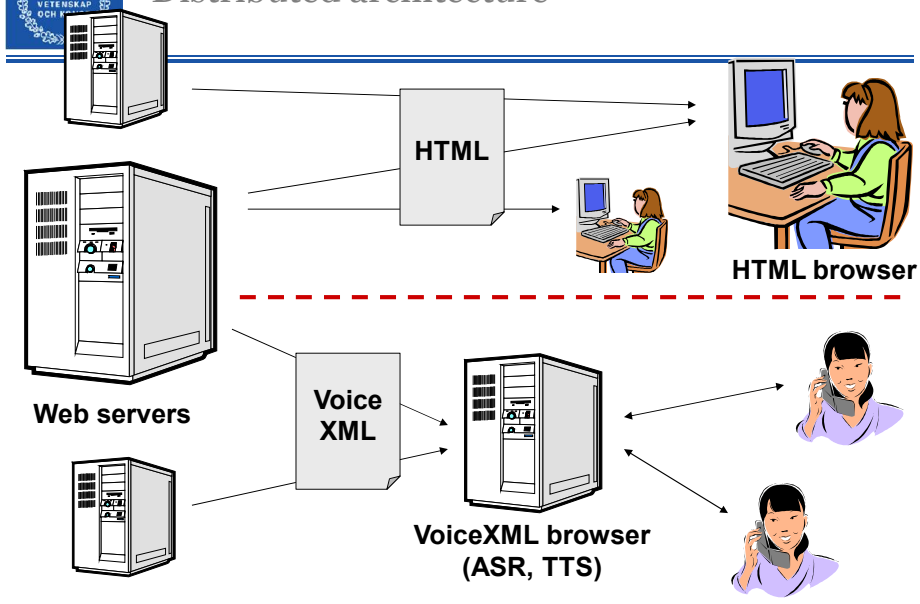
VoiceXML
Developing telephone-based dialogue systems

Gabriel Skantze
gabriel@speech.kth.se
Department of Speech Music and Hearing, KTH

What is VoiceXML?

- An XML-based script-language for developing telephone-based dialogue systems
- A standard developed by W3C
  - Most recent version is 2.0
  - Working draft of 3.0
- A VoiceXML-application is a web application
  - The application resides on a web server and is available over the Internet
  - The dialogue is executed by a **voice browser**
  - Distributed architecture
  - Compare with (X)HTML

Distributed architecture



VoiceXML manages:

- Prompts with synthetic speech
- Prompts with pre-recorded speech and sounds
- Recognition of spoken words and phrases
- Recognition of DTMF (buttons)
- Voice recordings
- Control of dialogue flow (menus, error handling, sub dialogue, etc)
- Control of telephony (call routing, hang-up, etc)

Related standards

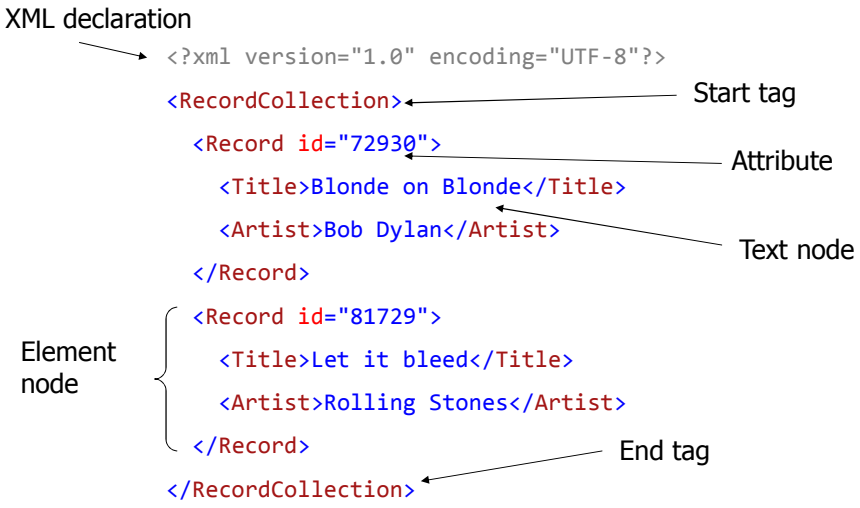
- SRGS - Speech Recognition Grammar Specification
- SISR - Semantic Interpretation for Speech Recognition
- SSML - Speech Synthesis Markup Language
- ECMAScript (JavaScript) - the scripting language used in VoiceXML

