

# A Dialogue System for Robots using VoiceXML

**Louise Funke**

Department of Computer Science  
F03, Institute of Technology  
Lund University, Sweden  
f031f@student.lth.se

**Marc Bauer**

Department of Computational Linguistics  
Friedrich-Alexander University  
Erlangen-Nuremberg, Germany  
mcbauer@linguistik.uni-erlangen.de

## Abstract

This paper describes a dialogue system, based on speech, for interaction between robots and humans. The dialogue system is based on a form for a wood sign process. The dialogue is implemented in VXML. A file containing all variables is created for further processing. The paper describes a project done during the course "Language Processing and Computational Linguistics" in autumn term 2007 at LTH.

## 1 Introduction

Robots in larger industries are often programmed to only do one thing, but smaller firms don't have the capital to invest in several robots, so a robot able to perform different assignments would be of great importance for these enterprises.

The European Union has realized this and taken an initiative to create a project, the SME robot, which is a cooperation between several enterprises, e.g. ABB, and universities, e.g. Lund University.

The robot at LTH (see images in Appendix 6.1) can for example solve sudokus and mill text and figures onto wooden plates. Our project is based on the latter process, called the wood sign process.

A speech-controlled robot could be of great use in the industry, as you don't need direct access to a computer. Our task is to implement a speech-operated dialogue system for interaction between robots and humans, based on the form used in the SME-robot-project for the wood sign process.

## 2 Dialogue Systems

A dialogue system is a system for interaction human-machine. The dialogue system may be based on text, speech, graphics or gestures.

In this case, the dialogue system is based on speech and interaction between human and robot. The robot is actually, in this early phase, a computer. In the future the speech dialogue system could be transferred to the robot itself.

The dialogue system for this human-robot interaction is based on a form for the wood sign process (see image in Appendix 6.2). Since we start out from a form, we have a limited number of questions and for each question there are some options, so the possible answers are also known. The answers are to be saved in a .txt-file for later processing.

### 2.1 Example dialogue

A dialogue may look as follow:

**Robot:** Welcome to the SME robot voice operating system. Please choose one of the following options. Wood sign process, startover or quit.

**Human:** (I'd like) wood sign process (please).

**Robot:** You have chosen the wood sign process. Do you want to continue? Say yes to continue or no to go back to the menu.

**Human:** Yes.

**Robot:** Step one, configuration of wood sign variables. What kind of shape do you want? Sharp corners, soft corners or cut corners.

**Human:** Soft corners (please).

**Robot:** You have chosen soft corners. Is that correct?

**Human:** Yes.

**Robot:** Please give the corner diameter in millimetres for the soft corners.

**Human:** 40.

**Robot:** You have chosen 40 millimetres. Is that correct?

**Human:** Yes.

**Robot:** Hole configuration. Please give number of holes.

**Human:** 2.

**Robot:** You have chosen 2 holes. Is that correct?

**Human:** Yes.

**Robot:** Which pattern should be used?

**Human:** 1.

**Robot:** You have chosen pattern number 1. Is that correct?

**Human:** Yes.

**Robot:** Data collected. Summarizing. You have chosen soft corners with a 40 millimetres radius, 2 holes and pattern number 1. Is that correct?

**Human:** Yes.

### 3 VoiceXML

One system to realize the dialogue is the usage of VoiceXML. VoiceXML is the world wide web consortium's (W3C) standard extensible markup language (XML) for voice dialogues. (Wikipedia VoiceXML, 2007).

#### 3.1 History and features of VoiceXML

The first version of VoiceXML was developed by AT&T, IBM, Lucent and Motorola who teamed up to form one standard instead of the various ones they were working on (W3C VoiceXML 2.0, 2004). It was later handed on to the W3C who developed the 2.0 and 2.1 version used in our program.

VoiceXML is made to create audio dialogues by using speech synthesis, recorded audio, speech recognition and DTMF input and is capable of recording audio input, transferring data and even mixed initiative (2004). Just like with HTML files and a web browser, all this happens through a voice browser that interprets the VoiceXML files. This voice browser has the needed speech recognition software foundation already installed so there is no need to program it for each task.

#### 3.2 Structure of a VoiceXML document

The top level of each VoiceXML document is <VXML> followed by a kind of dialogue, whether a <MENU> or a <FORM>. Menus are, as the term self describes, nodes of choices and forms give and might expect information. If input is expected there is a <FIELD> needed. Before continuing, the field must be filled with a value of various predefined types (e.g. string, number, boolean). Further important commands are if-else-elseif blocks, defined grammars and subdialogues.

