SiAM-dp: A Platform for the Model-Based Development of Context-Aware Multimodal Dialogue Applications

Robert Neßelrath
German Research Center
for Artificial Intelligence (DFKI)
Saarbrücken, Germany
Email: robert.nesselrath@dfki.de

Michael Feld
German Research Center
for Artificial Intelligence (DFKI)
Saarbrücken, Germany
Email: michael.feld@dfki.de

Abstract—**Intelligent Environments are highly interactive by integrating information and communication technology into the physical space. One goal is to provide user interfaces that are adaptive to the user and the environmental context, including the communication modalities. We present a new development platform for multimodal dialogue systems. A development approach based on semantic-models supports the creation of situation-aware dialogue applications in a declarative way.**

Keywords—**HCI, multimodal dialogue system, development toolkit, context adaptation, intelligent environment**

I. Introduction

Intelligent environments and ambient intelligence enhance natural environments of the daily life such as homes, offices and cars with the capability to make the complete environment highly interactive and user-friendly. The interconnection between a variety of computing units and electrical devices allows the environment to appear like a collective intelligence and to adapt and react to actual issues, that are triggered either proactively by the system or by persons that interact with the environment in a certain way. Three paradigms form the foundation for this technology: Ubiquitous computing, ubiquitous communication and intelligent adaptive interfaces [1]. While the two first paradigms concern the infrastructure of computing, actuator and sensing devices of the environment and their interconnection, the third one forms the interface between humans and the environment.

We aim at making this communication as effective and intuitive as possible. One approach is to simulate human-human-communication and extend classical desktop applications with modalities known from human communication like speech, gesture or facial expressions [2] [3]. New technologies and devices, available for the mainstream market, allow to introduce new interaction concepts, e.g., hand gesture control in the car [4].

Systems that incorporate and mix a variety of input and output devices and modalities are called multimodal dialogue systems [5] [6]. Multimodal dialogue systems make communication flexible and allow the system to adapt to restrictions given by the actual context. These can be caused by environmental circumstances, available technology, the users' physical and cognitive condition or their preferences. These circumstances are discussed in section II.

However, the increasing number of supported communication modalities and context-dependent adaptations have an influence on the effort that is necessary to develop, adapt, extent and maintain multimodal dialogue applications. In section III we introduce our new multimodal dialogue platform SiAM-dp (Situation-Adaptive Multimodal Dialogue Platform) that was developed with a special focus on flexibility in device/modality setup and on development concepts and tools for the rapid creation of multimodal dialogue applications in intelligent environments. An important feature is that all information is represented with semantic models (section IV). Besides the data that is processed by the application this includes the representation of the communicative function of an interaction act between user and system (as far as it can be correctly interpreted by the system). We show that we can use these models in order to define the application logic independently from the actual physical realization of communication with the user in the user interface. Models for GUI representation and speech interaction that support semantic data binding serve as an example to demonstrate how the connection between the semantic data and presentation concepts can be realized in a declarative and elegant way (section V). In section VI we explain how adaptation is supported in our architecture.

II. Context & Adaptation

Although the topic context-awareness and adaptation has been in focus of research for more than ten years, with the actual development resulting in a rising number of smaller, more ambient and better interconnected devices and sensors, it gains a new importance. A greater availability and selection set of devices, embedded in the environment, allows more room for multimodal interaction alternatives. Dey et. al [7] define context as follows: *Context is any information that can be used to characterize the situation of any entity. An entity is a person, place, or object that is considered relevant to the* *interaction between a user and an application, including the user and applications themselves.*

We recognize that *context* is a very far-reaching expression. In intelligent environments you can roughly classify context in environment- and user-related context. The environment-related context describes external conditions. This can be the infrastructure of the environment, the available interaction devices but also other factors like the actual time, persons nearby and other situational restrictions. In an intelligent home the selection of interaction devices is highly dependent on the devices actually available in a room. This completely distinguishes from the devices that are available in a car, which are optimized for offering functionality without distracting the driver from his primary task, that is to control the car. Speech based dialogue systems can simplify the communication with the system but are certainly not appropriate in the library, during a meeting, in loud environments or when processing sensitive information that can be overheard by persons nearby. Other scenarios like an operating room or a kitchen need hands-free controls in order to follow hygienic requirements.

The personalization of a system makes multimodal dialogues more user friendly and effective. Hence, user-related context and user models play a relevant role for adapting the performance of the dialogue system of a smart environment. In the simple case the system adapts to the user's personal preferences. That can be the kind of provided modalities and interfaces but also strategies for content presentation, e.g., the GUI layout, information density on a screen or the language. Speech based systems can be adapted by changing the acoustic, syntactic and speaking style, or the vocabulary.

It even makes sense to adapt the interaction strategy of the dialogue manager, e.g., an experienced user prefers a mixed-initiative system whereas an inexperienced one is more comfortable with a system-directed initiative. We demonstrate different task solving strategies at the example of a cinema seat reservation task. In order to successfully reserve a seat, the reservation system needs some relevant information like the movie name, and the day and time of the screening. The strategies can differ in the amount of information the system collects in a single dialogue turn. One approach is to collect all information at once: A GUI modality would provide a single screen with input elements for all values required ; to use speech dialogue, the system would allow more complex and content-rich utterances. A different approach is to collect the needed information step by step by asking the user in a question-answer-based speech dialogue or by providing multiple GUI windows with lower information density.

In ambient assisted living scenarios systems are designed to support the elderly or people with cognitive or physical impairments [8] [9]. Here the user-context adaptation is necessary in overcoming the restrictions caused by personal limitations of the user. For example it is necessary to adapt the color style of graphical user interfaces for color-blind people, choosing simple and clearly legible screen designs for the elderly, providing speech interaction for blind people or supporting other modalities like eye-tracker, gesture recognition or tongue control [10].

