SERVING COMPLEX USER WISHES WITH AN ENHANCED SPOKEN DIALOGUE SYSTEM

Sunna Torge, Stefan Rapp, Ralf Kompe

Sony International (Europe) GmbH
Advanced Technology Center Stuttgart, Germany
E-mail: {torge, rapps, kompe}@sony.de

ABSTRACT

In the past years spoken dialogue systems became more and more sophisticated, i.e., they allow for rather complex and flexible dialogues. On the other hand the functionality of devices, applications and services (henceforth applications) and the amount of digital content increased rapidly. Due to the network age (Internet, personal area networks,...) the applications we can/want to control change dynamically as well as the content. Therefore a dialogue system cannot be anymore manually designed for one application before product release. Instead a layer in between is required, which translates the device functionalities in a way that they can be used via a more or less generic dialogue system. Furthermore, a network of devices creates new functionalities not possible with a single device. Users will have complex wishes; often several devices will be necessary to solve such a wish. However, the user in most cases would not like to care about how many and exactly which applications are used to fulfill his wish. From his perspective he sees the functionality provided by a network of devices as a virtual device. In this paper a module is described which builds a link between dialogue system and an ensemble of applications. Functionalities of applications are described in a formal way. A planning component searches for the applications necessary to solve a complex user wish and the sequence of action which has to be performed. Simplicity of the approach is very important. the module has been successfully integrated in a prototype system which was publicly demonstrated.

1. INTRODUCTION

Although human-machine interactions are ubiquitous, they are not always a very pleasing experience. An often cited example is how inconvenient and error-prone it is to program a video cassette recorder (VCR). Another example is that currently users have a lot of troubles, when they want to connect bluetooth devices. The main problem with most of the user interfaces is that the user has to think in terms of devices and services (play, record, ...). Instead, the user should be able to interact with a system in a natural way, as he would do with a human assistant. To give an example, the user might put a document on the table and simply say "send this to Mr. Green". In consequence, the system should deal with the devices and services for him. It should scan the paper with a document camera, find out how to reach Mr. Green by consulting an appropriate address book, transmit the image either to an e-mail client or a fax modem, possibly doing format conversion before etc. The user in this case has a complex wish involving several devices and several actions per device. He wants simply his wish to be solved; he does not want to care about the individual devices or the action sequences to be carried out. Similar problems arise in the case of copying videos or combining EPG access and personal calendar. Note, that these tasks can only be solved by a system providing a central human-machine interface operating several devices.

Given a home network or personal area network, the problem we address in this paper is to enable a flexible and intuitive control of an ensemble of devices and services with a single intuitive human-machine interface. We focus on

- how complex user wishes can be served, and
- how plug&play can be realized on the side of the dialogue system.

In order to solve these problems different aspects need to be considered:

A traditional dialogue system used for the control of devices usually consists of an input understanding part, a dialogue manager, and the devices, which are to be controlled. The simplest way to control the devices is to have a unique mapping of the user input to the appropriate control command. Given e.g. a speech input "CD play" it uniquely can be mapped to the "play"-*command* of a CD player [1]. However, this approach does not allow to serve complex wishes like the above mentioned examples, since the user input for complex wishes cannot be uniquely mapped to a single control command.

In order to gain more flexibility w.r.t. adding new applications it is necessary to separate domain-dependent knowledge from domain-independent knowledge [2, 3, 4, 5].

Separation of domain-dependent and -independent knowledge can be obtained by introducing an additional module serving as an interface between the dialogue manager and the devices to be controlled, including models of the devices [6]. However, to serve complex user wishes it is not enough to consider models of single devices. Instead, some sort of reasoning on the provided services is necessary.

In [7] the use of pre-calculated plans for complex tasks is described. With this approach however, the user is limited to these tasks, where pre-calculated plans are available.

Concerning plug&play, there are existing physical solutions like Bluetooth and IEEE1394 (iLINK), which offer plug&play on a hardware level. This has also to be reflected in the human-machine interface since new devices have to be controlled through the human-machine interface.

In order to overcome these sort of problems a spoken dialogue system is enhanced with a planning component. This planning component consists of a reasoning component and a set of abstract models describing the functionalities of devices and services available in the network. Using these abstract models the human-machine interface can infer how to control the available services and devices.

