Communication by Voice to Interrogate an Inheritance Based Knowledge System

Nicolae ŢĂNDĂREANU

Faculty of Mathematics and Computer Science,
Department of Computer Science,
University of Craiova, Romania
ntand@rdslink.ro

Abstract. In this paper we describe the architecture of a system to communicate by voice with an inheritance based knowledge system. The speech synthesis part to answer by voice at some interrogation is treated in [8]. The speech recognition part was left as a future work an we treat in this paper the problem of the interrogation by voice. In this way the VoSys system described in [8] becomes a complete system of communication by voice with an inheritance based knowledge system. The implementation of this system is also presented and the communication by voice is exemplified.

Keywords: inheritance based knowledge system, speech technology, voice user interface, recursive transition network
Math. Subject Classification 2000: 68T30, 68T35

1 Introduction

The management of the communication between two entities (computer-user, robot-robot, robot-user, agent-agent) is a research subject with implications in various domains and applications of general interest. Various structures were implied to describe this communication: Augmented Transition Networks and Recursive Transition Networks for natural language representation ([1]), general automata for communication in human-computer systems([2]), semantic schemas for communication computer-user in knowledge based systems ([7]), communication between the components of a master-slave systems based on semantic schemas ([9]) and so on. More and more these structures are used to build and implement graphical user interfaces and moreover, interfaces by voice.

The study presented in this paper was suggested by the following facts:

- Several previous works proposed a model for inheritance based knowledge systems and using some lattices of ordered trees the computability properties of the answer function for such systems were studied ([4], [5] and [6]).
- A possible implementation in Java of such systems was described in [8]. A graphical user interface (GUI) was used to communicate with the system.

We can interrogate the system if we select an object Ob and an attribute Attr in GUI. If we press some button then the value Ans(Ob,Attr) of the answer mapping is computed. The result is displayed in some window of GUI by a sentence in a natural language or can be given by voice if a proper button is pressed.

- There are not many knowledge based systems incorporating natural language interfaces ([3]). Moreover, the *communication by voice* in such systems is a subject insufficient developed today due to difficult problems that arise in this process: the use of speech technology, the extraction of the semantics from the spoken sentences, the natural language representation, the connection with the reasoning module of the system.

In this paper we propose a model to use and implement a communication by voice between user and system for inheritance based knowledge systems. The use of speech technology to answer at the interrogation for such systems was presented in [8]. In this paper we append a module allowing the *interrogation by voice*. The following aspects are treated:

- The use of recursive transition networks to describe the syntax of the spoken sentences given at interrogation.
- The extraction of the semantics from the input sentence. This process takes into account the fact that using an inheritance knowledge base, any interrogation specifies an object and an attribute. Consequently, from the input sentence we extract these two entities.
- The architecture of the proposed system is presented.

The system was implemented in Java. To implement the voice communication we used JSAPI and Sphinx.

The paper is organized as follows: in Section 2 several preliminary notions concerning the speech technology and interfaces by voices are presented; in Section 3 we present the architecture of a system endowed with voice communication for an inheritance based knowledge system; the use of a Recursive Transition Network to specify the syntax of the spoken sentences is also presented in this section; Section 4 describes the implementation in Java of the proposed architecture; several conclusions of our study are presented in the last section.

2 Speech technology and voice user interfaces

In this section we present several basic concepts concerning the interfaces by voice.

From our point of view an *interface* is an aggregate of means by which a user can communicate with a system in order to perform some actions (by the user or by the system). As can be viewed in Figure 1 an interface is placed between a system and user. The system as well as the user can perform some actions as a result of the communication process. A user interface provides means of:

• **Input**: the user sends a message to system (commands for operating system, actions on a process, an interrogation etc);

• **Output**: the user receives a message from system (an answer for an interrogation, the result of a decision making etc)