When building user interfaces for the automotive domain it is especially important to design interaction concepts that keep the distraction of the driver low. Research just started to find strategies for predicting the actual cognitive load of the driver [11] [12] [13]. Based on these findings it will be possible to adapt the dialogue and presentation strategy of a user interface and provide simpler interfaces in situations that require high attention, e.g., during the rush-hour. Also in other domains like the crisis management support [14] situation adaptive multimodal systems are used to improve the performance of the user, which can be estimated with factors like the response competition time, reaction time and numbers of errors.

III. Siam Dialogue Platform

The expertise of many years of research in multimodal dialogue systems [15] [16] [17] [18] has lead to the development of the new dialogue platform.

One of the key goals of SiAM-dp is a modular architecture, which allows modules to be added, removed, and replaced. This is particularly needed for input and output components, since it is essential for a modern multimodal dialogue platform to allow the addition of new devices and modalities. SiAM-dp is implemented in Java/OSGi that allows to start and stop modules independently and during runtime. The OSGi service architecture is utilized to enable communication between modules and process dialogue acts.

A dialogue management component is responsible for determining the implications of user interaction and trigger system reactions, causing the dialogue to progress. Currently, the dialogue manager executes as a finite state automaton, which is a very robust and transparent means for implementing dialogue systems. The automaton itself is also referred to as the runtime dialogue model, which can include executable code. A large number of dialogue models can be implemented in this way; state charts are a natural way to model many dialogues, with flow charts as an intuitive extension. Even frame-based systems can be broken down (compiled) into state machines, covering a broad range of natural language use cases.

A fusion and discourse processing module resolves ambiguities of interaction acts which are caused by different interpretations or the absence of context information when observing single interactions of a user. The main source for this are the world and dialogue context. By combining the information contained in multiple inputs from different modalities arriving in close timely manner, new interpretations can be derived and references can be resolved. Some of the dialogue phenomena that are recognized by SiAM-dp are deictic references, anaphoras, exophoric references, and spatial references.

A presentation planning and fission module determines where output should be displayed (media allocation), when (scheduling), and in what mode. Using fission, a single output can be split across devices and modalities if purposeful.

The modelling language we use in SiAM-dp is the Eclipse Modeling Framework (EMF) that is a Java framework for generating code and editing tools based on a structured data model. This language does not provide the expressive power of a full ontology language like OWL [1] but since the model is rather used for the representation of information than for automated reasoning, EMF is suitable for this task. With this approach we follow former semantic dialogue system projects like [19] that use Typed Feature Structure (TFS) with an equivalent modelling potential.

EMF is used throughout the system, e.g., for describing entities, inputs, outputs, dialogue acts, GUIs or grammar models. It is also possible to combine these models and use, e.g., entity models inside an input representation model. A key advantage of this unified data model is that dialogue systems can be created in a declarative fashion. This is achieved by a dialogue definition project, which contains a set of models for dialogue, GUI, grammar and other specifications, plus some meta information such as the expected users and devices.

A clear advantage of EMF is its integration in the Eclipse Workbench[2], an expandable rich client platform. Thus it is possible to automatically generate Java classes, that represent our models. Productivity is increased by already existing functionality for serialization and validation of model instances. Furthermore the generation of model editors in Eclipse is automated, so that it is possible to quickly generate instance editors for our models, that can, in a manual step, be customized individually.

Since interfaces to devices are defined in EMF it is possible to integrate devices either directly in the platform by using the appropriate Java classes or their serialized XML-form for communication with external devices via a network (e.g., TCP or REST).

IV. Shifting interaction on a semantic level

The nature of a multimodal dialogue system means that we have to integrate a heterogeneous set of technologies, information and interfaces. In order to allow the dialogue manager to handle input from and provide output to all of these different modalities, it is necessary to agree upon a common modeling language and especially on the models themselves, that describe the content of an interaction. The interface between input/output modules and the dialogue platform should be open and flexible enough to integrate all the different modalities.

On a syntactic level the model includes annotations for common meta data for interaction acts, such as the begin time, duration of the action, device description, modality, initiator and addressee. The model is designed to support modality-specific content descriptions. This means that particularly modalities, such as speech input, contain predefined structures for representing information that applies only to this modality, such as recognition confidence, word lattices, alternatives, or phonetic transcriptions. Similar concepts are already defined for other common modalities but the model is free to be extended, if devices with new modalities are connected. Figure 1 exemplarily shows the syntactic representation of a speech input event. Figure 2 shows a speech synthesis output request.

The dialogue management is realized with a state machine and subscribes to input events from all connected devices. Input events are used to trigger state changes in the dialogue engine. If the content of an input event matches a pattern, that is specified for an outgoing transition of the actual state, this transition is fired. Figure 3 shows a pattern, that matches the previous speech input event example. The state machine also allows to directly send output events to the framework, which are then routed to the appropriate device.

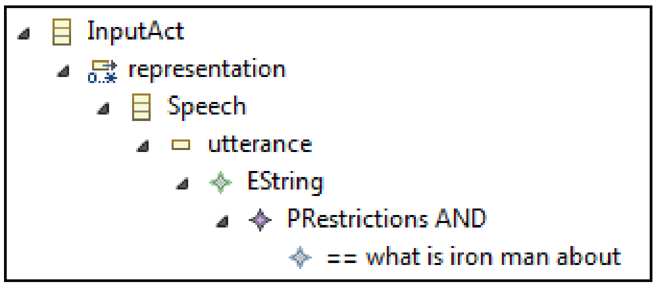

InputAct
representation
Speech
utterance
EString
PRestrictions AND
== what is iron man about

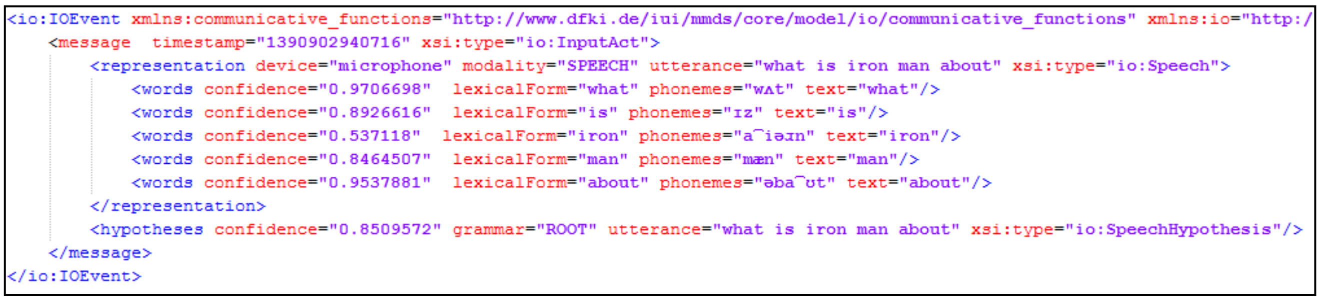
Fig. 3. Pattern that matches the syntactic representation of the speech input act: "What is Iron Man about".

