FI MU

Faculty of Informatics
Masaryk University

Improvements in a Dialogue Interface for Library System

by

Luděk Bártek

FI MU Report Series **FIMU-RS-2002-07**

 October 2002

Improvements in a Dialogue Interface for Library System

Luděk Bártek
bar@fi.muni.cz

Faculty of Informatics Masaryk University Brno
Botanická 68a, 602 00 Czech Republic

Abstract

This article describes methods and algorithms used to improve the dialogue interface of Library system at Faculty of Informatics (see [1]). This interface has been developed to enhance the accessibility of the system to the people with special needs as well as to enable telephone access to the system in the near future.

1 Introduction

The advantage of the dialogue interface is that it can reduce the amount of information transfered to the user.

Many users prefer either the user initiative dialogue interfaces or the mixed initiative dialogue interfaces to system initiative ones. Users are ready to use the system initiative dialogue only when they don't know what input values are expected. But the first version of the dialogue interface uses system initiative dialogue strategy.

2 Basic Scheme

The dialogue interface is based on VoiceXML interpreter Elvira (see [2]). Elvira provides the communication interface between the user and the library system as shown in figure 1.

The first version of the interface was limited by the Form Interpretation Algorithm (FIA) defined in VoiceXML (see [3]) and by the possibilities of

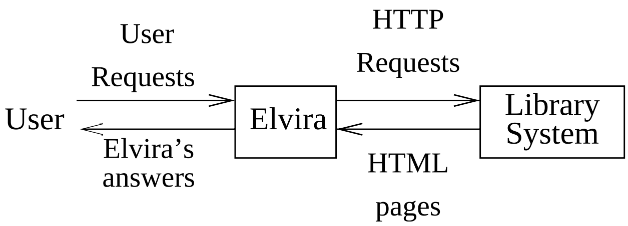

Figure 1: Basic Communication Scheme

grammars supported by Elvira, JavaSpeech Grammar Format (JSGF, see [5]) and Laboratory of Speech and Dialogue Regular Grammar (LSD-RG, see [2]). The grammar formats do not allow us to perform information extraction from the natural language sentence directly. In the first version of the interface the dialogue may look for example like this:

System: Enter either the surname of the author or some initial letters.
Press enter if you don't know the author.
User: *enters some values*
System: Enter either the title of the book or any part of it. Press enter
if you don't know the title.
User: *enters some values*
...

We have decided to modify the dialogue strategy to solve the problem of information extraction in following way:

1. The system gives a short description of its function to the user.

2. User describes the request.

3. VoiceXML pass this sentence (without processing) to the ECMAScript (see [4]) function that analysis the sentence and performs the information extraction (the principles will be described later).

4. The extracted values are assigned to VoiceXML variables. The names of the variables are defined in the rules used by the extraction algorithm (the grammar of the rules will be described later).

3 Information Extraction Algorithm and Its Implementation

This section discusses possible methods of extracting useful information from the user answers in VoiceXML. We also describe why we decided to use the ECMAScript-based solution as mentioned in the previous section.

In VoiceXML, the possibilities of extracting data from an user's responses are following:

- Rules in some of VoiceXML grammar format
- Implementation of some information extraction algorithm in ECMAScript
- Using VoiceXML objects stored in dynamically linked libraries implementing some information extraction algorithm

3.1 Grammars Based Solution

This is pure VoiceXML solution. However, it has the following disadvantages:

- The values to be returned must be strictly defined in the grammar. All authors names and titles of books cannot be specified in the grammar.
- The grammars allow to return only one value per rule while the user answer can contain multiple values (for instance the name of the author and title of the book).

3.2 ECMAScript Based Solution

The ECMAScript is used in the VoiceXML as a default scripting language. Using ECMAScript we can eliminate the disadvantages described in the previous section.

The only disadvantage of the solution may be its time requirements.

