starts with a common sense model, which defines what knowledge he probably already possesses. However, this model will change over time. Every time the user manages to accomplish a task or gets an explanation from the system, this information is registered in his profile. We differ between procedural knowledge (how to do something) and conceptual knowledge (what is something). Additionally, the knowledge fades away over time in our user model. This means that if a conceptual explanation about the cable format HDMI was presented to the user five minutes ago, we will reduce the explanation complexity about HDMI if it appears in an explanation. However, if it appeared in an explanation five days before, we will use the default explanation complexity again.

Managing the dialog involves several decisions which depend on the aforementioned models in the following way:

1) **User Model**: The user model is extremely important for the choice of the sequence and the content of dialog outputs. The dynamic knowledge model of the user is included in the user model, which will decide on the sequence and number of dialog steps to accomplish the current plan step (e.g. being an expert in connecting devices will result in a reduced instruction sequence for a matching abstract dialog goal). Additionally, the user knowledge will influence the content generation of responses which is stored in the information model (e.g. if a user is not familiar with the concept of HDMI, he will receive a conceptual explanation about HDMI).

2) **Dialog Model**: How to adapt to information contained in the user model is being modeled in the dialog model. So-called guards serve as constraints which have to be fulfilled in order to tackle the associated dialog goal. Based on these guards, complex adaptivity decisions become possible. Typical guards in our dialog model are the disposition of the user (i.e. positive or negative mood) or the knowledge of the user (i.e. conceptual or procedural). The result of a dialog step is called effect, which can fulfill guards of subsequent dialog steps.

3) **Information Model**: The generation of dynamic parts of the content of the information model is done in the dialog content generation. Especially the text of explanations is
adapted to the specific knowledge of the user. Due to the user’s domain knowledge modeled in the user model being dynamic, the information model only references information by information IDs.

B. Multimodal Fission

The system’s fission reasoning process is about how to combine information, widgets, and output components in an adequate constellation to communicate a given modality-independent dialog output received from the dialog manager. By reasoning about the adequate user interface, the fission transfers the dialog output to a modality-specific interaction output.

This process depends on some of the aforementioned models and their represented information. The (often sensor based) information has to be treated with the awareness, that the data might be defective. Therefore, we assign probability values to each information fragment within our knowledge base. Using such a knowledge extension, the system can perform probabilistic reasoning to determine the fission’s output. This in turn ensures the approach’s functional capability even with knowledge based on ambiguous multi-sensor real-world data. The reasoning approach is based on a set of evaluation functions. Each evaluation function comes with a reward. To infer the proper output each permutation of possible output combinations is judged by each reward function. The best performing constellation is applied for output. The interaction output is described in a presentation model which can be used within a fusion process to resolve references made by the user [12]. The fission’s reasoning approach for modality arbitration as well as its realization is described in more detail in [3]. The reasoning process is influenced by the following context models (cf. fig. 3):

1) Application Model: The application model is used to characterize a system’s functional interface. It can be expressed using UML to derive proper user interface elements as described in [13]. For a modeled media player, the fission’s reaction to changes in the application model results in an adapted UI, e.g. by adding additional interaction features for user inputs are given by the current dialog goal. The fusion module has to infer the adequate modality specific interaction output for each dialog fragment. The reasoning process also tries to respect the wishes for nominated output devices or channels. The fission module’s decisions are based on the following models.

2) Dialog Model: The dialog manager’s modality-independent dialog output serves as the fission’s main input. The fission module has to infer the adequate modality specific interaction output for each dialog fragment. The reasoning process also tries to respect the wishes for nominated output devices or channels. The fission module’s decisions are based on the following models.

3) Information Model: In our approach information is modeled independently from the dialog and application model. Information is referenced by an information id and can be of different types (audio, text, and picture). Due to the fission process, the UI is composed at runtime to communicate the available information, matching the constraints implied by the user and the situation. Another adaption is based on the privacy of information. If marked as private, such information is only communicated via device components that ensure privacy (e.g. PDA screens or in-ear speakers).

4) Environment Model: Different constraints from the environment can affect the reasoning process for multimodal fission. This model, e.g. may be an indicator for a silent working domain, like a library. In such a case, the UI will be adapted in the way that no disturbing acoustic output is generated. In addition, the fission process takes care about sensory data, which may also lead to an adaption of the UI (e.g. the number of present persons in combination with information marked as private).

5) User Model: Motivated by user centered design, the user model is seen as the essential model for UI adaption. As an example, if there is an indicator that the user is not able to perform a touch interaction (see fig. 5), the fission’s reasoning takes this into account and downgrades all interface possibilities where touch is the only input concept. As a result, the system may offer the user an adequate input possibility with an additional or exclusive speech interface.