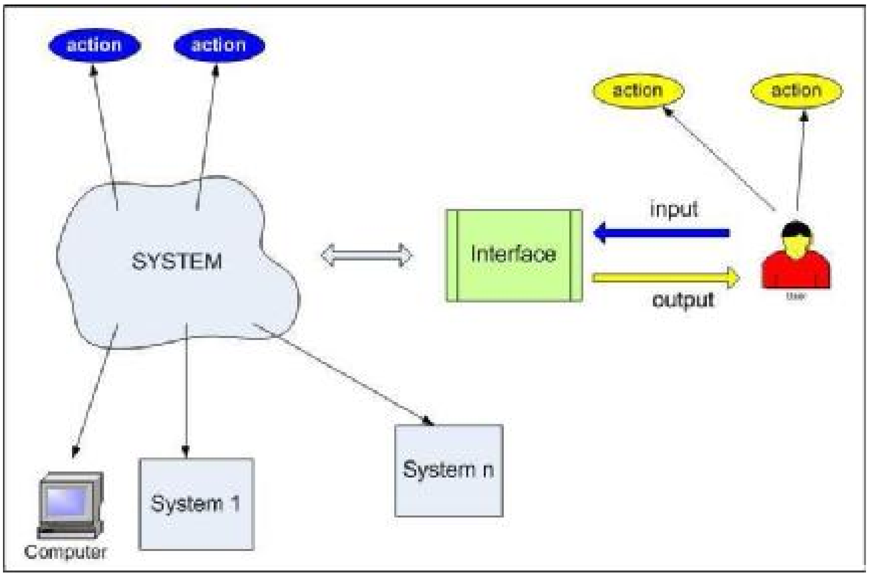


Fig. 1. Cooperation user-system by an interface

A *Voice User Interface* (VUI) is a user interface such that both the input and output are accomplished by voice communication. But a VUI can not be reduced to a simple communication by voice. In our opinion an adequate characterization of a VUI can be given by the following equation:

VUI=Speech Processing+Knowledge Management

The Speech Processing comprises two sides: *Speech Synthesis* and *Speech Recognition*. The Speech Synthesis produces synthetic speech from text. It is often referred to as *text-to-speech* technology (TTS). The converse process is given by Speech Recognition: from the spoken sentence this component determines what has been said. In other words, it processes audio input containing speech by converting it to text.

The software used in this paper to implement the voice user interface includes Java Speech API 2.0 and Sphinx 4.01 (Figure 2). *Java Speech API* was developed by Sun Microsystems, Inc. in collaboration with leading speech technology companies: Apple Computer, Inc., AT & T, Dragon Systems, Inc., IBM Corporation, Novell, Inc., Philips Speech Processing, and Texas Instruments Incorporated. *Sphinx* was created by a joint collaboration between the Sphinx group at Carnegie Mellon University, Sun Microsystems Laboratories, Mitsubishi Electric Research Labs (MERL), Hewlett Packard (HP), University of California at Santa Cruz (UCSC) and the Massachusetts Institute of Technology (MIT). Java Speech API is a standard extension to the Java platform that enables Java applications and applets to use speech input and output. It is not difficult to use this software product for *speech synthesis*. Sphinx-4 is a very flexible system capable of performing many different types of *speech recognition* tasks.

"Speech engine" is a software system designed to deal with either *speech input* or *speech output*.

The javax.speech package of the Java Speech API defines an abstract software representation of a speech engine. *Speech synthesizers* and *speech recognizers* are both speech engine instances.

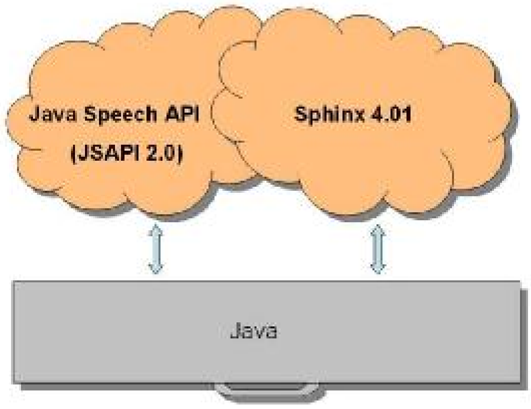


Fig. 2. Software for voice communication

3 A system by voice to communicate with an inheritance knowledge base

In this section we propose a possible architecture of a system allowing to communicate by voice with the user in order to interrogate a knowledge base of an inheritance based knowledge system.

In order to implement the system we used Java technologies ($jdk1.5.0\_02$, $Java\ Speech\ API$, $Sphinx4.01$), the language $Prolog$ and a bidirectional connection Java-Prolog (the product $JIProlog$ $v3.0.2-8$ of Ugo Chirico, [10]). As auxiliary software we used *Apache Ant* and XML to allocate the resources for speech technology.

3.1 Architecture of the system

The architecture of the system is represented in Figure 3. There are two modules in Prolog:

- The *inference module* IM is represented by 5 files:
  - **lib.load**: load the libraries *jipxterm.jar* and *jipxsystem* which are used to build the answer as a string;
  - **inherit_engine.pl**: accomplishes the computation of the answer mapping $Comp_{\mathcal{K}}$;
  - **init_comp.pl**: several steps necessary to initiate the computation of the answer mapping $Comp_{\mathcal{K}}$;
  - **list_obj.pl**: obtain the list of the objects from the file **Kb_name.pl**;
  - **list_attr.pl**: obtain the list of the attributes from the file **Kb_name.pl**.
- The *specific module SM* is composed from 4 files:
  - **Kb_name.kb**: this file specifies the name of the knowledge base; the selection of this name is the starting point of the application;
  - **Kb_name.pl**: the main part of this file defines the content of the knowledge base;
  - **Kb_name.txt**: the content of this file is displayed if the user wishes to see the knowledge base content;
  - **Kb_name.ans**: this file builds the sentences associated to answer.

