Ontology-driven VoiceXML Dialogues Generation

Marta Gatius, Meritxell González
TALP Research Center, Universitat Politècnica de Catalunya
Campus Nord, C6. Jordi Girona, 1-3, 08034 Barcelona, Spain
{gatius,mgonzalez}@lsi.upc.es

Abstract: This paper describes a proposal to improve the communication process and the engineering features in VoiceXML dialogues. This proposal is focused on the efficient and reusable representation of the knowledge involved in communication. In order to achieve a good dialogue design we propose the explicit representation of the conceptual information the application needs form the user in an ontology. A syntactic-semantic taxonomy is provided to facilitate the generation of the grammars and the system's messages. Using this separate, reusable knowledge organization, VoiceXML dialogues for different applications and in different languages can be generated semi-automatically.

1 Introduction

The widespread of the VoiceXML standard ([2]) has brought a radical change in speech-based commercial applications. VoiceXML, based on XML language, allows a rapid deployment of spoken dialogue systems to Internet-data. However, the capabilities of the VoiceXML systems are often limited. Although most of the efficient problems of VoiceXML systems are originated by the limitations in automatic speech recognition engines, some of them are also caused by a poor dialogue design. Because of the lack of skilled professionals, many of these systems are developed without a careful design of the different types of knowledge involved in communication: conceptual (application tasks and concepts) and linguistic.

In order to enhance the capabilities of the VoiceXML services, different tools and methodologies for helping designers have been provided (such as those developed by Loquendo [1]). Additionally, different groups are carrying out investigations on the opportunities for the integration of intelligent technologies with VoiceXML. In [Wi02] several improvements by applying Natural Language (NL) techniques in VoiceXML applications are suggested, being one of them the generation of system messages and prompts. There are several dialogue systems combining NL generation techniques and VoiceXML, such as the Meteorological Information System described in [VCH02].

In this paper, we study how the communication process and the engineering features in VoiceXML dialogues can be improved by using an ontology representing the application knowledge. For this purpose, we have followed the proposal described in [GR02] using ontologies for generating domain-restricted grammars for NL-guided dialogues.

In our proposal, the conceptual and linguistic knowledge involved in communication is represented in separate and reusable bases. The application concepts are represented in an ontology. These concepts are described by a set of attributes. During the communication, the value of these attributes are asked (or given) to the user. The VoiceXML services are generated from the application knowledge represented in the ontology. In order to facilitate the automatic generation of the sentences asking and giving information about the attribute values, all conceptual attributes are classified according to a syntactic-semantic taxonomy.

A VoiceXML service consists of a sequence of interactions between a user and an implementation platform. A VoiceXML service consists of a set of documents. It can be seen as a conversational finite state machine. Each document describes one or more specific interactions or dialogues with the user. Each interaction leads the user to the following interaction. Basically, designing a VoiceXML service consists of the following three steps: Designing the logical structure of the system/user interaction, designing the system's messages and defining the allowed user's messages. Next sections describe the tasks to be done in each of these steps and how these tasks are simplified in our proposal. Last section draws some conclusions and future work.

2 Designing the logical structure of the system/user interaction

The logical structural design defines the information the application needs from the user, the information the user needs from the application and how this information is delivered (i.e., the order in which the information is asked and given to the user). This step covers the design of all VoiceXML documents: the sequence of dialogues, the help to be provided to the user and the error recovering policies.

The correct dialogue design can help minimize the number of user/system interactions, reducing the chance recognition errors will occur. For this purpose, both the task model as well as the user's knowledge of the task have to be studied before designing the dialogue. Information about the user's task model as well as information about the user's expressions is often obtained from samples of human dialogues previously recorded. From this information, a task-oriented system-driven dialogue design can be developed. Well organized dialogues improve efficiency in communication because they help users to understand the information the application needs from them.

In our proposal, the logical structure of the system/user interaction is described in an ontology, where all concepts involved in the communication have to be represented. These concepts are described by attributes. Basically, dialogues consist of asking/giving the user values of these conceptual attributes. VoiceXML dialogues are generated automatically from the concepts in the ontology. Different types of dialogues can be generated for an application (system-driven dialogues or mixed-initiative dialogues). Simple system-driven dialogues are obtained by generating VoiceXML documents from the attribute describing the ontology concepts. Each VoiceXML document contains an interaction asking the user the value of a specific attribute. The order of the sequence of dialogues is obtained from the preconditions associated with the concepts and the attributes.

