A Flexible Framework for Evaluation of New Algorithms for Dialogue Systems

Pavel Cenek

Department of Information Technology, Faculty of Informatics
Masaryk University Brno, Czech Republic
xcenek@fi.muni.cz
Currently affiliated with Norut Information Technology Ltd,
Tromsø, Norway
pavel.cenek@itek.norut.no

Abstract. Research in the field of dialogue systems often involves building a dialogue system used for evaluation of algorithms, collection of data and various experiments. A significant amount of time is needed to create such a system. In order to facilitate this task, we created a flexible, extensible and easy to use framework which can be used as a base for experimenting with dialogue systems. Major features of the framework are introduced in the paper together with possible ways of their practical use.

1 Introduction

Research in the field of dialogue systems often involves building an experimental dialogue system which is used for statistical data collection, various measurements, performance tests, etc. A significant amount of time is spent on creation of such a system before the scientific work itself can start.

This paper introduces Elvira - a flexible, extensible and easy to use framework for building, evaluating and experimenting with dialogue systems. The framework makes it possible to concentrate on the scientific problem itself and can be easily used for a wide range of research tasks and experiments, e.g. discourse analysis, research in the field of dialogue strategies, statistical processing of dialogues, testing methods of natural language understanding and robust speech recognition, modeling of prosody and other human language technologies related to dialogue systems. ([1], [2], [3]).

2 Dialogue Flow Description

A flexible dialogue framework should support an easy to use method of the dialogue flow description with possibilities for easy modifications of the dialogue. In this aspect Elvira relies on VoiceXML ([4], [5]) which is generally accepted as a standard for the dialogue flow description. It is designed for description of information retrieval dialogues ([6]) and in addition to that the dialogue model used in VoiceXML allows to very simply distinguish each dialogue step and its phases.

The VoiceXML specification describes required and optional features of each VoiceXML interpreter, but says very little about ways of their implementation. Elvira offers an implementation which fulfills requirements of the VoiceXML specification and adds many features VoiceXML is not primarily designed for. This is achieved by the use of a sophisticated architecture rather than use of proprietary extensions of VoiceXML. Elvira features include: running of the dialogue using various sources of user's input including speech recognition, various kinds of output including speech synthesis and various grammars for defining expected user's input, broad logging possibilities and more.

Major features are described in the following section.

3 Elvira's Features and Possible Ways of Their Use

3.1 Extensibility and Modularity

Extensibility and modularity of the framework is achieved by the use of a component based architecture.

Elvira consists of several components. Each of them provides a set of precisely defined interfaces for communication with other components. The rest of the component is private and invisible for the rest of the world. Therefore it is very simple to replace one component by another with a different implementation as long as the other component implements the same interfaces.

Components can be linked to the core of the framework (also called *Elvira core*) dynamically, therefore a component can be replaced by another without touching the rest of the system.

Components have to implement only some "mandatory" interfaces which are essential for successful run of the system. By implementing other interfaces they can provide an extended functionality if it is useful for the current application. The situation when an interface is not implemented can be easily detected and Elvira core can avoid the use of such non-implemented interface. This principle allows researchers to deal only with things relevant for their current experiment.

Every component is characterized by its unique name and by its category. It ensures the great flexibility of Elvira. The names are typically used in the configuration files to specify which component should be used for a specific task in the system. Thus the composition and features of your current dialogue system can be determined just by changing a configuration file. In the configuration file, one can specify, among other characteristics, the input source, output destination and what kind of events should be logged and which components should be used for logging.

A component can also be picked based on its category. It is for instance used for the automatic support of new grammar analyzers. Each component ensuring grammar analysis defines its category so that it contains mime-type of the supported grammar format and every grammar used in a VoiceXML document specifies its mime-type. When Elvira core needs a specific grammar analyzer, it simply uses a component with the right category. Thus everything what we need to support a new grammar type is to copy the proper component into a location where Elvira core can find it.