We observe that the module IM does not depend on the feature of the knowledge base. The second module, which is not a component of the system, depends on this feature (the content of the knowledge base, the output sentences of the answer mapping etc). These modules, as well as the computation of the answer mapping are presented in detail in [8].

We observe that the system includes two interfaces:

- A **Graphical User Interface**, allowing the communication user-system by text and action buttons;
- A **Voice user Interface**, allowing the communication user-system by voice.

As we can observe in Figure 4 there are three kinds of communications:

- **Text to Text** communication: the interrogation and the answer are given by text using GUI;
- **Text to Voice** communication: interrogation by text, the answer by voice and text;
- **Voice to Text** communication: interrogation by voice, the answer by voice and text.

We mention the fact that answer by voice is reinforced by text because the communication by voice is not persistent.

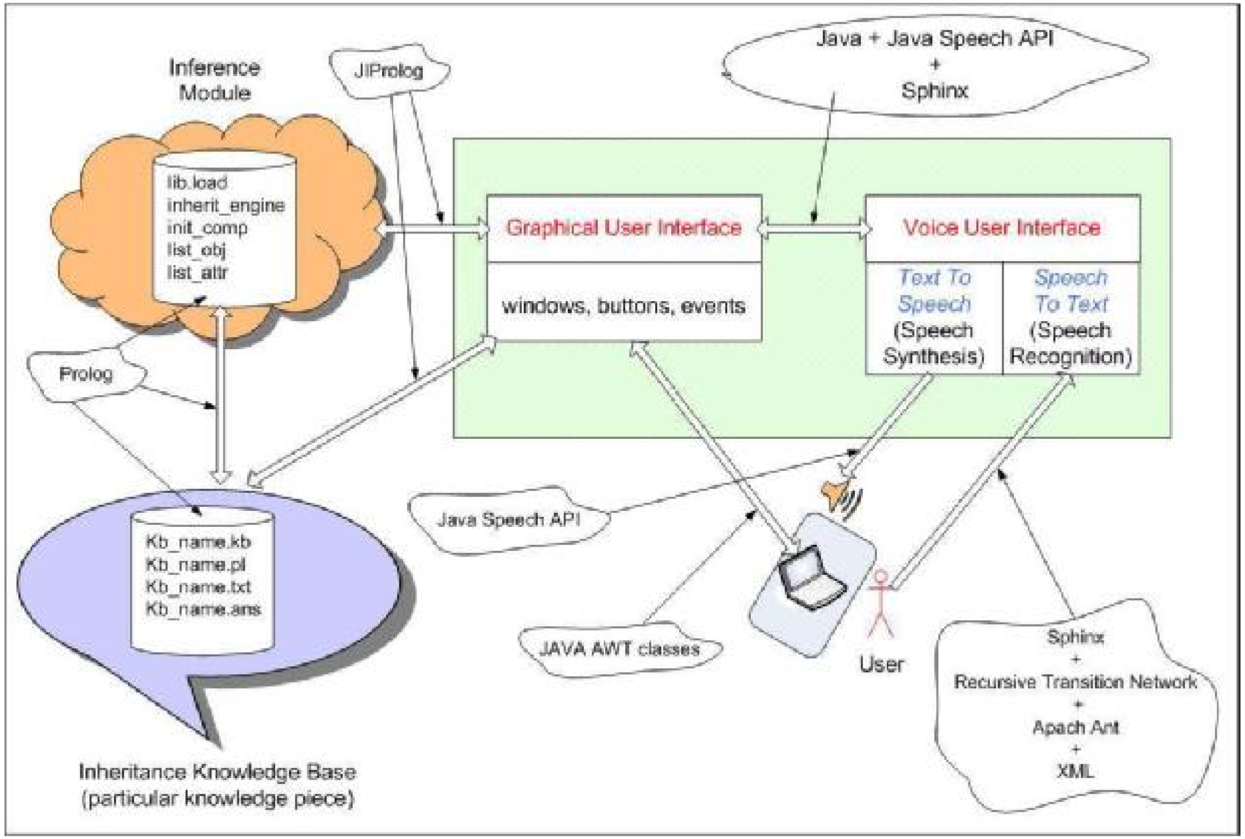


Fig. 3. Architecture of the system

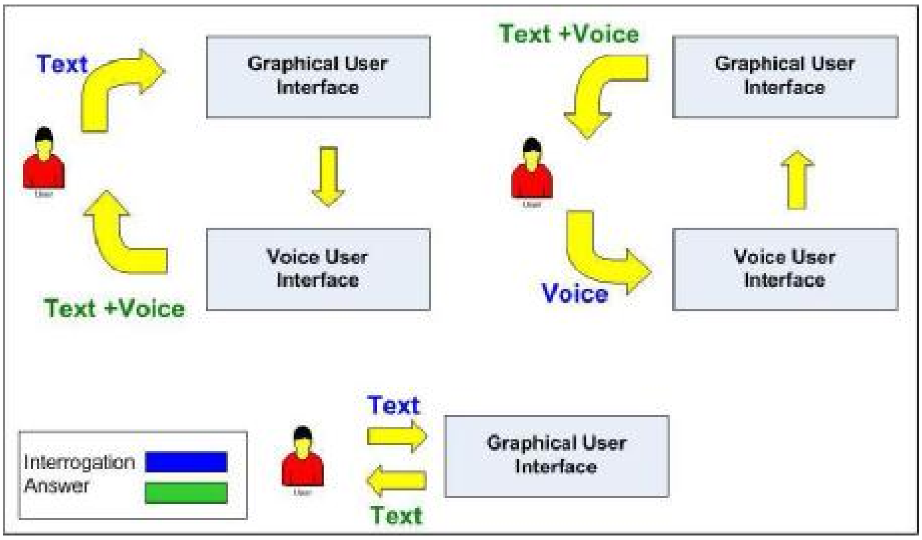


Fig. 4. Manners of communications

The system proposed in this paper includes two interfaces (graphical one and a voice interface), an inference engine to compute the answer mapping and a collection of knowledge bases. The information flow between these entities is presented in Figure 5.