2. THE PLANNING MODULE

As described above our focus is on serving complex user requests which may involve several devices and services. In order to handle requests like “send this to Mr. Green” or ”Please, record the film XYZ on Saturday” the system needs to find out

- *which* devices are necessary to serve the request,
- *how* to control the devices.

For this purpose a planning component and appropriate data structures for planning are needed. In a plug&play environment this can not be pre-programmed.

Therefore a new module, called planning module, is introduced and integrated to a spoken dialogue system. Its purpose is to allow the integration of the functionalities of several different devices.

In a traditional dialogue system the devices which are to be controlled by the system, are controlled by the dialogue manager directly. However, in order to keep the domain dependent knowledge out of the dialogue manager the enhanced system (see figure 1) provides a sort of intelligent interface, namely the planning module, between the dialogue manager and the devices to be controlled. The planning module consists of

- an abstract model of the functionalities of a device (so called functional model) for each device in the network,
- a reasoning component,
- the administration of the current state of each device.

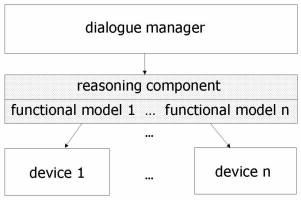

Fig. 1. Dialogue system with planning module consisting of a reasoning component and functional models

Instead of controlling the devices directly from the dialogue manager, this allows to formulate the requests given by the dialogue manager on an abstract level. Based on this abstract request, the planning module first calculates a plan to serve the request and then performs the plan.

The dialogue manager is therefore independent of the real devices, and robust against changes of them. The overall functionality of the given devices does not need to be known in the dialogue manager but it is deduced from the functional models of the given devices.

In the following, the different parts of the planning module, i.e. the functional model and the reasoning component are described in detail.

2.1. The Functional Model

The functional model is a descriptive model rather than a procedural one. It is separated into an external model and an internal model. This separation reduces the search space, since the external model only describes the knowledge necessary to find appropriate devices whereas the internal model describes the single device in more detail. From a practical point of view it was necessary to keep the model description simple.

2.1.1. External Model

The external model of a device describes the input data and the output data. As a very simple example let us consider a database, that allows the extraction of certain information by appropriate queries. From the viewpoint of a dialogue system a database is a device where a request could be sent and a result will be returned (see figure 2). This knowledge is necessary but also sufficient to decide whether the device “database” is appropriate to serve a given user request or not. The external model also allows to draw conclusions on which devices are to be combined. As a simple example let us consider the request “I’d like to see channel 4”. Assume the given network consists of a tuner and a display. The knowledge given by the external models of the tuner and the display is sufficient to infer that these two devices need to be combined in order to serve the request.

Note, that a single functional model may consist of several external models, since an external model corresponds to a functionality of the device. The functional model of an

video cassette recorder (VCR) consists e.g. of an external model describing "recording" and another one, describing "playback".

Fig. 2. An example of an external model

2.1.2. Internal Model

The internal model of a device describes the possible states of a device and the actions which are necessary to bring the device from one state into another. Regarding the knowledge to be modeled, i.e. states and actions, finite state machines (FSM) are an appropriate means to built up an internal model.

The states in the internal model are partly annotated with incoming or outgoing data respectively. This is an important point since these annotations ensure the connection between external and internal model of a device. In addition the states also might be annotated with pre- or postconditions of each state. The internal model of the database is depicted in figure 3.

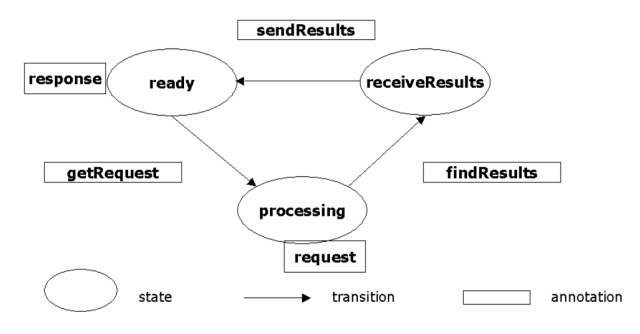

Fig. 3. An example of an internal model

2.1.3. Formal Description of the Functional Model

The functional model can be formally described as follows:

Definition 1 (Functional Model) *A* functional model $\mathcal{F}$ *is a tuple* $F = (\mathcal{D}, \mathcal{S}, \mathcal{T}, i, o, pre, post)$ *where* $\mathcal{D}$ *defines the external models and* $\mathcal{S}, \mathcal{T}, i, o, pre, post$ *define the internal model.*