A disadvantage of the representation on a purely syntactic level is that the input and output messages are very device specific. This means the model in the dialogue manager, that describes the interaction logic, must be adapted for every device and every modality that is supported by a dialogue application. In practice the application developer has to define one pattern for every input modality or one output event for every output modality, respectively.

In order to make the dialogue platform more flexible and adaptable to new modalities and devices we introduce a semantic model that describes the user's or system's intentions. Instances of this model contain the communicative function of an interaction act and the semantic content without defining how to realize it in the user interface. Thus, the model for controlling an application (i.e. the dialogue model) is completely independent from the actually used modalities and presentation strategies.

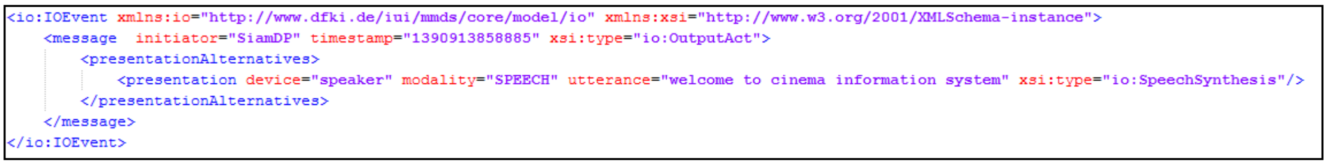
The model for defining communicative functions is inspired by a standard for the semantic annotation of dialogue acts (ISO/DIS 24617) [20]. This is an international standard for the annotation of dialogues with semantic information, in particular concerning the communicative functions of the utterances, the kind of content they address, and the dependency relations to what was said and done earlier in the dialogue. In our model we adopt the type hierarchy for communicative acts that is specified by this standard (figure 4 shows an excerpt). This can inter alia be a question, an information introduced into the dialogue, a turn taking or a task request. Originally the standard has been defined for annotating speech acts. We assume we can adopt this concept and annotate every modality that is applied in a multimodal dialogue application [21]

[1]http://www.w3.org/TR/owl2-syntax/
[2]http://www.eclipse.org/eclipse/



```xml
<io:IOEvent xmlns:communicative_functions="http://www.dfki.de/iui/mmds/core/model/io/communicative_functions" xmlns:io="http:/
    <message  timestamp="1390902940716" xsi:type="io:InputAct">
        <representation device="microphone" modality="SPEECH" utterance="what is iron man about" xsi:type="io:Speech">
            <words confidence="0.9706698"  lexicalForm="what" phonemes="wʌt" text="what"/>
            <words confidence="0.8926616"  lexicalForm="is" phonemes="ɪz" text="is"/>
            <words confidence="0.537118"  lexicalForm="iron" phonemes="a͡iəɹn" text="iron"/>
            <words confidence="0.8464507"  lexicalForm="man" phonemes="mæn" text="man"/>
            <words confidence="0.9537881"  lexicalForm="about" phonemes="əba͡ʊt" text="about"/>
        </representation>
        <hypotheses confidence="0.8509572" grammar="ROOT" utterance="what is iron man about" xsi:type="io:SpeechHypothesis"/>
    </message>
</io:IOEvent>
```

Fig. 1. XML representation of the speech recognition result: “What is Iron Man about”



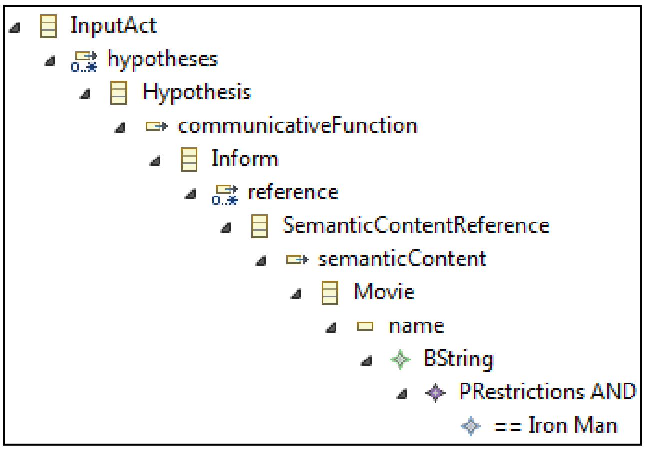
```xml
<io:IOEvent xmlns:io="http://www.dfki.de/iui/mmds/core/model/io" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <message  initiator="SiamDP" timestamp="1390913858885" xsi:type="io:OutputAct">
        <presentationAlternatives>
            <presentation device="speaker" modality="SPEECH" utterance="welcome to cinema information system" xsi:type="io:SpeechSynthesis"/>
        </presentationAlternatives>
    </message>
</io:IOEvent>
```

Fig. 2. XML representation of the speech synthesis output request: “Welcome to the cinema information system.”

[22]. Thus all modalities can be interpreted and semantically represented. For instance, a deictic gesture on an object can introduce this object into a dialogue or a head nod signals a positive feedback.