3.2 Grammar for spoken sentences

The syntax of the interrogation sentence is specified by means of a Recursive Transition Network (RTN). A RTN is a structure organized on several levels. There is a main level giving a meta-structure of the sentence. Each other level recognizes a part of this sentence and the transition from a level to other level is specified by the word PUSH. The main level has an initial state and a final state. A sentence S is accepted by the grammar if the final state is reached when the last entity of S is encountered. An example of RTN is given in Figure 6, where two accepted sentences are also specified. The sequence of the transient states is presented in this figure for each sentence.

This grammar depends on the knowledge base specified in the module SM. In order to exemplify the grammar we consider the following knowledge piece KP:

> *In a competition organized by some association, every candidate obtains the scores* $s1$, $s2$ *and* $s3$. *Consequently a general score* $s1 + s2 + s3$ *is obtained for each candidate. Peter, Elvis, Maria and Susan participate to this competition. Peter likes to play tennis and also likes to drink a cup of*

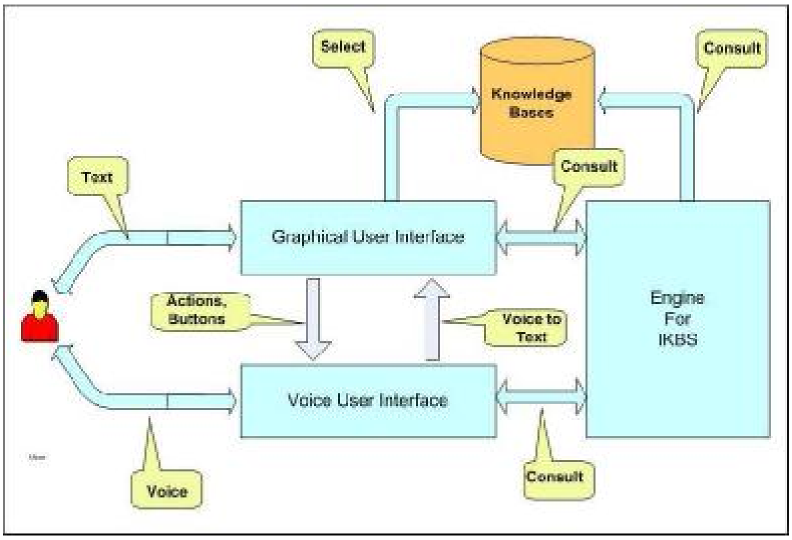
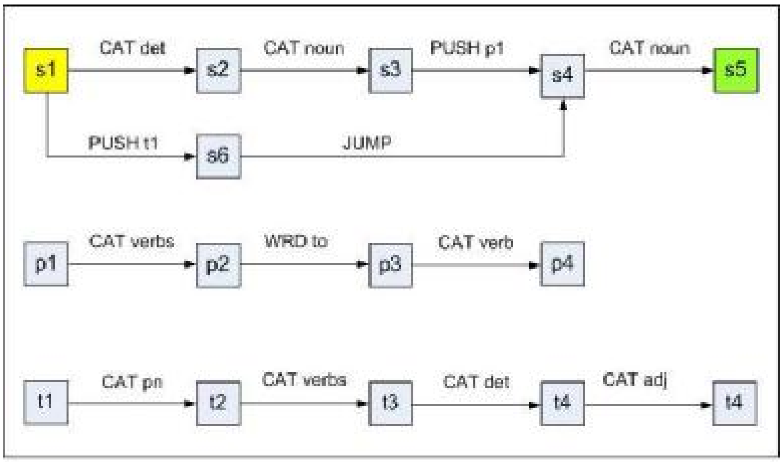