3.2 Output

As described above, output destination and features are dependent on the currently used output component. The most common types are textual output and speech output.

The textual output component is very simple to implement and the output itself is faster then speech output. It can save time in the debug and test phase and it can be easily replaced by a speech output in the final version of the system.

The output component is a good example of a component with an optional interface. Every output component can optionally implement an interface for processing the speech prosody. If the interface is available, Elvira core uses this interface for passing prosodic information from VoiceXML to the output component. A component which uses its own mechanism for prosody modeling or generates no prosody at all simply does not implement this interface.

3.3 Input and Grammars

Input source and capabilities depend on the currently used input component. Possible source types include, but are not limited to, keyboard, microphone, phone or text file.

Keyboard input is very simple and unlike speech recognition does not suffer from input recognition errors. It can be very well used for example for testing of some methods of natural language understanding. Input from a text file can be used for simulations without need of a human operator.

A microphone or a phone is typically used as input for real dialogue applications. Every input component has access to the list of currently active grammars. The grammar components can optionally provide interfaces for collaboration with the input component. Based on information from the grammars, a model of the expected user's input can be built for the speech recognizer. Communication between the input component and the active grammars is in no way influenced or restricted by the Elvira core. It opens large space for testing various kinds of algorithms of robust speech recognition.

The recognized utterance is analyzed using active grammars, and semantic information is returned. The semantic information is assigned into slots and then processed by the dialogue logic described by means of VoiceXML. The complexity of the semantic information is not limited by Elvira, it is the responsibility of the dialogue logic to deal with it. Since a true scripting language is a part of VoiceXML, the possibilities for the representation of semantics and its processing are very broad.

Also grammar analyzers can be of various kind. They can be very simple and recognize only semantics of words from a predefined list, or they can be much more complex using sophisticated language models extracting semantics from free form speech. Elvira can serve as a good framework for both cases.

3.4 Logging

Many research activities related to dialogue systems are based on statistical methods. Elvira can be used for collecting statistical data in various formats. Every interesting event in the Elvira core or another component can be logged. Of course Elvira uses components for this purpose.

An application can define which events should be logged and can assign different components for the logging of different events. It allows the creation of several separated logs containing only information which should be kept and processed together.

4 Conclusion

Elvira is a flexible framework for the work with dialogue systems. Only the basic characteristics were introduced in this paper. Also only very simple ways of their use were described here. Description of more sophisticated techniques is beyond the scope of this paper.

The utility of the framework also depends on the availability of various components. The basic distribution of Elvira offers some simple input, output and grammar components, so that it can be tested without any programming. Source codes of some components are also available and can serve as a base for development of other components. The repository of available components for Elvira will probably grow in time, making its utility higher and higher.

Elvira is available at http://www.fi.muni.cz/lsd/elvira.

References

1. De Mori, Renato: Spoken Dialogues with Computers. Academic Press, London, 1997.
2. Smith, R. V. , Hipp, D. R.: Spoken Natural Language Dialog Systems - A Practical Approach. Oxford University Press, 1994.
3. Cole, Ronald, et al., eds: Survey of the State of the Art in Human Language Technology, 21. November 1995. http://cslu.cse.ogi.edu/HLTsurvey/HLTsurvey.html last visited 30. March 2002.
4. VoiceXML Forum: Voice Extensible Markup Language (VoiceXML) Version 1.0. 7. March 2000. http://www.voicexml.org/specs/VoiceXML-100.pdf last visited 30. March 2002.
5. McGlashan, Scott, et al., eds: Voice Extensible Markup Language (VoiceXML) Version 2.0; W3C Working Draft, 23. October 2001. http://www.w3.org/TR/2001/WD-voicexml20-20011023/ last visited 30. March 2002.
6. Kopecek, I.: Modeling of the Information Retrieval Dialogue Systems; in Proceedings of the Workshop on Text, Speech and Dialogue - TSD'99, Lectures Notes in Artificial Intelligence 1692, Springer-Verlag, 1999, pp. 302-307.