EVALITA 2009: Loquendo Spoken Dialog System

Enrico Giraudo and Paolo Baggia

Loquendo S.p.A,
Torino, Italy
{enrico.giraudo, paolo.baggia}@loquendo.com

Abstract. This report describes the implementation of Loquendo's spoken dialog application for the Evalita 2009 Spoken Dialog System (SDS) Evaluation task. The application was designed in VoiceXML and runs on the Loquendo VoxNauta platform. The dialogue strategy is 'mixed-initiative', with flexible recognition grammars that were designed to be modular and easy to use in different dialogue application contexts. Change of context and complex requests from caller are allowed.

Keywords: Spoken Dialog Systems, Speech Recognition, Voice Browsing.

1 Introduction

Evalita 2009 [1] included the evaluation of spoken dialogue systems for the first time. The evaluation task required the participants to develop a telephone application able to access a database on the basis of speech-based interactions. The organizers provided both the common database and a set of scenarios that the callers followed during the trial. The domain was the sales of some brands of food, and the evaluated systems were designed to help the callers, each one playing the role of an individual salesman, to register their orders, to consult previous orders, and to ask various kinds of information about their clients and about the products they sold.

Loquendo Voice Technologies decided to submit a mixed initiative spoken dialogue system to the evaluation: this report describes the component technologies, the platform, and the spoken dialogue strategy adopted in the development of that system. The two-fold goal of the designers of the dialogue strategy was to allow the callers to use natural speech for interacting with the system, and to provide them with a range of strategies to recover from the possible imperfections of speech recognition, and/or the lack of coverage of the speech recognition grammars. The first goal requires designing of modular speech recognition grammars able to deal with complex utterances, and the ability of the dialogue strategy to deal with smooth transitions from one task to another, on the basis of the caller's intentions. On the other hand, the second goal requires balancing the recovery strategies with their costs. In our opinion, reaching a trade-off between those two aspects should improve the task success of the spoken dialogue application (see for example the guidelines in [2]). In particular, the system described in this paper enables task-orientated interactions with the user and, in case of misunderstanding, applies problem-solving

strategies whose goal is to improve the task completion at the cost of increased duration of the interactions [3].

Finally, it is worth noticing that the Loquendo SDS designed for the Evalita 2009 competition was completely designed in VoiceXML, which is one of the most promising emerging technologies for creating spoken dialogue systems [4] and exploits all the most recent standards released by the W3C Voice Browser [5].

The report is organized as follows: the next section describes the architectural aspects of Loquendo SDS Application; Section 3 details the dialogue strategy implemented; Section 4 illustrates the development of the grammars; Section 5 draws conclusions.

2 System Architecture

This section introduces the application domain and mainly describes the architecture selected.

2.1 Application Domain

The Loquendo SDS application was built for the Evalita 2009 SDS Evaluation task [1]. The dialogue application was designed for a Sales Force Automation system that supports sales people to perform actions like entering and transmitting orders for a customer, or asking to check the price or discounts on products in a catalog, and reviewing the status of placed orders. The spoken dialogue application had to work on both mobile or landline phone networks; for this reason it was based on a Voice Browsing platform.

2.2 Voice Browsing

A Voice Browsing has become the standard way of implementing voice application in recent years. There are two main evolutions under way. Firstly, Web paradigm was adopted in the development of speech applications. In the Web paradigm, the application is split into two parts: a voice web application and a general purpose voice browsing platform. The same way the HTML documents are fetched by a visual browser, the VoiceXML documents are fetched by a Voice platform. The Voice platform interprets VoiceXML documents by controlling Automatic Speech Recognition (ASR) and Text-To-Speech (TTS) to realize the dialogue interaction. Secondly, the languages used for defining the speech application are all based on the W3C standards, which means that industry and research can build on a solid and interoperable framework.

W3C Voice Browser working group [5] is the standardization body that created the first generation of Voice Browsing standards[1]. The prominent standards are:

[1] A second generation of the W3C Voice Browser standards is under development. It will include version 3 of VoiceXML and a state chart XML representation called SCXML 1.0.

VoiceXML 2.0 and 2.1 for defining the voice dialogs, CCXML 1.0 for handling call control, SRGS 1.0 and SISR 1.0 for encoding speech recognition grammars (the syntax and the semantics, respectively), SSML 1.0 for defining prompts, and PLS 1.0 for adapting and improving pronunciations. See Figure 1 for the W3C Speech Interface Framework, as designed by James Larson [6] in 2000.