Fig. 5. The flow of information



The boy learns to play tennis. (s1,s2,p1,p2,p3,p4,s5)

Peter eats a red apple. (s1,t1,t2,t3,t4,t5,s6,s4,s5)

Fig. 6. An example of RTN

> *coffee. He has obtained the scores $s1 = 8$, $s2 = 9$ and $s3 = 7$. Elvis has obtained the scores $s1 = 9$ and $s2 = 5$. The score $s3$ is unknown for Elvis because he did not participate to the last test. Elvis was born in 1987. He likes to drink a cup of milk. Susan has obtained the scores $s1 = 9$, $s2 = 8$ and $s3 = 5$. She is a student. Every student likes to edit a text using a computer. The age of Susan is 20. Susan likes to drink a cup of tee. Maria obtained the same scores as Susan, she likes to edit a text using a computer also and she is 5 years younger than Susan.*

The communication grammar is given in Figure 7. According to this grammar we can exemplify the following sentences for interrogation:

- How old is Peter?
- What does Susan like?
- Say me the born year of Maria.
- Find the score of Elvis.

Remark 1. As we can observe such a grammar is not a context free one in the Chomsky classification.

Remark 2. As a programming consideration we shall remark that the description of the grammar specified in this section is given in the file *interrogate.gram* whose content is the following:

```
#JSGF V1.0;

grammar interrogate;

public <proposition> = <prop1> | <prop2> | how old is <pn>;

<prop1> = <s1> <pn> <verb>;
<prop2> = <s2> <s3>;
 <s1> = what does;
 <s2> = (say me) | find;
 <s3> = the <s4> <prep> <pn>;
 <s4> = score | (born year) | age;
 <pn> = (Peter | Susan | Elvis | Maria);
 <prep> = of | for;
 <verb> = like | drink;
```