The following example (figure 5) demonstrates, how the semantic shift encapsulates the interaction logic from the actually used device. The movie ’Iron Man’ is introduced into the discourse by two different modalities. First by speech command (a), second by the click on a GUI element that shows a picture of the movie (b). Although the information provided by the input events is very device specific and different on a syntactical level, on a semantic level the communicative function and semantic content are equal. The pattern (figure 6) that is used in dialogue management to define the firing condition for a state machine transition verifies the input event only on the semantic level. Thus this transition would fire independently from the device actually used to introduce the movie into discourse. Furthermore it would also react on input events from new modalities that are interpreted with the same meaning, for instance an eye gaze at a movie poster or a deictic gesture that points to the poster.

The shifting task is performed by device interpreters that act as interface between the device and the core dialogue system. On the output side device specific generator components perform the step from a semantic representation to a syntactic presentation. These generators express the user’s intention, that is given by a semantic representation, with the possibilities given by the addressed device. This could be a natural language expression synthesized to speech or a GUI. The latter indicates that a huge amount of alternatives is possible, presentation can be varied in the type of displayed user interface elements or their layout. Using the example of the communicative act *Choice Question* we show how generators for different modalities generate output. A choice question contains a set of options the user can choose from.



```
InputAct
    hypotheses
        Hypothesis
            communicativeFunction
                Inform
                    reference
                        SemanticContentReference
                            semanticContent
                                Movie
                                    name
                                        BString
                                            PRestrictions AND
                                                == Iron Man
```

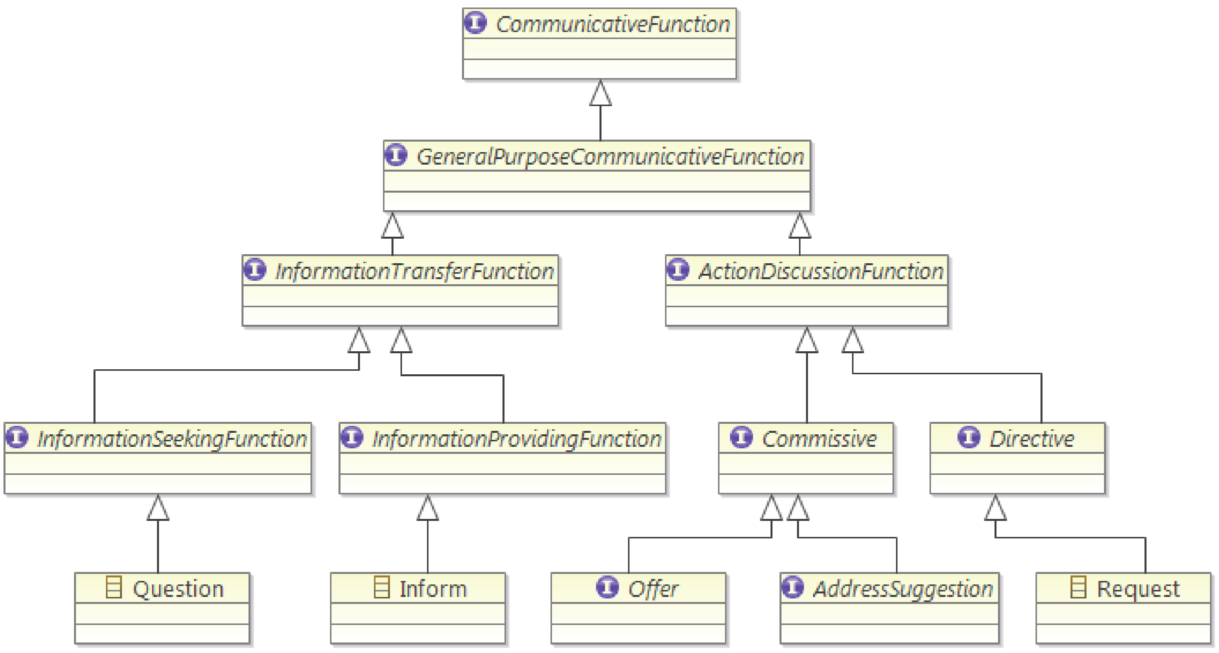
Fig. 6. Pattern that matches the device independent semantic representation of an input act that introduces the movie “Iron Man” into the dialogue.

A speech based system would realize this by enumerating the possible options via speech and extending the grammar for speech recognition with the selectable options. In a graphical user interface the choice question can be expressed by a select dropdown list, a collection of check boxes or a combo box.

V. Support of Development Process by Semantic Data Binding

In our development platform we have already provided models that specify the user interfaces of common modalities. Currently there exist models for the specification of GUIs and grammar rules for speech recognition. Furthermore models for the integration of other modalities, e.g. gestures, eye-gazing and virtual characters, are in preparation. A universal format that describes information by key-value pairs allows to communicate with modalities that are actually not considered.

Data binding is a software design pattern that simplifies the development of user interface applications by binding user



CommunicativeFunction
GeneralPurposeCommunicativeFunction
InformationTransferFunction
ActionDiscussionFunction
InformationSeekingFunction
InformationProvidingFunction
Commissive
Directive
Question
Inform
Offer
AddressSuggestion
Request

Fig. 4. Excerpt of the type hierarchy for communicative acts as defined in ISO/DIS 24617

interface components to entities of the application domain. Our models allow to bind syntactic UI specifications to both semantic entities and the semantic description of communicative functions.