- $\mathcal{D}$ *is a finite set of* external models *with*

$$\mathcal{D} = \{(d_j^i, d_j^o) \mid d_j^i \in \mathcal{D}^i,\ d_j^o \in \mathcal{D}^o\}, \mathcal{D} \subseteq \mathcal{D}^i \times \mathcal{D}^o$$

- $\mathcal{S}$ *is finite set of* states,
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$ *is a* transition function *and* $\mathcal{A}$ *a set of* actions *(including the empty action)*
- $i, o, pre, post$ *are* functions, *which informally spoken map additional information to a state* $s \in \mathcal{S}$. *They are defined as follows, where* ϵ *is the empty element:*
  $i : \mathcal{S} \rightarrow \mathcal{D}^i \cup \{\epsilon\}$
  $i : s \mapsto d^i$
  $o : \mathcal{S} \rightarrow \mathcal{D}^o \cup \{\epsilon\}$
  $o : s \mapsto d^o$

  Let $\mathbf{F} = \{\mathcal{F}_\mathbf{1}, ..., \mathcal{F}_\mathbf{n}\}$ *be a finite set of functional models with* $\mathcal{F}_i = (\mathcal{D}_i, \mathcal{S}_i, \mathcal{T}_i, i_i, o_i, pre_i, post_i)$, $i = 1, ..., n$ *and* $\mathcal{F} \in \mathbf{F}$.
  Let $\mathbf{D^i} = \bigcup_{\mathbf{j=1}}^{\mathbf{n}} \mathcal{D}_j^i$ *and* $\mathbf{D^o} = \bigcup_{\mathbf{j=1}}^{\mathbf{n}} \mathcal{D}_j^o$. *Then* pre *and* $post$ *are defined as*

  $pre : \mathcal{S} \rightarrow \mathcal{P}(\mathbf{D^i} \times \mathbf{D^o})$
  $pre : s \mapsto \{(d_j^i, d_j^o) \mid j = 1, ..., l, d_j^i \in \mathbf{D}^i, d_j^o \in \mathbf{D}^o\}$

  $post : \mathcal{S} \rightarrow \mathcal{P}(\mathbf{D^i} \times \mathbf{D^o})$
  $post : s \mapsto \{(d_j^i, d_j^o) \mid j = 1, ..., k, d_j^i \in \mathbf{D}^i, d_j^o \in \mathbf{D}^o\}$

Note, that the definition of i, o only refers to the external models which are part of the functional model to be defined, whereas pre and $post$ refers to the external models of all functional models under consideration. Note furthermore, that the definition of pre- and postconditions pre and $post$ corresponds to the definition of the external models.

2.2. The Reasoning Component

Each of the devices of the given network is modeled by its functional model. In order to use this knowledge the planning module in addition consists of a reasoning component. This is necessary, since given a complex user request and given a network of devices, the following steps need to be executed in order to serve the user request:

1. Find out the device/devices which are necessary to perform the request.
2. Find out how to combine and to control the devices, i.e. generate an appropriate plan of actions to be executed.
3. Execute the necessary actions.

Using the functional model of each device in the given network the reasoning component first searches for the necessary devices and then generates a plan of actions for every device, which is involved. The plan finally will be executed.

The functional models allow to describe how devices need to be combined. This is done via the input and the output data of each device. Furthermore it is possible to describe a temporal order in which the devices are to be controlled. This is done by formulating pre- and postconditions of single states of a device. Let us have a closer look to the algorithms.

In a first step a complex task T is mapped to a tuple $(d_T^i, d_T^o) \in \mathbf{D^i} \times \mathbf{D^o}$ (see definition 1). Let $\bigcup_{j=1}^{n} \mathcal{D}_j$ be the set of all external models which are currently available. Then, using the representation (d_T^i, d_T^o) of the task the device search algorithm returns a set of external models $\mathcal{D}^T$ of the necessary devices

$$\mathcal{D}^T \subset \bigcup_{j=1}^{n} \mathcal{D}_j, \mathcal{D}^T = \{(d_T^i, d_1^o), ..., (d_j^i, d_T^o)\}.$$

There are two cases to distinguish:
1. $|\mathcal{D}^T| = 1$, i.e. planning starts and ends with one single device.
2. $|\mathcal{D}^T| > 1$, i.e. an ordered set of external models is returned. Each model uniquely corresponds to one device. And for each device the search algorithms are performed as will be described for case 1.