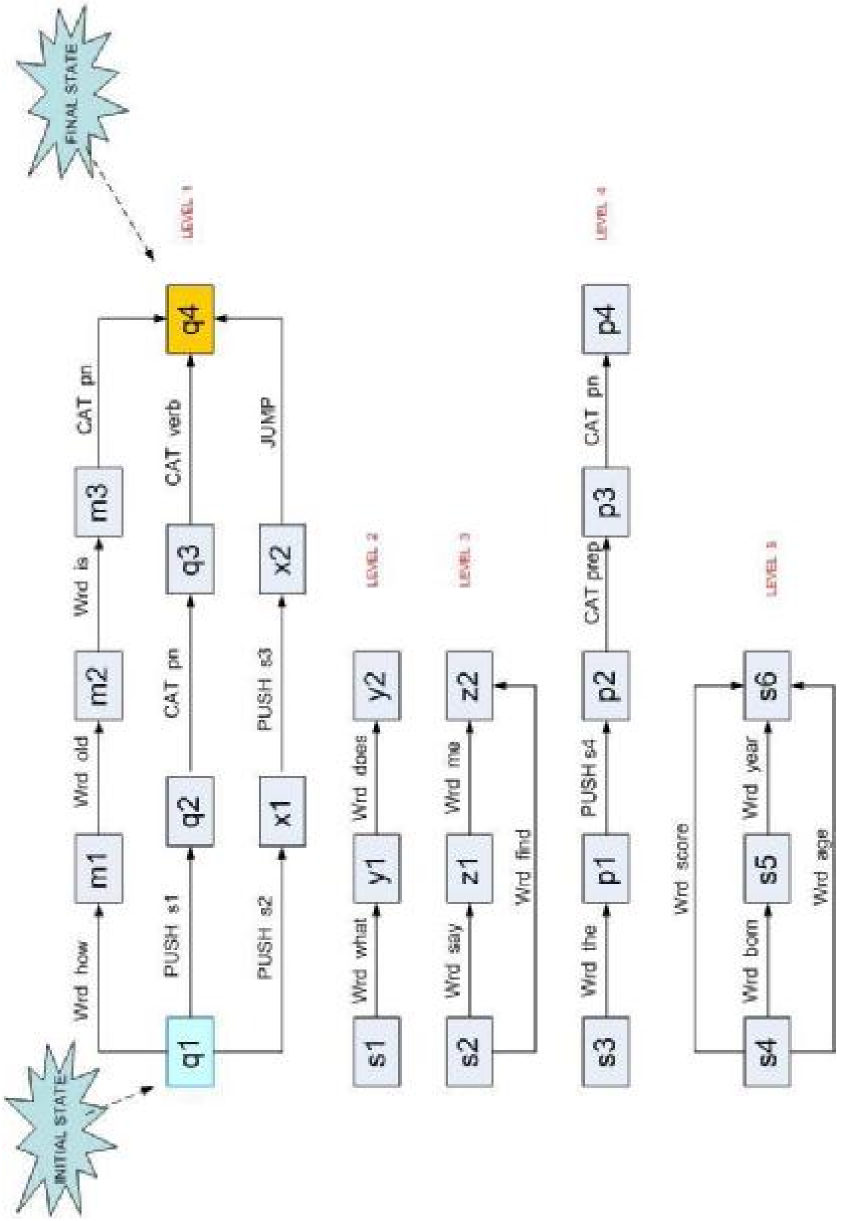


Fig. 7. RTN for KP

4 Running the application

If we run the application then the GUI represented in Figure 8 appears at the initial step. In this figure we explained the tasks of the buttons and the use of the windows.

We observe the GUI includes two windows. The first window is used to display the text of the answer if we interrogate the system. This text is displayed even if we ask the system to give the answer by voice. This requirement is explained by the fact that a spoken sentence is not persistent. The second window is used to display several auxiliary messages of the execution: loading a knowledge base, initializing of the computation for the answer mapping, the invitation to say a sentence by voice, the entities extracted from the spoken sentence, the value of the answer mapping etc.

The actions performed by pressing the buttons are described as follows:

- **Start-processing**: select the name of a knowledge base;
- **Load_KB**: load the selected knowledge base;
- **Load_objects**: extract the objects of the knowledge base and introduce these objects into the first pop-up menu (a Choice object) of GUI;
- **Load_attributes**: extract the attributes of the knowledge base and introduce these names into the second pop-up menu (a Choice object) of GUI;
- **Init_Comp**: perform several computations requested by the answer mapping;
- **Load_library**: load two Prolog libraries requested by the inference engine of the inheritance based knowledge system;
- **CompK**: compute the value of the answer mapping for some object and attribute extracted from the interrogation sentence or selected from the two pop-up menus;
- **Answer:Text & Voice**: write the text of the answer and pronounce the associate sentence;
- **Hello by Voice**: the system says an introductory text of kind "Hello user";
- **Voice_interrogation**: the user specifies the fact that the interrogation is given by voice;
- **Content of KB**: the user can visualize the knowledge base loaded by the button Load_KB;
- **Finish**: close all actions and finish the consultation.

We specify now some examples of consultation and the results of this consultation can be viewed in Figure 9:

- **Voice interrogation**: How old is Peter?
- **Answer by voice and text**: Peter is 27 years old.

- **Voice interrogation**: What does Susan like?
- **Answer by voice and text**: Susan likes to edit a text using a computer.

- **Voice interrogation**: Say me the born year of Maria.
- **Answer by voice and text**: Maria was born in 1992.