A. GUI Model

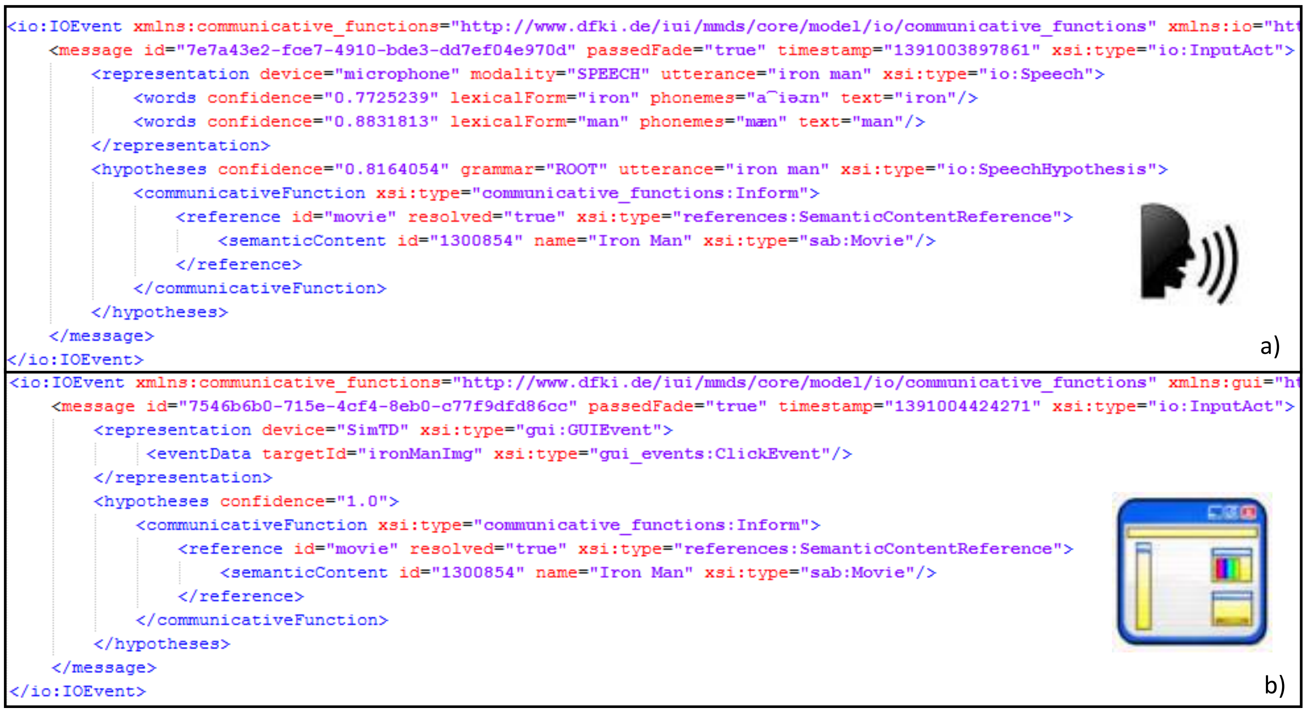
We follow a declarative GUI design approach, inspired by HTML and other application markup languages like XAML by Microsoft or the declarative GUI design for Android apps. The GUI model of our dialogue platform specifies the elements, that should be presented on the screen. So the GUI model defines the appearance of the application. The content itself is delivered by the underlying data that is bound to GUI elements. In a further step we also use data binding to enrich user interface models with information about the communicative function of the interaction of a user with it.

In figure 7 we give the example of a semantically annotated GUI model. The model instance describes a window that contains one semantic data element and two GUI elements. The semantic content (a) is the movie instance *Iron Man 3*. The instance originally contains more data but for simplification the figure only shows a list of cinema instances in which the movie is playing. The first GUI element is a label (b) that should display the movie name. The content of the label is bound to the semantic content by an adapter. The adapter defines a pattern describing the type of the semantic content that should be bound to the label. The adapter mapping describes how this content should fill the label's features. In this example the name of the movie is used to fill the text feature. The second GUI element is a list (c) that presents the cinemas in which the movie is running. The content of the list is again bound to the semantic content, in this case by an array adapter. The array adapter behaves different than the normal adapter, because it creates one GUI element for every item in a list of semantic content entities. Thus, for every cinema in which the movie is running, a list item is created with the name of the cinema. In *SupportedEvent* we define which types of interaction are supported by the list. In the example (d) this is a change event that fires whenever the user selects a new item in the list. The model allows to bind the semantic interpretation of the user's intention to this event. Here the semantic entity of the cinema is introduced into the dialogue with the communicative function *Inform*.

An internal dialogue platform component is responsible for resolving the data binding information and complements the GUI model with the necessary information that is retrieved from the bound semantic entity. Thus the displayed information changes if a new semantic entity (in this case a different movie) is attached. Furthermore the component is the interpreter for GUI input events that initially are represented syntactically, since the GUI client only provides information about the type of the event and the item on which it was performed. With the communicative function defined in the GUI model the interpreter supplements the semantically represented user intention.

