# The 2009 UNITN EVALITA Italian Spoken Dialogue System

Stefan Rigo, Evgeny A. Stepanov, Pierluigi Roberti,
Silvia Quarteroni, and Giuseppe Riccardi

Department of Information Engineering and Computer Science (DISI)
University of Trento, Italy
http://casa.disi.unitn.it

**Abstract.** This report describes the conversational system designed at the University of Trento for the Evalita 2009 Spoken Dialogue Systems (SDS) Evaluation task. The main features supporting the UNITN SDS are the mixed initiative control, which allows the caller to get partly in control of the dialog strategy, and the descriptive specification of dialog strategies. The application is based on a complex, high-recall grammar and a user goal planning script. The latter is tightly bound to the grammar and provides functionalities of error checking and recovery from missing or misinterpreted concepts (Automatic Speech Recognition and Spoken Language Understanding errors).

**Keywords:** Spoken Dialogue Systems, Spoken Language Understanding, Automatic Speech Recognition, Evaluation Metrics.

## 1 Introduction

One of the eight tasks of EVALITA 2009 – the $2^{nd}$ Italian national evaluation campaign of Natural Language Processing (NLP) tools for Italian – is Spoken Dialogue System (SDS) Evaluation. The goal of the task is to develop an information seeking voice application for a specific domain and evaluate it with callers following scenarios provided by the task organizers. The domain of Evalita 2009 SDS Evaluation is sales force and the participant systems' objective is to provide assistance to sales representatives [1].

This report describes the mixed initiative Spoken Dialogue System developed at DISI, University of Trento for participation in the task. State of the Art unconstrained conversational systems very often make use of Stochastic Language Model based recognition and Stochastic Conceptual Language Model based understanding [2], that can also be complemented by other machine learning techniques (e.g. [3]). Data-driven methods have gained popularity for Dialogue Management as well. However, due to the lack of in-domain data, the system described in this report makes use only of grammar-based ASR and SLU and rule-based dialogue management.

The main theme of the SDS development was to allow callers to issue task requests to the system in a conversational manner, i.e. without imposing a predefined structure or vocabulary constraints to the utterances. Moreover, the application was intended to be

able to accept any request of the implemented tasks in any "appropriate"[1] turn throughout the dialogue. To achieve this goal, a complex grammar was manually built with the focus on being close to the "natural" language use. This grammar is used in all the prompts, except the ones requesting confirmation, i.e. yes/no response, or a specific piece of information such as customer or product names.

The report is structured as follows. In Section 2 we describe the system our application runs on and the overall design of the SDS with separate sections devoted to dialogue strategy and grammar. In Section 3 we provide the evaluation results of our SDS together with the discussion. Section 4 provides some concluding remarks.

## 2 UniTN System Architecture

The University of Trento SDS application was built on the Voice Multi-Modal Application (VMMA) Framework [6] developed at the AMI2 Lab of University of Trento. The framework is based on VoiceXML and EMMA standards. The Automatic Speech Recognition (ASR) and Text-to-Speech Synthesis (TTS) functionalities are provided by the Loquendo VoxNauta [4] platform. VMMA framework allows for designing rule-based multi-modal dialogue system applications without specifying the whole structure of the call-flow [5], which is the case with purely VoiceXML based applications. Such a design resembles production rule systems in that rules are triggered based on the conditions they match. An application is described using several configuration files, some of which are essential for speech based application:

**Concept Ontology** : defining all the concepts and their relations, which are essentially slots to be filled;

**Action Ontology** : containing a set of actions with their connections to concepts (see Concept Ontology). This information is used by Dialogue Manager (DM) to access the concepts associated with particular action.

**User Goal Planning** : defining the triggering conditions for each action. This file contains information about the different contexts (use cases) the system can manage. The DM uses this information to select the next move. In case more than one action matches specified conditions, a random action is selected. The script also allows to specify the code (PHP) to be run by the DM before and after the action execution.

**Speech Turn Templates** : for each action defining prompts, grammars and various parameters used for VoiceXML page generation;

**Speech Server Settings** : defining default VoiceXML parameters.

Since the VMMA Framework is multi-modal, there are separate configuration files for visual components as well; however, they are out of scope of this report.

The call-flow is essentially driven by two main components: a user goal planning script and a grammar, which fills the slots used as conditions for action selection. The relation among configuration files in a call-flow is depicted on Figure 1. The following subsection provides more details on this feature.