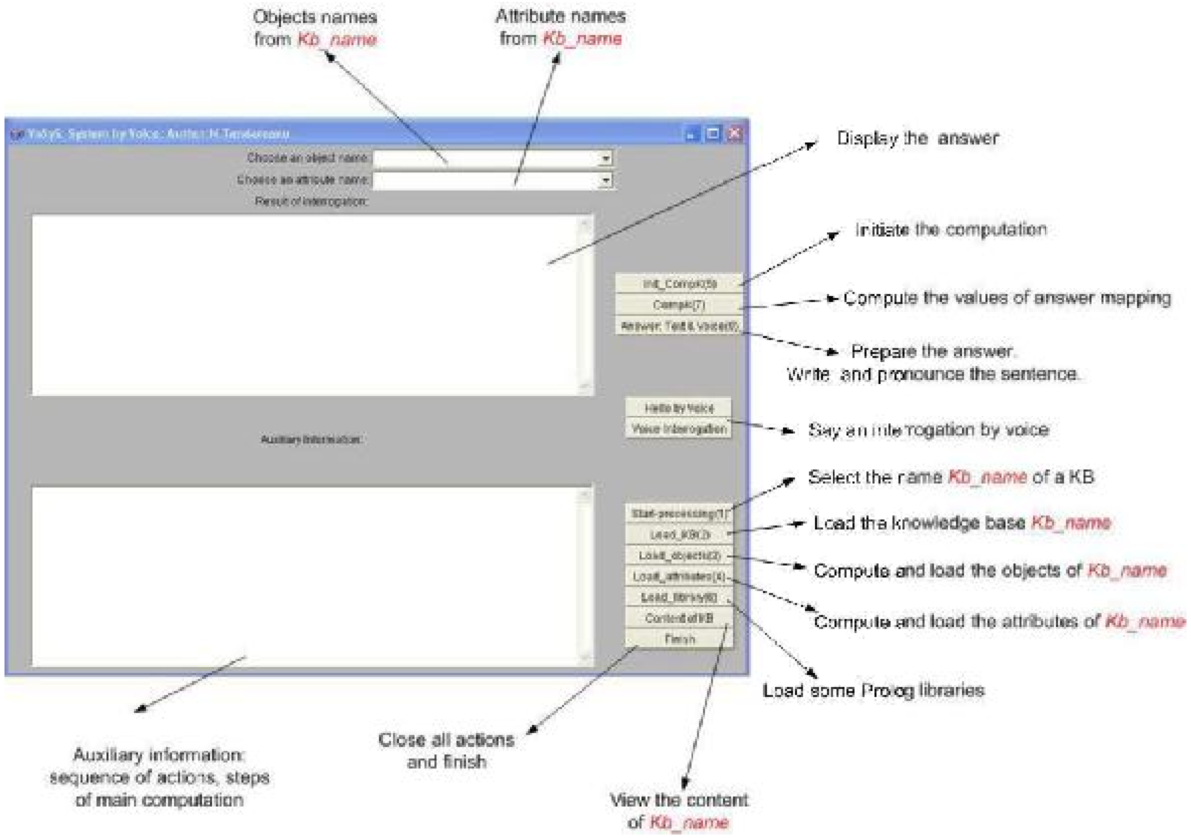
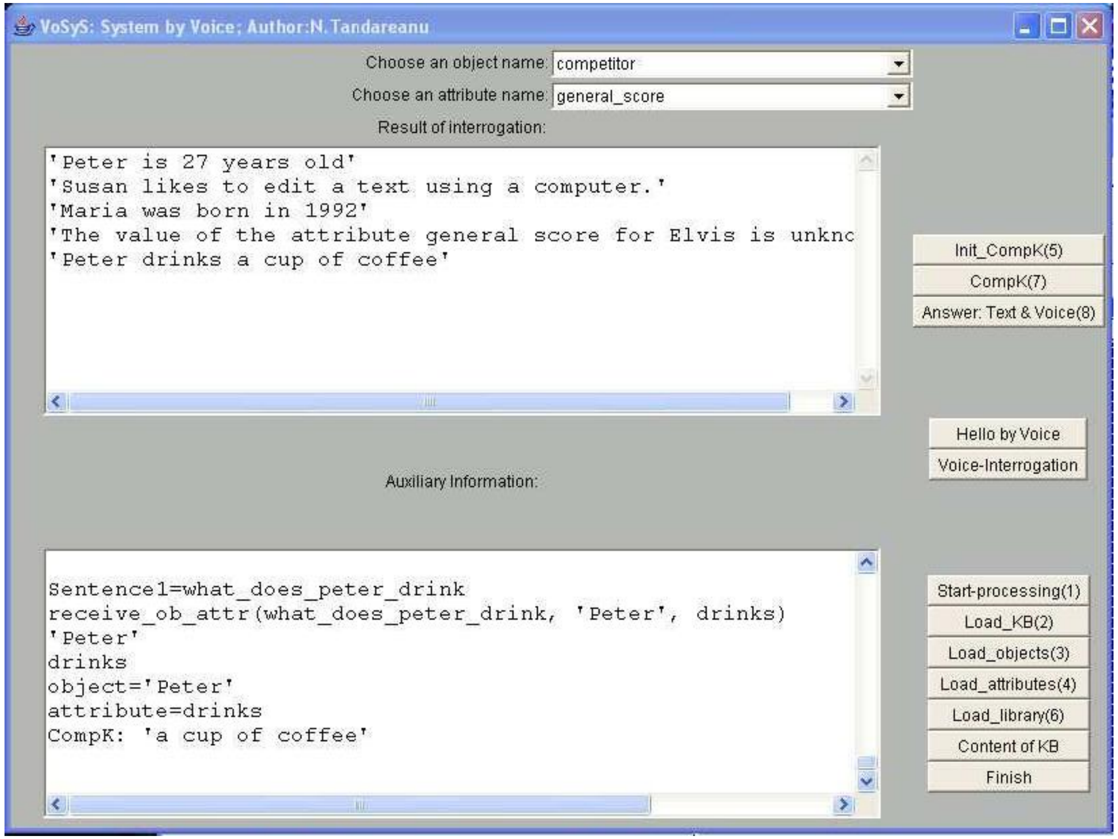