3.3 VoiceXML Objects Based Solution

This solution eliminates the possible disadvantage of the ECMAScript implementation (its speed), but it has the following disadvantages:

- Dynamically linked libraries are platform dependent, we need to recompile the library for each platform, where the system is running
- The realization of the OBJECT tag is not strictly defined in the VoiceXML specification. Every VoiceXML interpreter can implement the tag differently.

According to the previous paragraphs we decided to implement the information extraction algorithm using ECMAScript.

3.4 Information Extraction Algorithm

The chosen solution is realized by the following algorithm:

1. for all available rules do
    (a) if the rule matches the user sentence
        i. extract requested values from the user sentence
        ii. return to the dialogue
2. assign empty strings to the requested values

When the algorithm fails to extract the requested information, the VoiceXML starts a safe dialogue that asks the user for all required missing slots.

Similar dialogue is used when some information is incorrectly recognized.

The chosen algorithm is easy to implement using the ECMAScript and thanks the safe dialogue is robust enough.

3.5 Format of the Used Rules

We have chosen simplified Unix regular expression format as the rule format. But we made some modifications that allow us to define the semantical information used to assign the value(s) from the answer to the slot variables.

The regular expression may contain following constructions:

- wildcards (* – any string, ? – any character)
- optional parts – closed in parenthesis (for example (od))
- parts to be assigned as value of variable – closed in braces (for example: from {X})
- text

An example of a rule in the format:

I would like (a book) {book_name} by (author) {author}.

This rule will match for example the sentence:"I would like a book Psohlavci by Alois Jirasek."

The algorithm will assign Psohlavci to the variable book_name and Alois Jirasek to the variable author then.

4 Future Plans

Some of the tasks to be solved in the near future are:

- Extending the initial set of rules using the Wizard of Oz method (see [6])
- Designing the algorithm for automatic sorting of grammar rules – we propose to sort the the rules using the following principle: "How difficult it is to match the particular rule?" The rules will be sorted in descending order.
- Implementation of machine learning techniques that will allow to derive semi-automatically new rules from the user's answers. The techniques will be based on comparing the user's answers where the algorithm failed to extract the information to the slots filled by the safe dialogue. The algorithm will need a human help to eliminate the incorrect rules derived by the algorithm.

5 Acknowledgment

I would like to thank to my colleagues at Laboratory of Speech and Dialogue for their valuable hints and reading the draft of the paper.

This work has been partially supported by The Ministry of Education of Czech Republic under the grant LI200027.

References

[1] Cenek, P. - Dialogue Interfaces for Library Systems, June 2001. FIMU-RS-2001-04

[2] ELVIRA - Extensible LSD VoiceXML Interpreter for Dialog Applications.See http://gin2.itek.norut.no/elvira/

[3] Voice Extensible Markup Language (Voice-XML) Version 2.0. See http://www.w3.org/TR/voicexml20/, W3C Working Draft, April 24th 2002

[4] Standard ECMA-262 ECMAScript Language Specification, Standard ECMA-262, December 1999. See http://www.ecma.ch/ecma1/STAND/ECMA-262.htm

[5] Hunt, A. - JSpeech Grammar Format, W3C Note, June 2000. See http://www.w3.org/TR/2000/NOTE-jsgf-20000605/

[6] Fical, J. - Design of Dialogue application using the Wizard of Oz Method (in Czech), diploma thesis, Faculty of Informatics Masaryk Univesity, Brno 2001.

[7] Kopecek, I., Bartosek, M. - Developing Dialogue Library System, ICADL 2002 Conference, to appear.

Publications in the FI MU Report Series are in general accessible via WWW and anonymous FTP:

`http://www.fi.muni.cz/informatics/reports/`
`ftp  ftp.fi.muni.cz (cd pub/reports)`

Copies may be also obtained by contacting:

Faculty of Informatics
Masaryk University
Botanická 68a
602 00 Brno
Czech Republic