### 4 Implementation and Structure

The project's program was finally developed and tested in Tellme Studio®. Tellme, a Microsoft® subsidiary, offers free usage of their voice browser.

At the moment the client has to connect via Skype™ or telephone to the Tellme Studio® voice browser which interprets the VoiceXML files.

The program is structured in the following parts and also files.

- The menu, that guides to the main program and later in the future to other parts.
- The main program with questions, confirmations to answers, a summary and a transfer of data.
- The CGI script that receives the data and stores it to a text file.

The source code to those files are found in the appendix.

### 5 Conclusions

Working with VoiceXML and Tellme Studio showed to have many positive features. VoiceXML is very easy to program. Whoever has knowledge how to program a website with HTML should feel very comfortable with VoiceXML, both being a derivate of XML. It is simple structured, readable and has a steep learning curve. It is even extendable with other known "web-languages" as JavaScript and Perl. And due to the fact that the voice interpreting software is already well preprogrammed it is incredible advanced. The software listens to

keywords, so that the user can effectively talk in whole sentences with the system.

But the system also has some flaws. In the current state the system was very affected by surrounding noise. Using the program in its momentary state in a working environment is impossible. Effective use was only achieved by total silence and one single person speaking, the user. There are howsoever commands and tricks to reduce this affection. Another point is the missing flexibility. The program has a strict line of commands. If something was not anticipated by the programmer the program itself will either crash or repeat the same question over and over again. Each working step is predefined by the programmer and has to be changed by the programmer if the environment or task changes.

## Acknowledgements

We would like to thank Pierre Nugues and Mathias Haage, dept. of computer science, LTH, for their support and comments during this project. We would also like to thank the SME-robot-team at LTH.

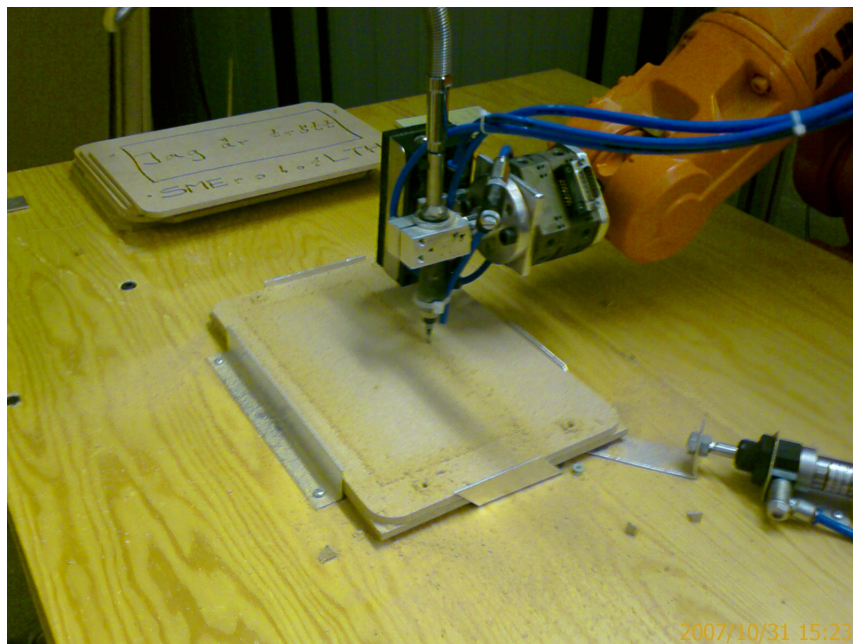
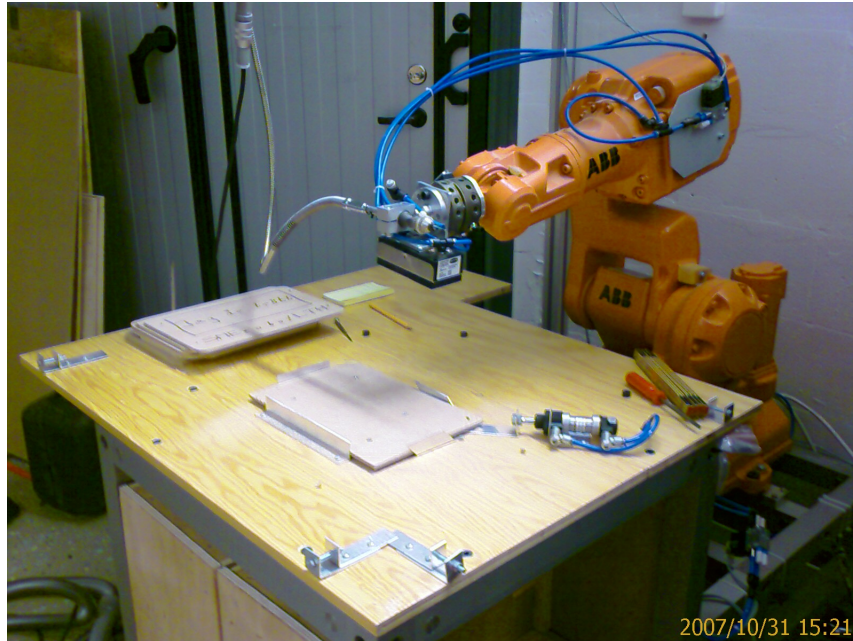
## References