Case 1: Given $\mathcal{D}^T$ and the internal model of the device, during the next step the initial and the final state of the device is searched for. This search is based on the annotation functions i and o, which establish the connection between external models and the appropriate states of the internal model. As mentioned before the functional model also includes the knowledge of the current state of each device. Therefore at that point the system knows about the current, the initial, and the final state to be reached by the device.

Thus, the next step is to search for a path from the current to the initial to the final state in the internal model of the device. Since the transitions of the internal model of a device are annotated with actions, a sequence of actions, i.e. a plan is returned.

While searching for a path (i.e. a plan), pre- and postconditions of the states are checked. Informally spoken, the pre- and postconditions model the necessity of using further devices and how to use them. Since pre- and postconditions are formulated like the external models (see definition 1), the same algorithms are applied to fulfill them, which return additional plans. These additional plans, corresponding to other devices, which are necessary to fulfill the given complex task, finally are integrated in the overall plan.
Case 2: See above.

It should be pointed out that the concept described above allows two possibilities to model the necessity of using several devices. First, it can be modeled by mapping the complex task to $(d_T^i, d_T^o) \in \mathbf{D^i} \times \mathbf{D^o}$, such that the device search algorithm returns an ordered set of external models (and also devices) as described above. Secondly, the concept of pre- and postconditions allows to model a more flexible temporal order.

3. REALIZATION IN A PROTOTYPE

The work described in this paper is done in the SmartKom project [8]. In SmartKom a dialogue system is being developed where spontaneous speech input is combined with pointing gestures. A first prototype implementation was shown at Eurospeech 2001 and at the international MTI Conference, Saarbruecken 2001. This successfully uses the reasoning component as well as the function model of microphone, loudspeaker, tourist information database, navigation, address book, tv, vcr, biometrics, e-mail service, telephone, and document camera.

4. FUTURE WORK

Future work will include refinement of the reasoning component concerning efficency and assessment of generated plans. Assessment will be based on costs, which are given through annotation of actions. Other desirable features of the reasoning component are the generation of conditional plans and exception handling. Instead of choosing one single plan the planning module also could propose several plans to the user and leave the decision to her. In order to realize plug&play the functional model of a device needs to be integrated in the device itself or on an internet server. This is simulated right now in the SmartKom system. Another issue is to consider complexity. In [9] complexity results are shown for a multi-agent system. The models used in this work are comparable to the functional model. Thus the complexity results may also apply to the work described in this paper.

5. ACKNOWLEDGMENT

This research was conducted within the SmartKom project and partly funded by the German Federal Ministry of Education and Research under grant 01IL905I7.

6. REFERENCES

[1] S. Rapp, S. Torge, S. Goronzy, and R. Kompe. Dynamic speech interfaces. In *Proc. of Workshop AIMS at ECAI 2000*, Berlin, 2000.

[2] A. Flycht-Eriksson. A domain knowledge manager for dialogue systems. In *Proc. of ECAI 2000*, Berlin, 2000.

[3] W. Thompson and H. Bliss. A declarative framework for building compositional dialog modules. In *Proc. of ICSLP 2000*, Bejing, China.

[4] J. Han and Y. Wang. Dialogue management based on a hierarchical task structure. In *Proc. of ICSLP 2000*, Bejing, China.

[5] O. Lemon et al. The witas multi-modal dialogue system i. citeseer.nj.nec.com/lemon01witas.html.

[6] X. Pouteau and L. Arevalo. Robust spoken dialogue system for consumer products: A concrete application. In *Proc. of ICSLP 1998*, Sydney, Australia.

[7] S. Larsson, R. Cooper, and S. Ericsson. menu2dialog. In *Proc. of Workshop Knowledge and Reasoning in Practical Dialogue Systems at IJCAI 2001*, Seattle.

[8] W. Wahlster, N. Reithinger, and A. Blocher. Smartkom: Multimodal communication with a life-like character. In *Proc. of Eurospeech 2001*, Aalborg, 2001.

[9] C. Domshlak and Y. Dinitz. Multi-agent off-line coordination: Structure and complexity. In *Proceedings of 6th European Conference on Planning, 2001*.