[1] E.g. confirmation turns do not qualify as "appropriate"

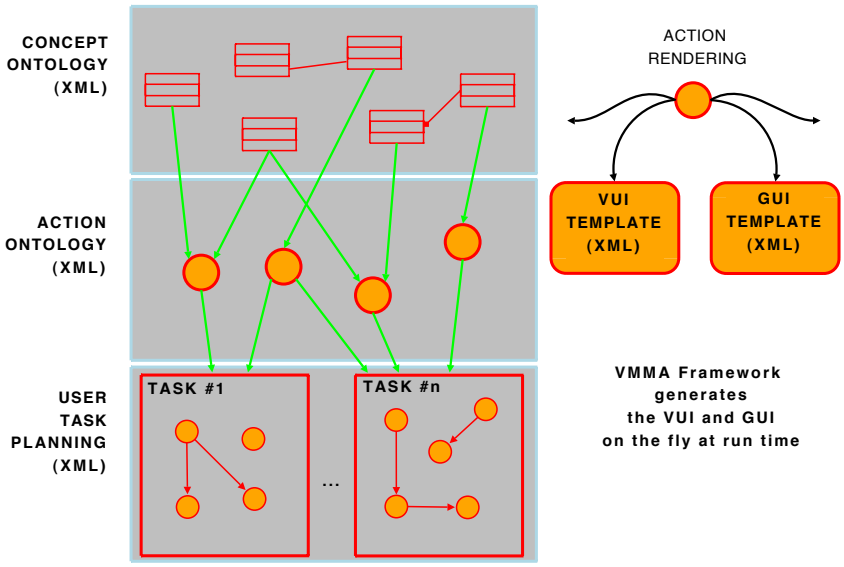


**Fig. 1.** VMMA Framework Call-Flow Description

### 2.1 Dialogue Strategy

The application is modularly built around three main action contexts :

1. the context for action selection, triggered after caller identification;
2. the context for catalog search related tasks; and
3. the context for customer related tasks.

The first context is essentially used to select among the latter two action contexts. The catalog and customer related contexts are structured similarly. Both have a default opening prompt used for requesting a task the caller would like to perform, and any of the implemented task could be selected. The system falls back to these default actions in case of error in other actions of the context. Similarly, both contexts have an open prompt for an action listing the information retrieved from the database. These open prompts use the main grammar. The remaining actions of the contexts are triggered in case a task was already requested, but some required information was not provided by the caller.

Complex task requests containing up to six concepts (i.e. slots to be filled) can be issued by the caller and successfully handled by the system. Given the large number of concepts, some basic memory features were implemented to ease the task from both system and user perspective. For instance, customer names and product information are saved for later use and in case the caller does not provide them in next task they are used to automatically fill the required slots; and confirmation is requested afterwards.

As an error-handling strategy, confirmation actions are always triggered when “No-Match” level of the concept is below a given threshold (e.g. 0.5). There is another kind

of yes/no confirmation prompt as well, which is triggered by the user goal planning script for the tasks considered to be sensitive, such as caller identification and writing to the database.

Moreover, there is a set of actions dealing with various error codes returned from the database interaction scripts. Finally, the caller can request basic help on interaction with the system, as well.

### 2.2 Grammars

In order to support a mixed-initiative interaction, the dialogue should have the ability to handle input not only covering the information expected by the system, but also some extraneous information. This is achieved by a complex grammar, which provides semantic interpretations of utterances, i.e. sets values for concepts, and passes them to the user goal planning script for next action selection.

On an implementation level, this is done by extending ABNF-grammar script. From the evidence collected from caller utterances, the grammar was tuned to catch the relevant concepts uttered by the users and decrease the possible misrecognition. Information regarding the most frequent words used to refer to concepts in various tasks, as well as the relative positions of most frequent "fillers" used together with these concept words was analyzed. This analysis coupled with careful placement of the special *$GARBAGE* rule, used to cover all irrelevant input, resulted in a "high-recall" grammar intended to cover the widest possible range of caller utterances. Such a grammar covers all possible actions implemented in the system, having a rule for each task request.

Nine grammars were built for the application with the certain amount of overlap in contents. The main – "high recall" – grammar is active in all open prompts and responsible for mixed-initiative dialogue. Several other grammars (3), limited to specific action context, are used for requesting information related to customers and product catalog. A number of grammars (5) were built for prompts requesting a specific piece of information[2], such as confirmations, user IDs, product names, and quantities.

## 3 Evaluation Results and Discussion

Evaluation of the system was carried on according to EVALITA guidelines. In this section we report on the evaluation results of the University of Trento Spoken Dialogue System participating in the task.

The evaluation results provided in this section were collected using UniTN Dialogue Statistics Web-Tool. 20 of dialogues were transcribed and annotated per participant system. The annotation formed basis for task-level metrics. Duration oriented dialogue-level metrics were computed on the same set of dialogues.

[2] Which usually occurs when the system either catches "NoMatch" event, executes "sensitive" action, or caller has only partially provided information required for certain task to be accomplished

### 3.1 Dialogue-level statistics

Duration related dialogue-level statistics reported for the system are the average dialogue duration and standard deviation both in seconds and the number of turns (see Table 1 ). The average dialogue duration for UniTN SDS is 206.4 seconds, which is quite high. This is partially caused by the system's explicit confirmations for "sensitive" tasks such as *identify representative* and *new order*. Moreover, the fact that UniTN dialogues contained high number of tasks per dialogue – 5.75 tasks per dialogue – also contributed to high average dialogue duration. Still, another reason is the feature of the system to assess confidence values for each concept in a multiple concept task, and accept ones with high confidence while reprompting for others; rather than assessing the confidence of whole utterance and rejecting it in case of low value.