B. Grammar Rule Model

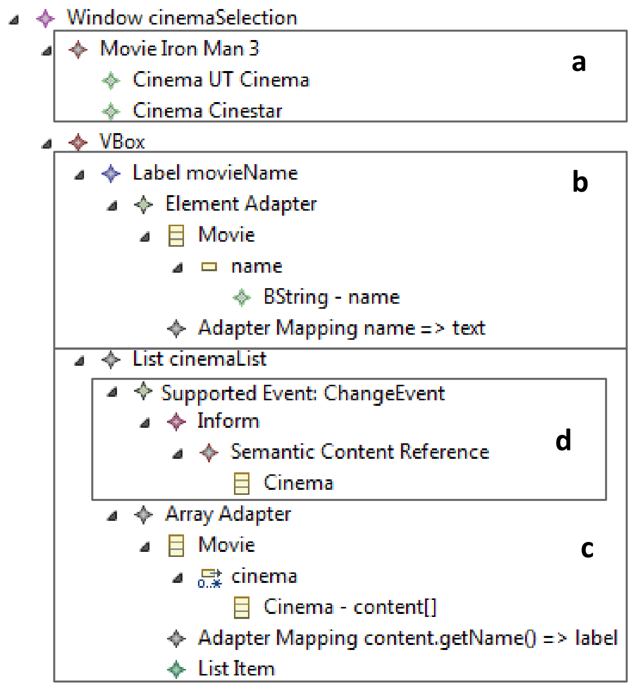
Our grammar model generally supports two types of rules. The first are entity rules. They define named entities which can be part of a speech utterance. Named entities are information units like names, including person, organization and location names, and numeric expressions including time, date, money and percent expressions [23]. Second are utterances, i.e. spoken input from the user. An utterance may be a single



```
<io:IOEvent xmlns:communicative_functions="http://www.dfki.de/iui/mmds/core/model/io/communicative_functions" xmlns:io="ht
    <message id="7e7a43e2-fce7-4910-bde3-dd7ef04e970d" passedFade="true" timestamp="1391003897861" xsi:type="io:InputAct">
        <representation device="microphone" modality="SPEECH" utterance="iron man" xsi:type="io:Speech">
            <words confidence="0.7725239" lexicalForm="iron" phonemes="a͡iəɹn" text="iron"/>
            <words confidence="0.8831813" lexicalForm="man" phonemes="mæn" text="man"/>
        </representation>
        <hypotheses confidence="0.8164054" grammar="ROOT" utterance="iron man" xsi:type="io:SpeechHypothesis">
            <communicativeFunction xsi:type="communicative_functions:Inform">
                <reference id="movie" resolved="true" xsi:type="references:SemanticContentReference">
                    <semanticContent id="1300854" name="Iron Man" xsi:type="sab:Movie"/>
                </reference>
            </communicativeFunction>
        </hypotheses>
    </message>
</io:IOEvent>
```

```
<io:IOEvent xmlns:communicative_functions="http://www.dfki.de/iui/mmds/core/model/io/communicative_functions" xmlns:gui="ht
    <message id="7546b6b0-715e-4cf4-8eb0-c77f9dfd86cc" passedFade="true" timestamp="1391004424271" xsi:type="io:InputAct">
        <representation device="SimTD" xsi:type="gui:GUIEvent">
            <eventData targetId="ironManImg" xsi:type="gui_events:ClickEvent"/>
        </representation>
        <hypotheses confidence="1.0">
            <communicativeFunction xsi:type="communicative_functions:Inform">
                <reference id="movie" resolved="true" xsi:type="references:SemanticContentReference">
                    <semanticContent id="1300854" name="Iron Man" xsi:type="sab:Movie"/>
                </reference>
            </communicativeFunction>
        </hypotheses>
    </message>
</io:IOEvent>
```

Fig. 5. XML representation of (a) the speech recognition result: “Iron Man” and (b) the click event on an image, that shows a picture of the movie. Both representations are enriched with a semantic interpretation of the input act that is the same for both modalities.



```
Window cinemaSelection
    Movie Iron Man 3
        Cinema UT Cinema
        Cinema Cinestar
    VBox
        Label movieName
            Element Adapter
                Movie
                    name
                        BString - name
                Adapter Mapping name => text
        List cinemaList
            Supported Event: ChangeEvent
                Inform
                    Semantic Content Reference
                        Cinema
            Array Adapter
                Movie
                    cinema
                        Cinema - content[]
                Adapter Mapping content.getName() => label
            List Item
```

Fig. 7. Example of a GUI model with (a) semantic content, a label for the movie title (b) and a list (c) that displays the cinemas in which the movie is running. Supported user interactions are semantically enhanced (d).

word, an entire phrase, a sentence or even several sentences. The grammars are written in Augmented-Backus-Naur Form (ABNF) that is part of the SRGS [3] standard.

The model again allows to annotate rules with semantic content. Typically named entities are connected to semantic entities. Utterances usually define complete dialogue acts and are semantically represented by communicative functions. Utterances can contain references to named entity rules that additionally provide semantic entities being content of the dialogue act.

A grammar management component is integrated in the core platform. The component has two tasks: First, to generate valid GRXML from our grammar model that is supported by the majority of actual speech recognition engines. Second, to interpret the result of a speech recognizer and to shift its content on a semantic level. For this, the GRXML description is annotated with special tags that allow the interpreter to build the semantic representations defined in the grammar rule model.

VI. Adaptation Strategies

In section II we discussed the usefulness of presentation and dialogue adaptation that allows to vary the functionality and appearance of a user interface for certain users and situations. We follow two concepts for realizing adaptation: adaptation rules and adaptation strategies. They are non-exclusive; in fact, adaptation rules are strategies without the information the system can use to calculate their effect. In the first approach the application developer declares rules that specify the adaptations for an application. These rules describe