Fig. 5. Excerpt from a user model. The user model shows a high emotional arousal. The user is rather not manual available for touch interaction. (Maybe the user is driving a car right now.)

6) Component Model: In ubiquitous scenarios the devices with their available components may permanently change, and again the fission’s reasoning process evaluates the current situation and adapts the UI according to the available devices’ properties as well as their varying distance relation to the user.

A probabilistic fission approach is suitable to handle uncertain context knowledge. In combination with the flexible expression based reasoning, modality arbitration can be performed even in smart environments with fluctuating device components. As a condition the components have to provide their component model and have to own the ability to interpret and render the fission’s interaction output, the presentation model.

C. Multimodal Fusion

The purpose of the fusion module is to map user inputs to their semantic representations that can be handled by the dialog management. Therefore all possible semantic representations for user inputs are given by the current dialog goal. The fusion itself takes place at a semantic level, i.e. the inputs for the fusion module are already categorized events from the input device components. Currently we make use of touch, speech and pointing gesture inputs.

The input fusion must handle unimodal and multimodal inputs that must both be mapped to given semantic representations. For the semantic mapping of unimodal inputs, an
approach similar to that of the MuDiS system is used. That means, all possible variations of an input in a single modality (esp. speech) are predefined and map to the same semantic representation. For example, if the user utterances “I want number two”, it gets mapped to the semantic representation “SELECT(2)”. The same mapping is applied for utterances like “number two” or just “two”. For the purpose of fusing multimodal inputs, e.g. speech and pointing gestures, an approach relying on evidential reasoning as described in [14] is used, which relies on an abstract interaction representation of combined events. This way, cross-modal inputs like deictic references in the form of the utterance “this one” together with a pointing gesture at a particular object on the screen can be resolved.

To advance current approaches of input fusion, knowledge from the following context models (cf. fig. 3) can additionally be used to increase the quality of the multimodal fusion in terms of robustness and accuracy:

1) Environment Model: Information about the current environment can be used to judge the reliability of the available input modes. For example in a noisy environment, speech input can be assigned a lower overall reliability than gesture input and therefore gestures are preferred in case those two modalities deliver contradictory events. These reliabilities can be applied in addition to uncertainty levels already provided with the events from the input components. The type of environment, public or private, also has an impact on the use of input modalities [15], that can also be used for likelihood and reliability estimations.

2) User Model: Information about the user can be used in several ways. Static a priori information on the gender or the user’s personality can be used to judge the likelihood of uni- and multimodal inputs. A recently conducted Wizard of Oz study suggests momentous influence of these parameters on the way users perform basic interactions (e.g. selections on a GUI) [16]. Again, reliabilities of different input modalities can be adapted, this time in a user individual manner. Dynamic information about the user by means of his current emotional state can be used to get early hints on possible misinterpretations, e.g. when the user signals surprise by nonverbal audio cues or shows a corresponding mimic expression. This way, implicit information can be used to proactively activate a clarification dialog without the need for the user to explicitly express his disapproval.

3) Presentation Model: Beyond the obvious use of the presentation model for resolving pointing gesture references and giving information about expected words in verbal utterances, it can also be used to alter likelihoods of input modalities. Our aforementioned study in accordance with a previous study by Bellik et al. [17] suggest, that there are also significant impacts of the presentation on the way users interact with a system. E.g. the likelihood of verbal input also increases, when the system has just used a verbal utterance.

IV. IMPLEMENTED PROTOTYPE

The presented adaptive multimodal framework for companion systems was implemented in a scenario in which a companion system assists a user in planning a move, and setting up a new home theater system for the housewarming party.

On the technical side we are using a modular architecture. The communication between each module is done by using a platform-independent middleware. This secures a trouble-free communication between consortium partners using different programming languages or operating systems. The message oriented middleware is realized using the SEMAINE API [18]. Semaine is an open source framework for building emotion-oriented systems. The communication in our system is most of the time based on XML.

As outlined in the theoretical work (see III-A), the course of abstract tasks the user has to accomplish, represented by the task model, is arranged by an intelligent artificial planner [4]. These abstract tasks (e.g. buy snacks, inform the neighbors, set up home theater) are passed sequentially to the adaptive and assistive dialog management which opens the matching dialog model and decomposes these tasks during interaction into smaller, for the user individualized dialog goals. An exemplary dialog model from our current implementation, which includes an adaption to the pleasure level from the user model, is visualized in figure 6.