- VoiceXML 3.0:
  - CCXML - Call Control eXtensible Markup Language
  - SCXML - State Chart XML

A crash course in XML

- XML = Extensible Markup Language
- Defined by W3C
  - Often used in web applications
- Used to encode arbitrary data structures
  - Tree-structured data
  - Domain Specific Languages (such as VoiceXML)
  - Human-readable, Machine-readable
- Related standards
  - Validation: DTD, XML Schema (XSD)
  - Transformation: XSLT

XML example: a record collection



XML: Things to remember

- Each start tag must have a corresponding end tag.
  `<Tag>...</Tag>`
- The attributes must be on the form:
  `attribute=”value”`
- An empty tag may be formatted as:
  `<Tag/>`
  which is exactly the same thing as:
  `<Tag></Tag>`
- Comments are written as:
  `<!-- this is a comment -->`
- The symbols < and > may not be used in any place except for formatting tags
  Use `<` and `>` instead

XML namespaces

```
<?xml version="1.0" encoding="iso-8859-1”?>
<RecordCollection xmlns=”http://www.myrecordcollection.com/”>
  <Record id="72930">
    <Title>Blonde on Blonde</Title>
    <Artist>Bob Dylan</Artist>
  </Record>
  <Record id="81729">
    <Title>Let it bleed</Title>
    <Artist>Rolling Stones</Artist>
  </Record>
</RecordCollection>
```

All the nestled tags are placed in a namespace

XML applications

- **XML** defines how the mark-up should be formatted. (but not *which* tags may be used)
- For an XML document to be “**wellformed**”, it must follow this standard.
- An **XML schema** or a **DTD** (document type definition) describes an **XML application**. This regulates which tags and attributes may be used and how they may be structured.
- If an XML document follows the **XML schema** or **DTD**, it is “**valid**”.

”Non-wellformed” XML

```
<?xml version="1.0" encoding="iso-8859-1”?>
<RecordCollection xmlns=”http://www.myrecordcollection.com/”>
  <Record id="72930">
    <Title>Blonde on Blonde</Title>
    <Artist>Bob Dylan</Artist>
  </Record>
  <Record id=81729>
    <Title>Let it bleed</Title>
    <Artist>Rolling Stones
  </Record>
</RecordCollection>
```

Attribute-values must be stated within quotations

End tag is missing for <Artist>

”Non-valid” XML

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE RecordCollection SYSTEM "RecordCollection.dtd">
<RecordCollection>
  <Record id="72930">
    <Title>Blonde on Blonde</Title>
  </Record>
  <Record id="81729">
    <Title>Let it bleed</Title>
    <Artist>Rolling Stones</Artist>
  </Record>
</RecordCollection>
```

A DTD (Document Type Definition) is specified for this document

Artist is missing (required by the DTD)

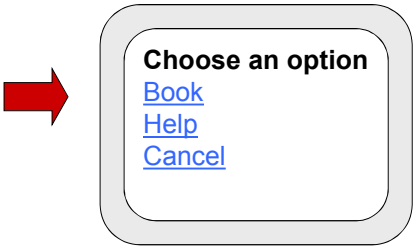
This document is well-formed, but not valid!

Different XML-applications

- **(X)HTML** is an XML-application for describing graphical presentation of text and interaction through links and forms.

XHTML (fragment)