- Wikipedia.org, various authors. 2007. *VoiceXML*, <http://en.wikipedia.org/wiki/VoiceXML> Date: 2007/12/17 at 21:00
- W3C, Scott McGlashan et al.. 2004. *Voice Extensible Markup Language (VoiceXML) Version 2.0*, <http://www.w3.org/TR/2004/REC-voicexml20-20040316/> Date: 2007/12/17 at 21:00

# Appendix

## 6 Pictures

### 6.1 SME robot







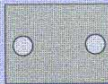
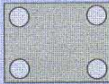


## 6.2 Input form


### Digitalt pappersformulär för anpassning och tillverkning av träskylt

*jayway*



Kontur		
<input type="checkbox"/> Raka hörn		
<input checked="" type="checkbox"/> Rundade hörn		
<input type="checkbox"/> Snedställda hörn		
	Radie / höjd	
	10 mm <input type="checkbox"/>	
	20 mm <input type="checkbox"/>	
	30 mm <input type="checkbox"/>	
	40 mm <input checked="" type="checkbox"/>	
Hålkonfiguration		
<input type="checkbox"/> 	<input checked="" type="checkbox"/> 	<input type="checkbox"/> 
Mönster och text		
<input type="checkbox"/> Inget mönster	<input type="checkbox"/> Rak kant	<input checked="" type="checkbox"/> Vågig kant
<input checked="" type="checkbox"/> Frihandstext	<input type="checkbox"/> Text OCR	
<div></div>		
Kalibrering arbetsstycke		
<input type="checkbox"/> Utför ej	Visa <input type="checkbox"/>	
<input type="checkbox"/> Manuell	Utför <input type="checkbox"/>	
Konturfräsning		
<input type="checkbox"/> Utför ej	Visa <input type="checkbox"/>	
<input type="checkbox"/> Automatisk	Utför <input type="checkbox"/>	
Hålfrysning		
<input type="checkbox"/> Utför ej	Visa <input type="checkbox"/>	
<input type="checkbox"/> Automatisk	Utför <input type="checkbox"/>	
Mönsterfräsning		
<input type="checkbox"/> Utför ej	Visa <input type="checkbox"/>	
<input type="checkbox"/> Automatisk	Utför <input type="checkbox"/>	
Fritextfräsning		
<input type="checkbox"/> Utför ej	Visa <input type="checkbox"/>	
<input type="checkbox"/> Automatisk	Utför <input type="checkbox"/>	
Utför allt automatiskt i sekvens <input type="checkbox"/>		
Utför allt automatiskt <input checked="" type="checkbox"/>		
Arbetsbeskrivning		
Kasta <input type="checkbox"/>	Klar <input checked="" type="checkbox"/>	Skicka <input checked="" type="checkbox"/>

Start

  
1234 5670

Stop

## 7 Sourcecode

### 7.1 The Menu

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<vxml version="2.0">

<menu id="sme_menu">

<!-- menu prompt -->
<prompt>
    Welcome to the S M E robot voice operating system.
    Please choose one of the following options.
    <break time="200ms"/>
    <enumerate/>
</prompt>

<!-- links -->
<choice next="http://www.linguistik.uni-erlangen.de/~mcbauer/
    vxml/project071210.vxml" >(wood sign process)</choice>
<choice next="http://www.linguistik.uni-erlangen.de/~mcbauer/
    vxml/project_presentation.vxml" >(presentation mode)</choice>
<choice next="http://www.linguistik.uni-erlangen.de/~mcbauer/
    vxml/project_menu.vxml" > or (startover)</choice>