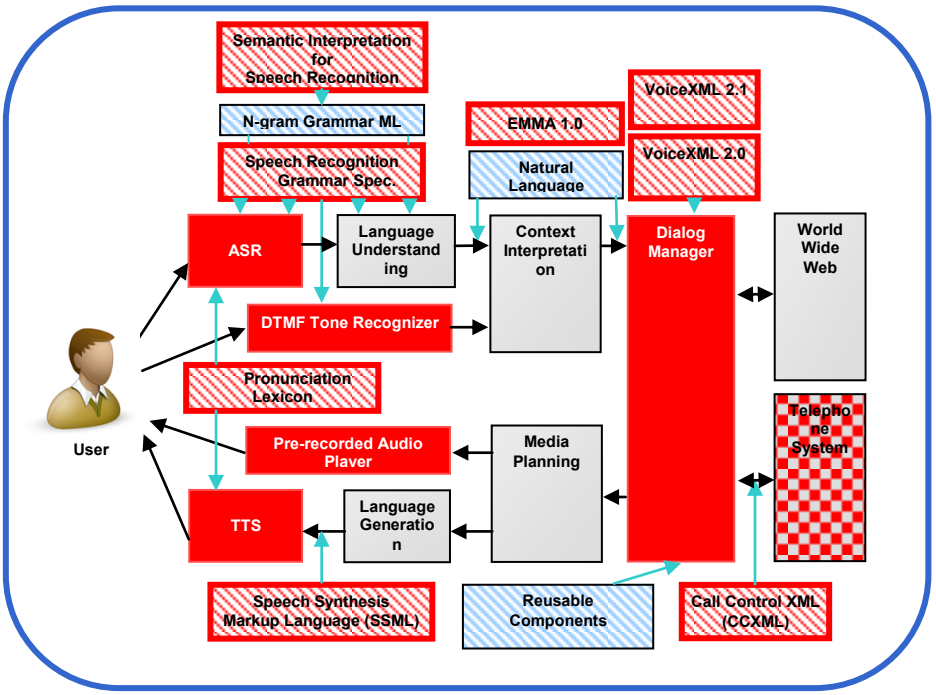

Fig. 1. The W3C Speech Interface Framework by James Larson, chairman of W3C Voice Browser.

Loquendo SDS application was developed on the Loquendo VoxNauta® platform [7], which is VoiceXML/CCXML compliant voice browsing platform. VoxNauta integrates and optimizes the use of Loquendo's ASR and TTS technologies. VoiceXML language allows both static and dynamic applications. Our system was developed as a dynamic VoiceXML application by using JSP technology, whereas all data were stored in a MySQL database.

3 Dialogue Strategy

A dialogue in SDS is a sequence of interactions between a caller (or a user) and a system. These interactions, called dialogue turns, account for prompts and user responses to be recognized by ASR. VoiceXML documents encode the dialogue strategy.

Different dialogue strategies account for different degrees of freedom given to the caller, in commercial applications, the strategy is often 'system-driven' (or 'system-directed') where the caller is focused on the only options allowed by the speech application. This dialogue strategy is rigid and allows the caller to have a very limited amount of freedom.

A more flexible dialogue strategy is called 'mixed-initiative' where the caller is given more freedom to interact and to shift the focus of the dialog.

The Loquendo SDS application was designed as a 'mixed initiative' dialogue to allow the user to select his/her strategy to accomplishing the task in the application domain. Very little work was done to deal with out-of-scope interactions; however a flexible feature was used at the recognition level, by using the SRGS 1.0 special 'garbage' rule (see Section 4 for details).

Moreover, the Loquendo SDS application allows the caller to 'barge-in' while the system is speaking prompts, so that the caller can interrupt prompts to allow a more flexible interaction. The system is also able to recognize conversational input by allowing multiple pieces of information in a single utterance. Finally, the caller can freely switch between different tasks.

A possible risk of the 'mixed-initiative' strategy is the "blank effect" resulting from open prompts that may create difficulties for a novice caller. This effect was balanced by more informative recovery strategies that were meant to progressively provide more information to guide the caller after misrecognition (nomatch) or no answer (noinput). After the first interactions, the caller should find more optimized and compact ways to address the given task.

3.1 Dealing with complex utterances

It might be useful to briefly analyse a complex single-sentence utterance in a single sentence, e.g. if the caller says: "Vorrei sapere il prezzo dei pomodori Del Colle?" (May I know the price of Del Colle's tomatoes?). The application should infer that the task is to check the price of a specific product of a given producer. Then, the application can address the caller's need in one turn.

Example of VoiceXML document from Loquendo SDS application.

```
<form id="menu">
  <grammar
    srcexpr="'gramMenu.jsp?id_rapp='+application.id_rapp"/>

  <block>
    <prompt>
          A quale servizio vuole accedere?
    </prompt>
  </block>

  <field name="servizio"/>
  <field name="prodotto"/>
  <field name="cliente"/>

  <filled mode="any" namelist="servizio prodotto cliente">
    <!-- Validation of results and transition -->
  </filled>
</form>
```

As we can see in the previous example, the dialogue defines only one grammar. This grammar can return semantic results for utterances related to three generic concepts: “servizio” (service), “prodotto” (product) and “cliente” (customer). Not all of these concepts are required to be present in the same utterance. The aim of the grammar is to identify the task that has to be taken care of. Thanks to its encapsulated structure, the grammar is able to return all the required information to immediately accomplish the detected task (see section 4 for more details).

Moreover, if the caller provides only some of the information, the application needs to be able to ask for the missing information.

