Special Lecture (406)  
Spoken Language Dialog Systems  
VoiceXML: Dialogs, Forms and Fields  

Rolf Schwitter  
schwitt@ics.mq.edu.au
Recap: VoiceXML Architecture

- regular phone
- wireless phone
- soft phone

- telephony interface
- voice browser
- automated speech recognition
- text-to-speech synthesis
- touchtone
- audio play/record

- VoiceXML documents
- audio files
- service logic (CGI)
- transaction processing
- database interface
Recap: VoiceXML

```xml
<?xml version = "1.0"?>
<vxml version = "2.0">
  <form id = "start">
    <field name = "answer" type = "boolean">
      <prompt> Say yes or no </prompt>
      <catch event = "nomatch noinput">
        <reprompt/>
      </catch>
      <filled>
        <submit next = "cgi-bin/script.py"/>
      </filled>
    </field>
  </form>
</vxml>
```
Dynamic Scripting (CGI: script.py)

#!/usr/local/bin/python

import cgi

form = cgi.FieldStorage()

print "Content-type: text/xml\n\n"

if (form["answer"].value == 'true') :
    print "<vxml version=""2.0"">\n        <block>You just said yes</block>\n    </form></vxml>"
else:
    print "<vxml version=""2.0"">\n        <block>You just said no</block>\n    </form></vxml>"
VoiceXML Documents

• A VoiceXML document forms a conversational finite state machine.
• The caller is always in one conversational state, or dialog, at a time.
• Each dialog determines the next dialog to transition to.
• Transitions are specified using URIs, which define the next document and dialog to use.
• Execution is terminated
  – when a dialog does not specify a successor, or
  – if it has an element that explicitly exits the conversation.
Dialogs

• Instructions are presented as a set of dialog elements.
• There are two kinds of dialog elements:
  – forms
  – menus.
Forms

- Forms collect values for a set of field item variables.
- Grammars define the allowable inputs for fields.
- Platform throws events if the input is out-of-grammar.
- Actions are performed when field items are filled.
Example: Forms

<form id = "Identifier">
  <block> Message </block>
  <field name = VariableName>
    <prompt> Question </prompt>
    <grammar src = URI type = MediaType/>
    <catch event = EventType> HandlerMessage </catch>
    <filled> Actions </filled>
  </field>
</form>
Menus

- Menus present the caller with a set of options.
- Transitions to another dialog are based on a choice.
- The `<menu>` element is a shortcut for a form with only one field.
- It is a convenient way to ask the user to pick one option from a list.
Example: Menus

<menu id = Identifier>
  <prompt> Question <enumerate/> </prompt>
  <choice next = URI-1> Phrase-1 </choice>
  <choice next = URI-2> Phrase-2 </choice>
  <choice next = URI-3> Phrase-3 </choice>
  <noinput> Message <enumerate/> </noinput>
</menu>
Transitions

- Transitions are specified using URIs.
- URIs define the next document and dialog to use.
- If a URI does not refer to a document, the current document is assumed.
- If it does not refer to a dialog, the first dialog in the document is assumed.
- Transitions can be requested - for example - by:

  <choice next = URI>
  <goto next = URI>
  <link next = URI>
A Dialog for Credit Card Information

Computer: We now need your credit card type, number, and expiration date.

Computer: What kind of credit card do you have?
Caller: Discover

Computer: I did not understand what you said.

<platform-specific default message>
A Dialog for Credit Card Information

Computer: Type of card?
   <the second prompt is used now>

Caller: Shoot.
   <fortunately treated as "help" by this platform>

Computer: Please say Visa, Master card, or American Express.

Caller: Uh, Amex.
   <this platform ignores "uh">
A Dialog for Credit Card Information

Computer: What is your card number?
Caller: One two three four ... wait ...
Computer: I did not understand what you said.
Computer: Card number?
Caller: <uses DTMF>
    1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 #
A Dialog for Credit Card Information

Computer: What is your card’s expiration date?
Caller: one two oh four
Computer: I have Amex number 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 expiring on 1 2 0 4. Is this correct?
Caller: Yes
<form id = "get_card_info">
  ...
  <field name = "card_type">
    ...
  </field>
  <field name = "card_num">
    ...
  </field>
  <field name = "expiry_date">
    ...
  </field>
  <field name = "confirm">
    ...
  </field>
</form>
VoiceXML Code

```xml
<?xml version = "1.0"?>
<vxml version = "2.0">

<form id = "get_card_info">
  <block>
    We now need your credit card type, number, and expiration date.
  </block>
</form>
```
Field: card_type

<field name = "card_type">
    <prompt count = "1">
        What kind of credit card do you have?
    </prompt>
    <prompt count = "2">
        Type of card?
    </prompt>
</field>
<grammar type = "application/srgs+xml" root = "r2" version = "1.0">
  <rule id = "r2">
    <one-of>
      <item> visa </item>
      <item> master <item repeat = "0-1"> card </item> </item>
      <item> amex </item>
      <item> american express </item>
    </one-of>
  </rule>
</grammar>
Field: card_type