We exemplify this proposal by describing our work in a dialogue system for collecting objects from houses. Basically, this application allows the user to ask for information about the collection service, to fix a date for collection or to cancel a previous request. These application services are represented in the ontology in four conceptual classes: COLLECTIONSERVICE and its three subclasses, INFORMATIONSERVICE, REQUESTSERVICE and CANCELLATIONSERVICE. The concept COLLECTIONSERVICE is described by the attribute *servicetype*. The possible values of this attribute are *information*, *request* and *cancellation*. In order to fix a date for a collection, the application needs the user address and the type of object that has to be collected (i.e., *a table*, *a refrigerator*). For this reason, the REQUESTSERVICE concept is described by the attributes *address* and *object*. The different types of objects that can be collected from a house are represented in a separated taxonomy.

Figure 1 shows the program used to generate VoiceXML documents. For each conceptual attribute this program generates a VoiceXML document asking the user the attribute value, if this document does not already exist. VoiceXML documents for attributes appearing in several applications (i.e. *address*, *telephone number* and *name*) can be reused. To create a new document for a specific attribute, the program uses a general document containing the VoiceXML code for asking the user the value of a conceptual attribute.

```
for each CONCEPT in APPLICATION_ONTOLOGY do
   for each ATTRIBUTE in CONCEPT do
      if not ExistVXMLDoc(ATTRIBUTE) then
         GenerateVXMLDoc(CONCEPT, ATTRIBUTE)
      endif
   endfor
endfor
```

Figure 1: The procedure to generate VoiceXML documents

Figure 2 shows one of the general documents that have been used for generating a VoiceXML dialogue. In the document generated, the word "*ATTRNAME"* is substituted by the name of the attribute, the word "*VAL1ATTRNAME*" by the first possible value of the attribute and the word "*VAL2ATTRNAME*" by the second value. In case the attribute has more than two values the corresponding conditions would be added to the *if* structure. For example, from the attribute *servicetype*, describing the concept COLLECTIONSERVICE, a document asking the user what type of service he wants is generated. This document indicates that if the user's answer is *information* the next document is *informationservicetype.vxml*, in case the response is *request* next document is *requestservicetype.vxml* and if the answer is *cancellation* next is *cancellationservicetype.vxml*.

Our proposal implies the study of the task model as well as the user behaviour. It also implies the explicit representation of this knowledge in the ontology. For skilled developers the dialogues, grammars and prompts generated from this representation can be a starting point to develop more complex dialogues.

Inferential features of ontologies can also be used to improve communication. The dialogues generated can deal with a user response not matching the value expected (one of the possible values of a specific attribute) but matching its hyperonym. In this case, a clarification dialogue asking the user for more specific information will be activated. For example, when asking the user the type of object he wants to throw out, if the user's answer is *furniture* the system will recognize it, even when the answer expected is a particular type of furniture (i.e., *table*). Next, the system will ask the user to specify the type of furniture he wants to throw out.

```
<?xml version="1.0"?>
<vxml version="2.0" xml:lang="es-es">
  <form id="formATTRNAME">
    <field name="attrATTRNAME">
     <grammar src="file://localhost/grammars/gramATTRNAME.sjv" type="application/x-jsgf-flx"/>
            <!-- PromptAskATTRNAME-->
             <prompt count = 1>
                          PROMPTATTRNAME1
             </prompt>
             <prompt count = 2>
                          PROMPTATTRNAME2
             </prompt>
             <filled>
                <if cond="attrATTRNAME=='VAL1ATTRNAME'" >
                   <assign name="varATTRNAME" expr="'VAL1ATTRNAME'"/>
                          <goto next="VAL1ATTRNAME.vxml"/>
                 <elseif cond="attrATTRNAME=='VAL2ATTRNAME'"/>
                   <assign name="varATTRNAME" expr="'VAL2ATTRNAME'"/>
                          <goto next="VAL1ATTRNAME.vxml"/>
                </if>
              </filled>
      </field>
    </form>
</vxml>
```

Figure 2: A document containing the VoiceXML code for asking the attribute value

3 Designing system's messages and prompts

The system's interventions in dialogues basically prompt the user to introduce the correct input, ask the user data confirmation and give application information to the user. In our proposal, the developer does not need to write any prompt. Instead of that, the designer only has to classify the attributes describing the concepts according to the syntactic-semantic taxonomy provided. The basic classes of this taxonomy are associated with the different grammar roles appearing in human-machine conversations.

