Generating Dialogue Applications with the GEMINI Platform*

Stefan W. Hamerich, Volker Schless, Volker Schubert, Yu-Fang H. Wang
TEMIC Speech Dialog Systems, Ulm, Germany
{stefan.hamerich|volker.schless|volker.schubert|helena.wang}
@temic-sds.com

Abstract: Within the EC funded research project GEMINI (Generic Environment for Multilingual Interactive Natural Interfaces) we aim at the development of a platform that assists the user to semi-automatically produce interactive multilingual and multimodal dialogue interfaces to databases. To demonstrate the platform's efficiency two different applications were generated using this platform. Main features of the platform are its effectiveness and the adaptability of generated applications to reduce costs and to ensure reusability, the compliance with current standards, and the possibility to model many aspects of real world applications like user levels and overanswering.

1 Introduction

The project GEMINI [1] exploits experience gained from previous projects (see [EHH+97, LSG+00]) and from real-world use of similar systems, to create a generic platform for the development of user-friendly, natural, high quality, platform independent, multilingual and multi-modal interactive interfaces to a wide area of databases and back-end services employed by information service providers.
GEMINI's main idea is that, given a database structure and a rough idea of the dialogue flow, the system should be able to semi-automatically generate the necessary dialogue scripts for the service application. In a sense, the information provided to the system corresponds to what a human operator in a call center needs to know in order to perform his job. Within the project we strive to get as close as possible to this ideal.
The way of setting up new dialogue applications in GEMINI differs in main points from other approaches. Compared with the REWARD system [BBO98] the GEMINI platform allows the generation of dialogues for several modalities and it generates dialogues in standardised description languages.

This paper is structured as follows: First we describe in detail the application generation platform (AGP). Afterwards we shortly introduce the two pilot applications that were set up using the GEMINI AGP and conclude our major findings.

*This work was partly supported by the European Commission's Information Society Technologies Programme under contract no. IST-2001-32343.The authors are responsible for the contents of this publication.

[1] For details see the GEMINI homepage at www.gemini-project.org.

2 Application Generation Platform

The focus of the GEMINI project is the development of a platform for generating interactive, multilingual and multi-modal dialogue interfaces to databases with a minimum of cost and human effort. We had several objectives in mind when building the AGP: Firstly we aimed at minimising the effort for the designer to build a new dialogue application. Secondly the platform supports libraries in order to allow reusage of dialogue modules and to speed up the development process. Finally we put strong effort in standardisation and descriptive modelling. To achieve this goal, all models generated by the AGP are described in GDialogXML (GEMINI Dialog XML), which is an object-oriented abstract dialogue modelling language.[2] By using Qt, the AGP can be compiled for every platform. And finally, the output of the platform is XHTML scripts for web applications and VoiceXML scripts for speech applications.

All components of the AGP are integrated into one common GUI. In Figure 1 the architecture of the AGP is illustrated. The whole AGP consists of three layers. These layers are described in more detail in the following sections.

2.1 Framework layer

The framework layer is the first layer of the AGP (cf. Figure 1). In this layer the basic models for an application are generated, which are no parts of the dialogue but form the application framework. This layer includes the application description assistant (ADA), the data modelling assistant (DMA), and the data connector modelling assistant (DCMA). As indicated by the black arrow in the upper left corner of Figure 1, all assistants are controlled manually.

The designer has to provide the application description, which mainly consists of the modalities for which the AGP should generate dialogue scripts, the languages of the application and settings for error handling. For further information about the error handling capabilities of the AGP refer to [WHS03].

The DMA helps creating the data model, which consists of class descriptions. Each class is characterised by a list of attributes, a description, and a list of base classes (inheriting their attributes).

Finally the DCMA as the third assistant in the framework layer helps creating APIs and implementation references for application specific data access functions.[3] These functions will be used in the runtime system without any knowledge of the existing database.

[2] For a detailed description of GDialogXML refer to [HWS+03].

[3] The implementation of data access functions has to be done outside of the AGP context, since special knowledge about the database itself is needed for this. Thus the AGP is database independent.

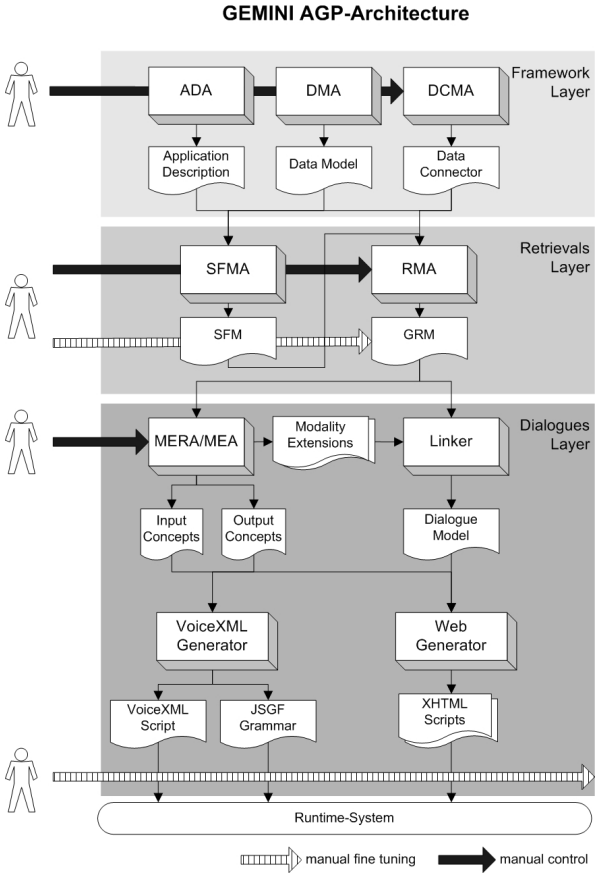

Figure 1: Schematic view of the AGP architecture.