**Table 1.** Dialogue-level statistics: Duration

| **Dialogue Duration** | |
|---|---|
| **seconds** | $206.4 \pm 81.7$ |
| **# of turns** | $24.4 \pm 10.1$ |

### 3.2 Task-level statistics

The system was evaluated with respect to two kinds of task-level statistics: task duration and task success rate (see Table 2). Task duration, measured in number of turns, gives a general idea about how long it takes to accomplish a task. Such a number may be at the same time due to information requirements of the task and a possible number of confirmation turns in the dialogue. In average, all tasks took around 3 turns to complete. However, a task of recording a new order took longer (7.5 turns on average). For the UniTN system this is caused by the forced confirmation turn for this task, this task requiring the system to acquire the highest number of concepts from the caller, and the concept confidence assessment policy described in previous subsection.

Task success rate [7] is measured as a ratio of successfully completed tasks to the total number of requested tasks of the same type as: $tsr_C(t_i) = corr_C(t_i)/req_C(t_i)$, where $t_i$ is a task in a dialogue collection $C$, and $corr_C(t_i)$ and $req_C(t_i)$ is a number of successfully completed and requested tasks of that kind, respectively. From Table 2 we can see that the UniTN system has lower task success rate for *exit application* task. In this case the system failed to complete the task due to the grammar involved. Sometimes the request to exit was misrecognized as another concept (e.g. product). However, in most cases the reason was "NoInput" or "NoMatch" event, i.e. the input was out of grammar.

**Table 2.** Task-level statistics: Task Duration & Task Success Rate

| **Task-level Statistics** | | |
|---|---|---|
| **Task** | **Duration** (turns) | **TSR** (corr/req) |
| Identify representative | $3.1 \pm 0.5$ | 90.5% (19/21) |
| Ask customer detail | $3.4 \pm 1.6$ | 54.6% (12/22) |
| List orders | $3.0 \pm 0.0$ | 75.0% (3/4) |
| Show last order | - | - |
| List customers | $3.0 \pm 0.0$ | 66.7% (2/3) |
| New order | $7.5 \pm 2.8$ | 63.2% (12/19) |
| List products by category | $3.0 \pm 0.0$ | 100.0% (3/3) |
| List products by brand | $3.0 \pm 0.0$ | 50.0% (1/2) |
| List products - other | $3.8 \pm 1.6$ | 44.4% (4/9) |
| Search single product | $3.5 \pm 2.5$ | 78.6% (11/14) |
| Ask for help | $2.0 \pm 0.0$ | 100.0% (2/2) |
| Exit application | $2.4 \pm 0.8$ | 25.0% (4/16) |
| Total | - | 63.5% (73/115) |

## 4 Conclusions

We have described the University of Trento system developed for Evalita 2009 Dialogue System Evaluation task. The task success rate for the system is 63.5% and the average dialogue duration is 24 turns. Given that there were 5.75 tasks in average per dialogue, a task took around 4 turns in average. These indicate that there is a room for improvement in both grammar and user goal planning. The main strength of the system is its being mixed initiative while retaining good control over tasks – the system was able to correctly interpret task request with many concepts most of the time. This, at the same time, causes the weak point of the system – the main grammar sometimes caused task failures for simple action such as *exit application*.

## References


1. Baggia, P., Cutugno, F., Danieli, M., Pieraccini, R., Riccardi, G.: Evalita 2009 - spoken dialogue system task: Detailed guidelines, http://evalita.fbk.eu/dialogue.html (2009)
2. Dinarelli, M., Moschitti, A., Riccardi, G.: Re-ranking models for spoken language understanding. In: Proceedings of EACL2009. Athens, Greece (2009)
3. Dinarelli, M., Stepanov,E.A., Varges, S., Riccardi, G.: The luna spoken dialogue system: Beyond utterance classification. Submitted to ICASSP 2010 (2010)
4. Loquendo, http://www.loquendo.com.
5. Riccardi, G., Baggia, P., Roberti, P.: Spoken dialog systems: From theory to technology. In: Proceedings of Workshop Toni Mian. Padua, Italy (2007)
6. Riccardi, G., Mosca, N., Roberti, P., Baggia, P.: The voice multimodal application framework. In: Proceedings of AVOIS 2009. San Diego, CA (2009)
7. Varges, S., Riccardi, G., Quarteroni, S.: Persistent information state in a data-centric architecture. In: Proceedings of SiGdial 2008. Columbus, Ohio (2008)