The syntactic-semantic classification of the attributes allows a variety of different linguistic coverage for each attribute class. Each attribute class is associated with a set of general patterns. These patterns represent different interrogative and declarative sentences for asking (or giving) the attribute value. There is a pattern for asking the attribute value in a more general form, another for asking this information in a more focused manner, a pattern for confirming the value of the attribute, etc. To adapt the general patterns to each specific attribute, the lexical entries associated with the concept, the attribute and its values must be provided. A lexical entry can be a word or a phrasal noun.

Consider, for example, the concept COLLECTIONSERVICE described in Section 2. The attribute describing this concept, *servicetype*, belongs to the syntactic-semantic class OFTYPE. The general pattern associated with this class is "*The <conceptnoun><setof_and_attributevalueverb>. What do you want*?" where *setof_and_attributevalueverb* indicates that the values of the attributes are expressed by a set of verbs connected by the preposition *and*. In this example, the lexical entry corresponding to the concept is "*Object Collection Service*" and the verbal groups associated with the attribute values are "*to give information*", "*to fix a data for collection*" and "*to cancel collection*". The prompt resulting when adapting the general pattern would be "*The Object Collection Service gives information, fixes a data for collection and cancels collection. What do you want?* "

The second and most focused pattern associated with attributes belonging to the class OFTYPE is "*Say what you want: <setoff_or_attributevaluenoun>*". Because the nouns associated with the values of the attribute *servicetype* are "*information*", "*collection*" and "*cancellation*", the resulting prompt would be "*Say what you want: information, collection or cancellation*".

4 Defining allowed user's messages

The information allowed from the user can be represented as grammars or option lists (menus) that would be presented to the user. Defining a grammar consists of writing the grammar rules covering all possible words that users are expected to say when prompted by the system. Semantic information can also be associated with the grammars. The incorporation of semantic information in the grammars facilitates the process of obtaining the precise information needed by the application from the user's sentence. Usually, a grammar covers only a dialogue turn, although more general grammars can also be defined for managing help messages and misunderstanding recovering.

In our proposal, grammars are generated automatically from the attribute values. The words in the grammar rules are obtained from the lexical entries associated with the values of the attributes. The semantic information associated with each grammar rule corresponds to those attribute values.

One of main strengths of our proposal is that the syntactic-semantic taxonomy is easily portable to different languages. Although the attribute classification is based on Spanish linguistic distinctions, most of them can be reusable across several languages. Following our proposal, VoiceXML dialogues in several languages can be generated from a single conceptual representation.

Conclusion and future work

We are studying how the communication process and the engineering features in VoiceXML dialogues can be improved by representing appropriately the different types of knowledge involved in communication. In order to achieve a good dialogue design we propose the explicit representation of the application knowledge appearing in dialogues in an ontology. In order to facilitate the generation of system prompts and grammars needed in each interaction, we propose the use of a syntactic-semantic taxonomy. This taxonomy relates the attributes describing the concepts to the linguistic expressions necessary for asking (giving) the user the value of these attributes. The representation we propose allows the automatic generation of VoiceXML dialogues for different languages. The dialogues generated have to be tested and refined manually.

We have followed this methodology for generating simple VoiceXML documents containing only interactions asking the user for a specific attribute value. Currently, we are improving a prototype for collecting objects. We want to study how clarification and error recovering policies have to be incorporated when automatically generating the documents. Future work will also include adapting this proposal to more complex applications. In those applications, we want to study the general patterns used to generate system's messages, how they can be adapted to different situations (casual or habitual users, formal or informal style), and how the information dynamically obtained from the application can be incorporated into the system messages and the grammars.

References

[Da96] Danieli, M. On the use of expectations for detecting and repairing human-machine miscommunication. AAAI-96, Workshop Detecting, Repairing, and Preventing Human-Machine Miscommunications at the Thirteenth National Conference on Artificial Intelligence, Portland, USA.

[GR02] Gatius, M. and Rodríguez, H. Natural Language Guided Dialogues for Accessing the Web. 5th International Conference, Text, Speech and Dialogue 2002, Brno, Czech Republic.

[VCH02] Villarejo, L., Castell, N. and Hernando, J. Dialogue Management in an Automatic Meteorological Information System. International Conference IEA/AIE 2003 (Industrial and Engineering Applications on Artificial Intelligence and Expert Systems). June, 23-26, 2003, LNAI vol. 2718, pp. 495-504. Loughborough, England.

[Wi02] Winiwarter, W. Recent Developments in Human Language Technology. R. R. Wagner (ed). A Min Tjoa @ Work, books@ocg.at, Vol. 164, Vienna, OCG, 2002.

1. Loquendo. http://www.loquendocafe.com/index.asp
2. [W3C]. Voice Extensible Markup Language. Version 2.0 http://www.w3.org/TR/2004