[3]http://www.w3.org/TR/speech-grammar/

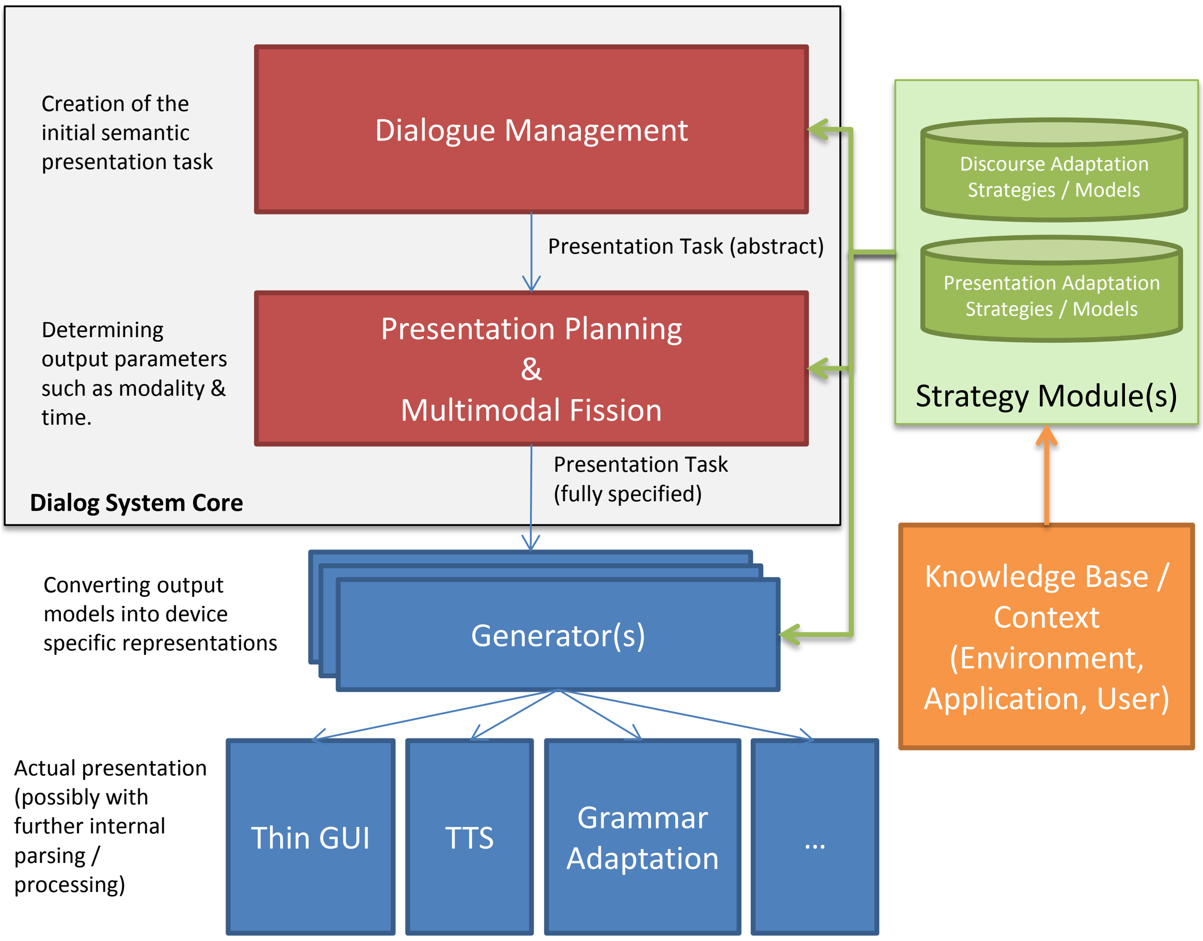

Creation of the initial semantic presentation task
Dialogue Management
Presentation Task (abstract)
Determining output parameters such as modality & time.
Presentation Planning & Multimodal Fission
Presentation Task (fully specified)
Dialog System Core
Discourse Adaptation Strategies / Models
Presentation Adaptation Strategies / Models
Strategy Module(s)
Converting output models into device specific representations
Generator(s)
Knowledge Base / Context (Environment, Application, User)
Actual presentation (possibly with further internal parsing / processing)
Thin GUI
TTS
Grammar Adaptation
...

Fig. 8. Architecture concept for adaptation of dialogue behaviour and presentation.

under which context conditions the presentation and dialogue behaviour should be adapted. Thus, the dialogue application is adapted to certain situations, users or user groups by the developer. This behaviour is defined with semantic models we introduced in section IV. Alternatively strategies calculate the effect of a context condition on the system. These modules follow a more generic approach and provide application independent functionality. Here the adaptation is considered as an optimization problem with a goal function, optional additional constraints and conflict resolution. The system for example can decide to switch the modality from a speech based to GUI based conversation, if the environment is too noisy or too many speech recognition problems occur. For the automotive domain we are planning to develop algorithms and heuristics that evaluate dialogue and presentation models in terms of their distraction to the driver. If the system recognises (e.g. via biometric sensors) that the driver is in high cognitive load demanding situation, the strategy switches to a model with a lower impact on this.

Adaptation strategies have to be distinguished by the initiator. This is the component that a) determines when a (possible) adaptation occurs, b) lists all available strategies, and c) implements the result of the adaptation strategy, if it was selected. Possible initiators in our dialogue system are:

- Dialogue Management: These strategies affect state changes in the dialogue discourse, e.g., what happens when we press the "send" button or whether the system will confront the user when he drives too fast. It can also filter information that is presented to the user.
- Presentation Planning and Fission: These strategies affect the parameters determined by modules concerning the presentation, e.g., the presentation time and modality. Information that is presented to the user can be distributed across several modalities. Furthermore they control how strategies apply to presentation models, e.g., how information of a communicative act is mapped to graphical items (or possibly ignored). The initiator would be the generator that transforms a semantic model of communicative acts and data into a GUI description.
- Representation Specific Generator: Strategies particularly affecting the generation of the representation, e.g., the abstract GUI model or TTS string, are initiated by a generator dedicated to this purpose. The implementation has to conform to a special (e.g., GUI) adaptation language that contains, e.g., styles to be adjusted (e.g., text size, layout change, hiding items, limiting list size etc.).

Figure 8 displays our architecture concept for the adaptation. The modules for adaptation (strategy modules) contain a set of strategies/rules and models, that define the application's appearance and behavior dependent from different context (user, environment and application) states. The modules evaluate context information they retrieve from the knowledge base and select the best matching models for the actual context state. These models change the behavior of the dialogue management, presentation planner, multimodal fusion and device specific generators.

VII. Conclusion

In this paper we presented a multimodal dialogue development platform that allows to rapidly build multimodal dialogue applications which are adaptable to the actual context. We show how the semantic shift of contents and communicative acts helps to define the behavior of the system independently of the actually connected devices. Device-specific representation is realized by declarative models that we exemplarily provide for the modalities speech input and graphical user interfaces. An advantage of our models is that they can directly be bound to the semantic representations of their content and interaction intentions. This model based approach supports the process of building context adaptive dialogue applications.

Our research focuses on exploiting the platform's benefits in order to develop and evaluate adaptation strategies for modern multimodal in-car user interfaces. An adaptation criteria of high value will be the actual cognitive load of the driver.

Acknowledgements

This work was funded by the German Ministry of Education and Research (project SiAM: Situation-Adaptive Multimodal Interaction for Innovative Mobility Concepts for the Future; grant number 01IW11004).

References

[1] R. López-Cózar and Z. Callejas, "Multimodal dialogue for ambient intelligence and smart environments," in *Handbook of ambient intelligence and smart environments*. Springer, 2010, pp. 559–579.

[2] B. Dumas, D. Lalanne, and S. Oviatt, "Multimodal interfaces: A survey of principles, models and frameworks," in *Human Machine Interaction: Research Results of the MMI Program*, 2009, pp. 3–26.

[3] M. Turk and M. Kölsch, "Perceptual interfaces," University of California, Santa Barbara, Tech. Rep., 2003.