<!-- help and exception catching -->
<help>
    You are in the menu?
    <reprompt/>
</help>
<catch event="noinput nomatch">
    Sorry. I didn't get that.
    <reprompt/>
</catch>
<catch event="noinput nomatch" count="2">
    Sorry. I didn't get that.
    Please say <enumerate/>.
</catch>

</menu>
</vxml>
```

## 7.2 The main program

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<vxml version="2.0">
  <var name="calling_dialog" expr="'none' " />
  <var name="user" expr="'unknown' " />
  <var name="shape" />

  <!--FORM __ confirmation-->
  <form id="main">
    <!-- boolean type field -->
    <field name="yn" type="boolean">
      <prompt>
        You have chosen the wood sign process.
        Do you want to continue?
        Say yes to continue or no to go back to the menu.
      </prompt>
      <!-- help and exception catching -->
      <help><reprompt/></help>
      <catch event="noinput nomatch">
        Sorry. I didn't get that.<reprompt/>
      </catch>
      <catch event="noinput nomatch" count="2">
        Sorry. I didn't get that.
        Please say yes to continue or no to go back to the menu.
      </catch>
    <!-- awaiting yes or no as an answer -->
    <filled>
      <if cond="yn">
        <goto next="#var"/>
      <else/>
        <goto next="http://www.linguistik.uni-erlangen.de/~mcbauer/
          vxml/project_menu.vxml"/>
      </if>
    </filled>
  </field>
</form>

  <!--FORM __ variables-->
  <form id="var">
    <block>
      <assign name="calling_dialog" expr="'main'"/>
      <assign name="user" expr="'noah'"/>
    </block>
```

```

<var name="vari" />
<field name="shape">

<!--shape-->
<prompt>
  Step one, configuration of wood sign variables.
  What kind of shape do you want?
  Sharp corners, soft corners or cut corners.
</prompt>
<!-- grammar for predefined answers -->
<grammar type="application/x-gsl" mode="voice">
  <![CDATA[
  [
    [(sharp corners) (sharp)] {<shape "sharp_corners">}
    [(soft corners) (soft)] {<shape "soft_corners">}
    [(cut corners) (cut)] {<shape "cut_corners">}
  ]
  ]]>
</grammar>

<help><reprompt/></help>
<catch event="noinput nomatch">
  Sorry. I didn't get that.<reprompt/>
</catch>
<catch event="noinput nomatch" count="2">
  Sorry. I didn't get that. Please say sharp corners,
  soft corners or cut corners.
</catch>

</field>
<!-- calling subdialog for confirmation -->
<subdialog src="#confirmation" name="oResult">
<prompt> You have chosen <value expr="shape"/>.
  Is that correct?
</prompt>
<filled>
<if cond="oResult.iconf">
<if cond="'sharp_corners'==shape">
  <goto nextitem="holes"/>
<elseif cond="'soft_corners'==shape"/>
  <goto nextitem="corner"/>
<elseif cond="'cut_corners'==shape"/>
  <goto nextitem="corner"/>
</if>

```



```

<else/>
  <clear namelist="shape"/>
  <clear namelist="oResult"/>
</if>
</filled>
</subdialog>

<!-- holes -->
<field name="holes" type="number">
<prompt>Hole configuration. Please give number of holes.</prompt>
  <help><reprompt/></help>
  <catch event="noinput nomatch">
    Sorry. I didn't get that.<reprompt/>
  </catch>
  <catch event="noinput nomatch" count="2">
    Sorry. I didn't get that.
  </catch>
</field>
<subdialog src="#confirmation" name="oHoles">
<prompt> You have chosen <value expr="holes"/> holes.
  Is that correct?
</prompt>
<filled>
<if cond="oHoles.iconf">
  <goto nextitem="pattern"/>
<else/>
  <clear namelist="holes"/>
  <clear namelist="oHoles"/>
</if>
</filled>
</subdialog>

<!-- pattern -->
<field name="pattern" type="number">
<prompt>Wich pattern should be used?</prompt>
  <help><reprompt/></help>
  <catch event="noinput nomatch">
    Sorry. I didn't get that.<reprompt/>
  </catch>
  <catch event="noinput nomatch" count="2">
    Sorry. I didn't get that.
  </catch>
</field>