![Graphical representation of an exemplary dialog model](image)

**Fig. 6.** Graphical representation of an exemplary dialog model, where two alternative sub-goals can be used to realize a cable selection. The realization depends on the user’s pleasure level referenced by the guards. Each of the different templates reference individual information items, each with an emotion-specific verification. This allows to deal with different emotional states of the user.

To control the decomposition we use a constraint programming approach [19], which has proven to be especially useful in problems on finite domains where many conditions limit the possible variable value configurations [20]. It is a technique to find solutions to problems by backtracking and efficient reasoning. As a dialog in our case is limited to a certain number of possibilities how the system can interact with the user, it is reasonable to use a finite domain for the variables that constrain the execution of the dialog goals. These conditions make the dialog control suited for applying a finite domain constraint solving mechanism. This means that the
sensors’ values are mapped to a finite domain on which a constraint solver is able to find out the variable configurations that are possible in the dialog at a certain time. By knowing the variable configurations which are possible at a given time, it is clear which dialog goals are executable because these are directly related to the variable conditions. For example, if the user is in a stressed condition, our prototype will reduce the decisions the user has to take to a minimum. This will result in a working home theater configuration, which may be not perfect for the individual user, however working to throw the housewarming party. The resulting dialog acts are then analyzed in terms of their required knowledge [21], and if necessary additional dialog steps are included to prepare the user for the upcoming task. For example, if the next dialog goal is to connect the satellite receiver and the TV with a HDMI-cable, but the user’s knowledge on HDMI is low, an additional dialog goal explaining HDMI will be included in the course of the dialog. This analysis is based on the user’s knowledge contained in the user model (see fig. 7).

The dialog output, representing the current dialog model (see fig. 8), is then sent to the multimodal fission module via the SEMAINE middleware. The fission component is responsible to explore and judge each possible output configuration as described in section III-B, based on the knowledge from the information, the environment, the user, and the available component models. A possible available application model is inspected and results e.g. in additional interaction offers to control a displayed video, if necessary.

At the end of the reasoning process the best rated output configuration is the basis for the model-to-model transformation from the abstract dialog output to the concrete interaction output. Along with that, each abstract information item is replaced by its deduced concrete information representation. The information mappings are pre-defined in the information model for each referenceable information object. Then the interaction output (cf. fig. 9) is broadcasted to each connected device, where an interpreting client runtime processes the output description and realizes the final UI using the devices’ addressed components. Figure 10 shows the resulting user interface for the cable selection task on a large scale screen. After the fission reasoned about the concrete output configuration, the resulting interaction output (see fig. 9) is used as the presentation model by the fusion component. Containing all possible (abstract) actions a user can perform, and their representations on specific devices, an internal model for all possible interactions is created from it as described in [12]. Received sensory inputs can be fused, as described in [14].

![Fig. 7. Excerpt from the user model showing the user’s overall knowledge, where the user is a novice regarding the HDMI concept.](image7.png)

![Fig. 8. An exemplary abstract and modality independent dialog output. The output contains a topic, a dialog act with an selection, and a demand to realize a listening concept for possible user requests. The dialog's control flow is influenced by the object IDs. The interaction’s information flow is defined by the information IDs.](image8.png)

![Fig. 9. Excerpt from the interaction output based on the dialog output from Figure 8. Referenced media can be linked relative or via absolute URIs.](image9.png)

Some of the mentioned models (like user and environment model) can not only be influenced by sensory data, but can also be directly manipulated via a tablet device, in order to easily create specific constellations.

Author pre-print version. Please check the IntEnv 2013 proceedings for the final version.

IE’13, July 18-19, 2013, Athens, Greece
V. Conclusion

We introduced a cross-disciplinary research effort towards the development of context aware companion technology. We discussed interactive companion-like systems and presented our adaptive approach for context-sensitive dialog management and multimodal interaction. Thereby, we presented an evolution chain towards Companion Systems and developed a layered adaption framework for multimodal systems. We motivated context awareness on different levels and demonstrated how contextual knowledge can be modeled and utilized for adaption. A prototype implemented in our collaborative research laboratory demonstrates the basic feasibility of the approach, though not all mentioned context models are fully utilized as described in the framework. As the case may be, there will never be a single system using all models and adaptions of the framework. But never the less, it provides a comprehensive overview of adaption possibilities of modern, model-based systems.

Acknowledgements

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).

References


Fig. 10. A user interacting with the prototype of a companion system within our research lab. The context models utilized for the system’s permanent adaption are influenced by environmental sensors or, for test cases, via changing parameters on a simulating tablet device.