User:	Mi servirebbe un indirizzo. (I need an address)
System:	A quale cliente è interessato? (Which customer are you interested in?)

In the same way, when a caller concludes a task, s/he can easily jump into another without returning to the main menu.

System:	Vuole ricercare un nuovo articolo? (Would you like to search a new article?)
User:	No, vorrei sapere l’indirizzo del cliente XYZ. (No, I’d like to have the address of the customer XYZ.)

4 Grammar Handling

Traditional SDS applications include speech grammars to activate the recognition process. In general, they are limited to a small set of keywords or short sentences. In our SDS, the grammar was separated as a 'component' whose goal was to recognize a single domain concept, while other grammars are used to specialize component or to 'connectors' components.

Component Grammars mainly consisted of domain concepts, they were generated using database data to capture domain concepts or restricted generalizations. Each concept attribute was described by a specific grammar rule. The meaning representation returned by the grammar were the actual values related to the concepts (e.g. "pomodori" for product or "Del Colle" for producer).

Complex Grammars. Different Component Grammars could be combined to handle complete user utterances, using Semantic Interpretation to assign values to related concepts. A Complex Grammar carried all the recognition capability of the grammars that composed it. The further the caller from achieving the task, the more the grammar complexity grows: component grammars were composed into complex grammars in order to cover articulated utterances from the callers. In this way, all required information to perform an action could be collected at the same time and the dialogue could focus on asking something missing, therefore reducing the complexity in the next turn. The same task could be accomplished by different callers using different strategies in a single turn or in many subsequent turns. A grammar that combines all other grammars gives the caller the maximum expression power.

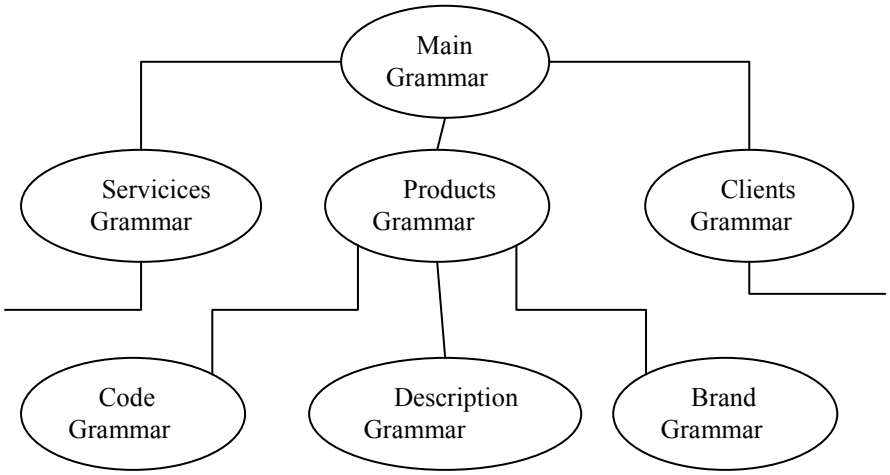

Fig. 2. Composition of grammar modules in the Loquendo SDS application

4.1 Adding flexibility

To add flexibility the SRGS 1.0 special 'garbage' rule was added not only at the beginning or at the end of an utterance, but even between structured concepts.

The advantage is to cover many more user utterances while maintaining the capability of extracting covered sentences and discarding other input. A possible disadvantage is the deletion of small words; this is particularly dangerous, for instance in the case of strings of digits.

The performance of the garbage rule was very good to achieve flexibility and to save costs for grammar development.

5 Conclusions

Evalita 2009 allowed the evaluation of different implementations of dialogue strategies on the same given domain in a common framework.

The Loquendo SDS application is completely based on voice standards and runs on a VoiceXML/CCXML platform (Loquendo VoxNauta). The dialogue strategy aims to allow maximum flexibility to the caller, by exploiting a 'mixed-initiative' strategy.

Spoken grammars were used to allow speech recognition, but they were designed to allow a conversational interaction to allow a 'mixed-initiative' dialogue with the caller. The SRGS 1.0 special 'garbarge' rule was useful for reducing the development effort for generating conversational grammars.

References

1. Baggia, P., Cutugno, F., Danieli, M., Riccardi, G.: Evalita 2009 – Spoken Dalog System Task: Detailed Guidelines, http://evalita.fbk.eu/dialogue.html (2009)
2. Dahl, D. (ed.): Practical Spoken Dialogue Systems. Kluwer Academic Publishers, The Netherlands (2004)
3. Möller, S.: Quality of Telephone Based Spoken Dialogue Systems. Springer Science, New York (2005)
4. Larson, J.A.: VoiceXML. Introduction to Developing Speech Applications. Prentice-Hall, New Jersey (2003)
5. W3C Voice Browser working group, http://www.w3.org/voice
6. Larson, J.A.: Introduction and Overview of W3C Speech Interface Framework, W3C Working Draft, http://www.w3.org/TR/voice-intro/ (2000)
7. Loquendo VoxNauta platform, http://www.loquendo.com/en/technology/voxnauta_platform.htm