```

```

<subdialog src="#confirmation" name="oPattern">
<prompt> You have chosen pattern number <value expr="pattern"/>
    Is that correct?
</prompt>
<filled>
<if cond="oPattern.iconf">
    <goto nextitem="summary"/>
<else/>
    <clear namelist="pattern"/>
    <clear namelist="oPattern"/>
</if>
</filled>
</subdialog>

```

```

<!-- summary of given data
    as boolean for validation
    and transferring the data -->
<field name="summary" type="boolean">
<prompt> Data collected. Summarizing.
<if cond="'soft_corners' == shape">
    <prompt>
    You have chosen <value expr="shape"/>
    with a <value expr="corner"/> millimeter radius,
    <value expr="holes"/> holes and
    pattern number <value expr="pattern"/>.
    </prompt>
<elseif cond="'cut_corners' == shape"/>
    <prompt>
    You have chosen <value expr="shape"/>
    with a <value expr="corner"/> millimeter height,
    <value expr="holes"/> holes and
    pattern number <value expr="pattern"/>
    </prompt>
<else/>
    <prompt>
    You have chosen <value expr="shape"/>
    with <value expr="holes"/> holes and
    pattern number <value expr="pattern"/>
    </prompt>
</if>
Is that correct?
</prompt>
    <help>

```

```

        <reprompt/>
    </help>
    <catch event="noinput nomatch">
        Sorry. I didn't get that.<reprompt/>
    </catch>
    <catch event="noinput nomatch" count="2">
        Sorry. I didn't get that.
        Please say yes to continue or no to go back.
    </catch>
</filled>
<if cond="summary">
<!-- transferring the data -->
<submit
    next="http://alpha2k.ice-server.com/cgi-bin/project_output.cgi"
    method="post" namelist="shape holes pattern"/>
</else/>
    <clear namelist="shape"/>
    <clear namelist="holes"/>
    <clear namelist="pattern"/>
    <clear namelist="oResult"/>
    <clear namelist="oHoles"/>
    <clear namelist="oPattern"/>
</if>
</filled>
</field>

<!--corners-->
<field name="corner" type="number">
<prompt>
Please give the corner diameter in millimeter
for the <value expr="shape"/></prompt>
    <help>
        <reprompt/>
    </help>
    <catch event="noinput nomatch">
        Sorry. I didn't get that.<reprompt/>
    </catch>
    <catch event="noinput nomatch" count="2">
        Sorry. I didn't get that.
    </catch>
</field>
<subdialog src="#confirmation" name="oCorner">
<prompt>
You have chosen <value expr="corner"/> millimeter.

```

```

    Is that correct?
</prompt>
<filled>
<if cond="oCorner.iconf">
    <goto nextitem="holes"/>
<else/>
    <clear namelist="corner"/>
    <clear namelist="oCorner"/>
</if>
</filled>
</subdialog>
</form>

<!-- confirmation called by subdialog -->
<form id="confirmation">
<block>
<log>calling dialog is <value expr="calling_dialog"/></log>
</block>

<field name="iconf" type="boolean">
    <catch event="noinput nomatch">
        Sorry. I didn't get that.
        <reprompt/>
    </catch>
    <catch event="noinput nomatch" count="2">
        Sorry. I didn't get that.
        Please say yes to continue or no to go back.
    </catch>
</field>
<filled>
<log> Recognized <value expr="iconf"/> </log>
<return namelist="iconf"/>
</filled>
</form>

</vxml>

```

### 7.3 The CGI script

```
#!/usr/bin/perl

# reading input
read(STDIN, $buffer, $ENV{'CONTENT_LENGTH'});
@pairs = split(/&/, $buffer);
foreach $pair (@pairs) {
    ($name, $value) = split(/=/, $pair);
    $value =~ tr/+// ;
    $value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
    $value =~ s/<!--(.|\n)*-->//g;
    $in{$name} = $value;
}

# opening file to write data into
open(INFO, ">>export.txt");
# printing data in html format
# to be inserted into a php page
print INFO " <br>Shape: $in{shape}
           <br> Holes: $in{holes}
           <br>Pattern: $in{pattern}
           <br>";
close (INFO);

# returning vxml output to client
print "Content-type: application/x-vxml\n\n";
print <<VXML;
<vxml version="2.1">
    <form id="play_info">
        <block name="block1">
            <prompt>
                Data has been written!
                Disconnecting.
            </prompt>
        </block>
    </form>
</vxml>
VXML
```