[4] A. Riener, A. Ferscha, F. Bachmair, P. Hagmüller, A. Lemme, D. Muttenthaler, D. Pühringer, H. Rogner, A. Tappe, and F. Weger, "Standardization of the in-car gesture interaction space." in *AutomotiveUI*, J. M. B. Terken, Ed. ACM, 2013, pp. 14–21.

[5] W. Wahlster, Ed., *SmartKom: Foundations of Multimodal Dialogue Systems*. Berlin: Springer, 2006.

[6] M. Turk, "Multimodal interaction: A review." *Pattern Recognition Letters*, vol. 36, pp. 189–195, 2014.

[7] A. K. Dey and G. D. Abowd, "Towards a better understanding of context and context-awareness," in *Workshop on The What, Who, Where, When, and How of Context-Awareness*, ser. 2000 Conference on Human Factors in Computing Systems, 2000.

[8] J. Frey, C. H. Schulz, R. Neßelrath, V. Stein, and J. Alexandersson, "Towards pluggable user interfaces for people with cognitive disabilities," in *Proceedings of the 3rd International Conference on Health Informatics. HEALTHINF-2010, in Conjunction with Proceedings of the 3rd International Conference on Health Informatics, January 20-23, Valencia, Spain*. Springer, 1 2010.

[9] C. Jian, H. Shi, F. Schafmeister, C. Rachuy, N. Sasse, H. Schmidt, V. Hoemberg, and N. Steinbüchel, "Touch and speech: Multimodal interaction for elderly persons," in *Biomedical Engineering Systems and Technologies*, ser. Communications in Computer and Information Science, J. Gabriel, J. Schier, S. Huffel, E. Conchon, C. Correia, A. Fred, and H. Gamboa, Eds. Springer Berlin Heidelberg, 2013, vol. 357, pp. 385–400.

[10] J. Struijk, "An inductive tongue computer interface for control of computers and assistive devices," *Biomedical Engineering, IEEE Transactions on*, vol. 53, no. 12, pp. 2594–2597, Dec. 2006.

[11] W. Piechulla, C. Mayser, H. Gehrke, and W. König, "Reducing drivers' mental workload by means of an adaptive man-machine interface," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 6, no. 4, pp. 233 – 248, 2003.

[12] R. Nesselrath and M. Feld, "Towards a cognitive load ready multimodal dialogue system for in-vehicle human-machine interaction," in *Adjunct Proceedings of the 5th International Conference on Automotive User Interfaces and Interactive Vehicular Applications, Eindhoven*, Oct. 2013, pp. 49–52.

[13] M. Schwalm, A. Keinath, and H. D. Zimmer, "Pupillometry as a method for measuring mental workload within a simulated driving task," *Human Factors for assistance and automation*, pp. 1–13, 2008.

[14] A. Niculescu, Y. Cao, and A. Nijholt, "Manipulating stress and cognitive load in conversational interactions with a multimodal system for crisis management support," in *Development of Multimodal Interfaces: Active Listening and Synchrony*, ser. Lecture Notes in Computer Science 5967, A. Esposito, N. Campbell, C. Vogel, A. Hussain, and A. Nijholt, Eds., Heidelberg, 2010, pp. 134–147.

[15] J. Schehl, A. Pfalzgraf, N. Pfleger, and J. Steigner, "The BabbleTunes system—talk to your iPod!" in *Proceedings of the 10th International Conference on Multimodal Interfaces (ICMI '08), Chania, Crete, Greece*, 2008, pp. 77–80.

[16] D. Porta, D. Sonntag, and R. Neßelrath, "A multimodal mobile b2b dialogue interface on the iphone," in *Proceedings of the 4th Workshop on Speech in Mobile and Pervasive Environments in conjunction with MobileHCI '09. SiMPE-09, September 15, Bonn, Germany*, 2009, multimodal Interaction, Mobile Business Services, Usability, User Experience, Productivity.

[17] R. Neßelrath and D. Porta, "Rapid development of multimodal dialogue applications with semantic models," in *Proceedings of the 7th IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems (KRPD-11). Twenty-Second International Joint Conference On Artificial Intelligence (IJCAI -11)*, Barcelona, Spain, July 2011.

[18] S. Bergweiler, M. Deru, and D. Porta, "Integrating a multitouch kiosk system with mobile devices and multimodal interaction," in *ACM International Conference on Interactive Tabletops and Surfaces*, ser. ITS '10. New York, NY, USA: ACM, 2010, pp. 245–246.

[19] I. Gurevych, R. Porzel, E. Slinko, N. Pfleger, J. Alexandersson, and S. Merten, "Less is more: Using a single knowledge representation in dialogue systems," in *Proceedings of the HLT-NAACL 2003 Workshop on Text Meaning*, 2003, pp. 14–21.

[20] H. Bunt, J. Alexandersson, J. Carletta, J.-W. Choe, A. C. Fang, K. Hasida, K. Lee, V. Petukhova, A. Popescu-Belis, L. Romary, C. Soria, and D. R. Traum, "Towards an iso standard for dialogue act annotation." in *LREC*, N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner, and D. Tapias, Eds. European Language Resources Association, 2010.

[21] V. Petukhova and H. Bunt, "The coding and annotation of multimodal dialogue acts," in *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, N. C. C. Chair), K. Choukri, T. Declerck, M. U. Do?an, B. Maegaard, J. Mariani, J. Odijk, and S. Piperidis, Eds. Istanbul, Turkey: European Language Resources Association (ELRA), May 2012.

[22] H. H. Vilhjálmsson, "Representing communicative function and behavior in multimodal communication," in *Multimodal Signals: Cognitive and Algorithmic Issues*, ser. Lecture Notes in Computer Science, A. Esposito, A. Hussain, M. Marinaro, and R. Martone, Eds. Springer Berlin Heidelberg, 2009, vol. 5398, pp. 47–59.

[23] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Linguisticae Investigationes*, vol. 30, no. 1, pp. 3–26, January 2007, publisher: John Benjamins Publishing Company.