2.2 Retrievals layer

The retrievals layer (the second layer in Figure 1) mainly consists of the state flow modelling assistant (SFMA) and the retrieval modelling assistant (RMA). This layer is modality and language independent and as its main result it produces the generic retrieval model (GRM), which models the abstract retrievals of the dialogue application.
The designer first uses the SFMA to create the abstract dialogue flow by specifying the high-level states of the dialogue, including information about what slots are asked of the user and which are the following states. To specify the slots, attributes from the data model are offered to speed up the design. It is a very high level definition of the dialogue.
The following assistant is the RMA, which provides a user-friendly interface to generate dialogue modules. The resulting output of the RMA is the GRM, which consists of the modality and language independent parts of a dialogue. The GRM, formulated in GDialogXML, models the information flow between dialogue modules. Each dialogue

module contains actions, like calls to other dialogue modules, calls to the back-end system and control structures. The GRM has the expressive power of an object oriented, strongly typed programming language, formulated however in a descriptive, hierarchical way.

2.3 Dialogues layer

In the dialogues layer the complete dialogue of an application is generated. This layer is modality and language specific as now the modality extensions from the modality extension retrieval assistant (MERA) and the language dependent extensions from the modality extension assistant (MEA) are added to the retrieval model. The modality extensions consist of special subdialogues which are specific for one modality only. The current implementation of the AGP supports the generation dialogues for speech and web modality.
In order to generate runtime scripts the GRM has to be enriched by the modality extensions using the Linker. The resulting model is called dialogue model, which is processed by the speech script generator and/or the web-page script generator depending on the selected modalities in the application description.
The VoiceXML generator reads in all models and works directly on an internal representation of the models. This is only possible since GDialogXML is very powerful and concept-oriented, and therefore needs no expensive parsing or decoding. In order to connect the runtime services (including the back-end system), a data bridge has been developed. It allows for incorporating result values from calls into the VoiceXML interpreter by dynamically producing VoiceXML code. The bridge connects the runtime services via the newly developed GEMINI service protocol on top of HTTP.
The main runtime services are (1) the data connector, building the access layer to the database, (2) the user level detector, switching between different user types, (3) the prompt generator, generating natural language prompts on the fly, (4) the speaker verification component, accepting or rejecting the speaker by performing pattern recognition on audio input, (5) the language detector, identifying the currently spoken language.
For the web modality a web-page script is generated out of the dialogue model which enables dynamic web pages. For the speech modality, some more tools are relevant, namely the language modelling tool and the vocabulary builder.

3 Applications

Two pilot applications have been set up using the AGP for evaluation and validation. Both were generated in a very user friendly way, taking into account the error handling capabilities of the AGP.

The voice banking application called EG-Banking constitutes a voice portal for user-friendly, high-quality interactions for bank-customers. The main functionality includes a general information part available to the public and a transaction part available to customers only. This application covers speaker identification, language detection, and user

modelling. EG-Banking has been developed by Greek and Spanish project partners, the Greek version of it is used for phone banking by a Greek Bank.

CitizenCare is an e-government dialogue system for citizen-to-administration interaction (via multiple channels like internet and public terminals), filled with content for an exemplary community. The main functionality is an interactive authority and information guide, providing different views like an administrative view, based on the hierarchical structure of the authorities, and a concern-oriented view, giving the citizen all the information needed to make use of services offered by public administration authorities, e.g. what to do when applying for a new identity card. This application has been developed by the German project partners for web and speech modality. The latter allows overanswering and has a mixed initiative dialogue strategy. CitizenCare is available in English and German.

4 Conclusion

During the GEMINI project we developed an application generation platform, which generates state of the art speech and web applications. The platform architecture is able to effectively set up dialogue applications from different databases. Within a three level architecture the components of the AGP support the user to set up new applications semi-automatically. Reusability of applications generated by the platform is ensured by the use of standards like VoiceXML, XHTML and a platform independent programming environment. Integration of new runtime services and additional modalities can be done easily by extending the modular architecture of the platform.

References

[BBO98] Brøndsted, T., Bai, B. N., und Olsen, J. .: The REWARD Service Creation Environment, an Overview. In: *Proceedings ICSLP*. Sydney, Australia. 1998.

[EHH+97] Ehrlich, U., Hanrieder, G., Hitzenberger, L., Heisterkamp, P., Mecklenburg, K., und Regel-Brietzmann, P.: ACCeSS - Automated Call Center through Speech Understanding System. In: *Proceedings EUROSPEECH*. Rhodes, Greece. 1997.

[HWS+03] Hamerich, S. W., Wang, Y.-F. H., Schubert, V., Schless, V., und Igel, S.: XML-Based Dialogue Descriptions in the GEMINI Project. In: *Proceedings of the 'Berliner XML-Tage 2003'*. Berlin, Germany. 2003.

[LSG+00] Lehtinen, G., Safra, S., Gauger, M., Cochard, J.-L., Kaspar, B., Hennecke, M. E., Pardo, J. M., de Córdoba, R., San-Segundo, R., Tsopanoglou, A., Louloudis, D., und Mantakas, M.: IDAS: Interactive Directory Assitance Service. In: *Workshop on 'Voice Operated Telecom Services'*. COST 249. Ghent, Belgium. 2000.

[WHS03] Wang, Y.-F. H., Hamerich, S. W., und Schless, V.: Multi-Modal and Modality Specific Error Handling in the GEMINI Project. In: *Workshop on 'Error Handling in Spoken Dialogue Systems'*. ISCA. Chateau d'Oex, Switzerland. 2003.