YaSyS_System by Voice : Author: N.Tandareanu
Choose an object name:
Choose an attribute name:
Result of interrogation:
Auxiliary information:
Init_Comp(5)
Comp(7)
Answer: Text & Voice(8)
Hello by Voice
Voice interrogation
Start-processing(1)
Load_KB(2)
Load_objects(3)
Load_attributes(4)
Load_library(6)
Content of KB
Finish
Objects names from Kb_name
Attribute names from Kb_name
Display the answer
Initiate the computation
Compute the values of answer mapping
Prepare the answer.
Write and pronounce the sentence.
Say an interrogation by voice
Select the name Kb_name of a KB
Load the knowledge base Kb_name
Compute and load the objects of Kb_name
Compute and load the attributes of Kb_name
Load some Prolog libraries
Auxiliary information: sequence of actions, steps of main computation
Close all actions and finish
View the content of Kb_name

Fig. 8. GUI of the system: first step



Choose an object name: competitor

Choose an attribute name: general_score

Result of interrogation:

```
'Peter is 27 years old'
'Susan likes to edit a text using a computer.'
'Maria was born in 1992'
'The value of the attribute general score for Elvis is unknc
'Peter drinks a cup of coffee'
```

Auxiliary Information:

```
Sentence1=what_does_peter_drink
receive_ob_attr(what_does_peter_drink, 'Peter', drinks)
'Peter'
drinks
object='Peter'
attribute=drinks
CompK: 'a cup of coffee'
```

Fig. 9. Results of the interrogation

- **Voice interrogation**: Find the score of Elvis.
- **Answer by voice and text**: The value of the attribute general score for Elvis is unknown.

- **Voice interrogation**: How does Peter drink?
- **Answer by voice and text**: Peter drinks a cup of coffee.

5 Conclusions

In this paper we finalized a research work concerning the knowledge modeling based on inheritance. This work included the following aspects:

- The definition of a mathematical model for knowledge systems based on inheritance and the study of the computability properties for the answer mapping of such a system ([4], [5], [6]).
- The implementation of the inference engine to compute the answer mapping and the communication by voice of the answer ([8]).
- The implementation by voice of the interrogation for such a system, that was described in this paper.

As a result of this research we obtained a system of communication by voice to interrogate and answer for an inheritance based knowledge system. The implementation was accomplished using Java Technology and a bidirectional connection Java-Prolog ([10]). A similar study can be obtained for knowledge systems based on other methods of knowledge representation such as semantic schemas. An interesting developing of this research refers to the introduction of the *Voice Recognition* technology to recognize the owner of the software or to use this technology to allow the interrogation of some kink only for some person.

References

[1] **L.Bolc** (Ed.):- The Design of Interpreters, Compilers,and Editors for Augmented Transition Networks, Springer-Verlag, Symbolic Computation, 1983

[2] **E.C.Koenig**:- Knowledge Structures for Communications in Human-Computer Systems, General Automata-Based, Wiley-Interscience, John-Wiley & Sons, 2007

[3] **Marta Gatius Vila**: - Using an Ontology for Guiding Natural Language Interaction with Knowledge Based Systems, Ph.D. Barcelona, University of Catalunya, 2000

[4] **N. Ţăndăreanu**: - Lattices of Labelled Ordered Trees (I), Annals of the University of Craiova, Mathematics and Computer Science Series, Vol. XXVIII, 2001, 29-39

[5] **N. Ţăndăreanu**: - Lattices of Labelled Ordered Trees (II), Annals of the University of Craiova, Mathematics and Computer Science Series, Vol. XXIX, 2002, 137-144

[6] **N. Ţăndăreanu**: - Inheritance-based knowledge systems and their answer functions computation using lattice theory, Romanian Journal of Information Science and Technology, Vol.6, Numbers 1-2, 2003, 227-248

[7] **N. Ţăndăreanu**: - Transfer of knowledge via semantic schemas, 9th World Multi-Conference on Systemics, Cybernetics and Informatics, July 10-13, Vol. IV, 2005, 70-75

[8] **N. Ţăndăreanu**: - VoSyS: A System by Voice to Answer in Inheritance-Based Knowledge Systems, 6^{th} International Conference on Artificial Intelligence and Digital Communications, Thessaloniki, Greece, Vol.106, August 2006, p.91-101

[9] **N. Ţăndăreanu**: - Master-Slave Systems of Semantic Schemas and Applications, International Conference on Intelligent Systems and Control (ISC 2007), November 19-21, 2007, Cambridge, USA (to appear)

[10] **Ugo Chirico**:- JIPrologRefManual.pdf, http://www.ugosweb.com/jiprolog