<help>
   Please say Visa, Mastercard, or American Express.
</help>
</field>
Field: card_num

```xml
<field name = "card_num">
  <grammar type = "application/srgs+xml"
           src = "/grammars/digits.grxml"/>
  <prompt count = "1">
    What is your card number?
  </prompt>
  <prompt count = "2">
    Card number?
  </prompt>
</field>
```
Field: card_num

```xml
<catch event = "help">
    <if cond = "card_type == 'amex' ||
        card_type == 'american express'">
        Please say or key in your 15 digit card number.
    </if>
    <else/>
        Please say or key in your 16 digit card number.
    </else/>
</catch>
```
<filled>
  <if cond="(card_type == 'amex' ||
              card_type == 'american express') &&
              card_num.length != 15">
    American Express card numbers must have 15 digits.
  </if>
  <clear namelist = "card_num"/>
  <throw event = "nomatch"/>
Field: card_num

<elseif cond = "card_type != 'amex' &amp;&amp; card_type != 'american express' &amp;&amp; card_num.length != 16"/>

Mastercard and Visa card numbers have 16 digits.
<clear namelist = "card_num"/>
<throw event = "nomatch"/>
</if>
</filled>
</field>
Field: expiry_date

<field name = "expiry_date">
    <grammar type = "application/srgs+xml"
        src = "/grammars/digits.grxml"/>
    <prompt count = "1">
        What is your card's expiration date?
    </prompt>
    <prompt count = "2">
        Expiration date?
    </prompt>
    <help>
        Say or key in the expiration date, for example one two oh one.
    </help>
</field>
Field: expiry_date

<filled>
    <var name = "mm"/>
    <var name = "i" expr = "expiry_date.length"/>
    <if cond = "i == 3">
        <assign name = "mm" expr = "expiry_date.substring(0, 1)"/>
    </if>
    <elseif cond = "i == 4">
        <assign name = "mm" expr = "expiry_date.substring(0, 2)"/>
    </if>
Field: expiry_date

<if cond = "mm == ' ' ||
    mm < 1 ||
    mm > 12">
    <clear namelist = "expiry_date"/>
    <throw event = "nomatch"/>
</if>
</filled>
</field>
<field name = "confirm">
    <grammar type = "application/srgs+xml"
             src = "/grammars/boolean.grxml"/>
    <prompt>
        I have <value expr = "card_type"/>
        number <value expr = "card_num"/>
        expiring on <value expr = "expiry_date"/>
        Is this correct?
    </prompt>
</field>
Field: confirm

```xml
<filled>
    <if cond = "confirm">
        <submit next = "place_order.asp"
            namelist = "card_type card_num expiry_date"/>
    </if>
    <clear namelist = "card_type card_num expiry_date confirm"/>
</filled>
</field>
</form>
</vxml>
```
Prompt Element

• The `<prompt>` element controls the output of synthesized speech and prerecorded audio.

• Important attributes are:
  – bargein controls whether the caller can interrupt a prompt
  – cond an expression telling if the prompt should be spoken
  – count a number that allows to emit different prompts
  – timeout for the following caller input
Tapered Prompts

• Prompts can be used to vary a message given to the human.

<field name = "card_type">
  <prompt count = "1">
    What kind of credit card do you have?
  </prompt>
  <prompt count = "2">
    Type of card?
  </prompt>
</field>

• Prompts may be tapered to be:
  – more terse with use (field prompting)
  – more explicit (help prompts).
Catch and Help Element

• The `<help>` element is an abbreviation for

  `<catch event = "help">`
  ...
  `</catch>`

• For example

  `<help>`
  Please say Visa, Mastercard, or American Express.
  `</help>`

• Additional attributes are: "count" and "cond".
Conditions

- The `<if>` element is used for conditional logic.
- It has optional `<else>` and `<elseif>` elements.
- The expression language used is ECMAScript.

```xml
<if cond = "(card_type == 'amex' || card_type == 'american express') && card_num.length != 15">
  ...
  <elseif .../>
  ...
  </if>
```

- The "cond" operator "&&" needs to be escaped "& &".
Variables

• Variables are declared by `<var>` elements:

  ```xml
  <var name = "mm"/>
  <var name = "i" expr = "expiry_date.length"/>
  ```

• Variables are also declared by form items:

  ```xml
  <field name = "card_type"> ... </field>
  ```

• Attributes are:

  – name  the name of the variable that will hold the result
  – expr  initial value is optional
Assign Element

• The `<assign>` element assigns a value to a variable:

  `<assign name = "mm" expr = "expiry_date.substring(0,1)"/>

• Variables need to be declared before making an assignment.

• Attributes are:
  – name the name of the variable being assigned to
  – expr the new value of the variable
Clear Element

• The `<clear>` element resets one or more variables.
• For example:
  <clear namelist = "card_num"/>
• The attribute "namelist" contains the variables to be reset.
Throw Element

- The `<throw>` element throws an event.
- This can be a pre-defined one:
  
  `<throw event = "nomatch"/>`

  or an application-defined one:

  `<throw event = "com.att.portal.machine"/>`
Submit Element

• The `<submit>` element is used to submit information to a server:

  `<submit next = "place_order.asp"
          namelist = "card_type card_num expiry_date"/>

• It lets you submit a list of variables to the document server via an HTTP GET or POST request:

  `<submit next = "place_order.asp" method = "post"
          namelist = "card_type card_num expiry_date"/>`
Take-Home Message

- A VoiceXML document forms a finite state machine.
- Each dialog determines the next dialog to transition to.
- There are two kinds of dialogs: forms and menus.
- The expression language used in "cond" and "expr" is ECMAScript.
- Operators "<", "<=", and "&&" must be escaped in XML.