COMP349
Spoken Language Dialog Systems
Dialogs, Forms and Fields

Rolf Schwitter
schwitt@ics.mq.edu.au
Today’s Program

• Documents
• Dialogs
• Forms
• Menus
• Transitions
• Dialog for Credit Card Information
• More VoiceXML Elements
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

• Dialog elements present information and collect data.
• 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.
- Event handlers process any resulting events.
- Actions are performed when a field item variable is filled.
Example: Form

```html
<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>
```
Example: Form

```xml
<form id="get_state">
  <field name="state">
    <prompt> Please say the name of a state. </prompt>
    <grammar src="http://mygrammars.example.com/states.gram"
      type="application/srgs" />
    <filled>
      <if cond="state$.confidence &lt; 0.4">
        <throw event="nomatch"/>
      </if>
    </filled>
  </field>
</form>
```

- Note: a shadow variable is referenced as `name$.shadowvar`
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 that
  – prompts the caller to make a choice
  – transitions to different places based on the choice.
• It is a convenient way to ask the user to pick one option from a list.
Example: Menu

```xml
<menu id = "Identifier">
  <prompt> Question <enumerate/> </prompt>
  <choice next = "URI-1"> Phrase-1 Grammar-1 </choice>
  <choice next = "URI-2"> Phrase-2 Grammar-2 </choice>
  <choice next = "URI-3"> Phrase-3 Grammar-3 </choice>
  <noinput> Message <enumerate/> </noinput>
</menu>
```
Example: Menu

```xml
<menu>
  <choice next="http://www.sports.example.com/vxml/start.vxml">
    <grammar src="sports.grxml" type="application/srgs+xml"/>
    Sports
  </choice>
  <choice next="http://www.weather.example.com/intro.vxml">
    <grammar src="weather.grxml" type="application/srgs+xml"/>
    Weather
  </choice>
  <choice next="http://www.stargazer.example.com/voice/astronews.vxml">
    <grammar src="astronews.grxml" type="application/srgs+xml"/>
    Stargazer astrophysics
  </choice>
</menu>
</vxml>
```
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

<table>
<thead>
<tr>
<th>Computer:</th>
<th>Type of card?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;the second prompt is used now&gt;</td>
</tr>
<tr>
<td>Caller:</td>
<td>Shoot.</td>
</tr>
<tr>
<td></td>
<td>&lt;fortunately treated as &quot;help&quot; by this platform&gt;</td>
</tr>
<tr>
<td>Computer:</td>
<td>Please say Visa, Master card, or American Express.</td>
</tr>
<tr>
<td>Caller:</td>
<td>Uh, Amex.</td>
</tr>
<tr>
<td></td>
<td>&lt;this platform ignores &quot;uh&quot;&gt;</td>
</tr>
</tbody>
</table>
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 six
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 6. 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" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml">
  <form id = "get_card_info">
    <block>
      We now need your credit card type, number, and expiration date.
    </block>
  </form>
</vxml>
```
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>
Field: card_type

<grammar type = "application/srgs+xml" root = "r2" version = "1.0">
  <rule id = "r2" scope = "public">
    <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

<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>
<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>
Field: card_num

<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' && card_type != 'american express' && 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 six.
  </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)"/>
  </elseif>
</if>
Field: expiry_date

```xml
<if cond = "mm == '' ||
     mm < 1 ||
     mm > 12">
    <clear namelist = "expiry_date"/>
    <throw event = "nomatch"/>
</if>
</filled>
</field>
```
Field: confirm

<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 name="confirm">
  <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"/>
</field>
Prompt Element

• The `<prompt>` element controls the output of
  – synthesized speech
  – recorded 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` interval of silence before a `<noinput>` event is thrown.
Tapered Prompts

• Prompts can be used to vary a question:

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

• Prompts may be tapered to be:
  – more terse with use (field prompts)
  – 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 (= JavaScript).

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

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

• Variables are declared by `<var>` elements:
  
  `<var name = "mm"/>`
  `<var name = "i" expr = "expiry_date.length"/>`

• Variables are also declared by form items:
  
  `<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:

  ```xml
  <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:

  ```html
  <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:

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

- 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 JavaScript.
- Operators "<", "<=" , and "&&" must be escaped in XML.