```
<p>
  <b>Choose an option</b>
  <br/>
  <a href="book.html">Book</a>
  <br/>
  <a href="help.html">Help</a>
  <br/>
  <a href="end.html">Cancel</a>
</p>
```

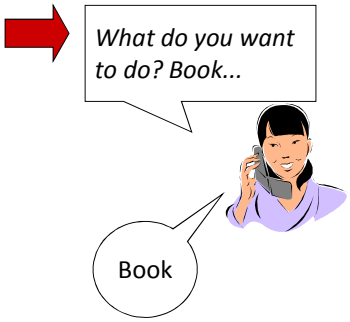


VoiceXML

- In a similar fashion, **VoiceXML** describes the dialogue, using grammars, mark-up for speech synthesis, etc.

VoiceXML (fragment)

```
<menu>
  <prompt>What do you want to do?  Book,
          help or cancel?</prompt>
  <choice next="book.vxml">book</choice>
  <choice next="help.vxml">help</choice>
  <choice next="end.vxml">cancel</choice>
</menu>
```



Hello World

```
<?xml version="1.0"?>
<!DOCTYPE vxml
  PUBLIC "-//W3C//DTD VOICEXML 2.1//EN"
    "http://www.speech.kth.se/~gabriel/voicexml/vxml.dtd">
<vxml xmlns="http://www.w3.org/2001/vxml" version="2.1”
    xml:lang="en-US">
  <form>
    <block>
      <prompt>Hello world</prompt>
    </block>
  </form>
</vxml>
```

Menu, Forms and Links

S: Welcome to the travel booking system.
What do you want to do?
Book a trip, Speak to an operator, End the call.

Menu

A: Book a trip

S: From where do you want to go?

A: From Paris

S: Where do you want to go?

A: London

S: Which day?

Form

A: *Abort*

S: Aborting. Welcome to the travel…

Link

Menu: <menu>

```
<?xml version="1.0"?>
<vxml xmlns="http://www.w3.org/2001/vxml" version="2.1"
       xml:lang="en-US">
  <menu>
    <prompt>
      Welcome to the travel booking system. What do you want to
       do?
      <enumerate/>
    </prompt>
    <choice next="booking.vxml">book a trip</choice>
    <choice next="operator.vxml">speak to an operator</choice>
    <choice next="end.vxml">end the call</choice>
  </menu>
</vxml>
```

Form: <form>

```
<?xml version="1.0"?>
<vxml xmlns="http://www.w3.org/2001/vxml" version="2.1" xml:lang="en-US">
  <form>
    <field name="origin">
      <grammar src="grammar.xml#Origin"/>
      <prompt>From where do you want to go?</prompt>
    </field>
    <field name="destination">
      <grammar src="grammar.xml#Destination"/>
      <prompt>Where do you want to go?</prompt>
    </field>
    ...
    <filled>
      <prompt>
        Ok, you want to go from <value expr="origin"/> to
          <value expr="destination"/>
      </prompt>
      <submit next="booking_to.vxml"/>
    </filled>
  </form>
</vxml>
```

Link: <link>

- A Link may either:
  - Transfer the dialogue to another state ("goto")
  - Throw an event

```
<link next="main_menu.vxml">
  <grammar>...</grammar>
</link>

<link event="help">
  <grammar>...</grammar>
</link>

<catch event="help">
  <prompt>Just say the name of the city you are travelling from. To
      return to the main menu, say "main menu".</prompt>
</catch>
```

Grammar in a <field>

- A grammar specifies what the user can say
  - Option-list

```
<field name="color">
    <prompt>Which color?</prompt>
    <option>red</option>
    <option>blue</option>
    <option>green</option>
</field>
```

  - Field type

```
<field name="confirm" type="boolean">
   <prompt>Please say yes or no</prompt>
</field>
```

    date, time, boolean, number, currency, etc.

  - Grammar
    external

```
<field name="origin">
     <grammar src="grammar.xml#Origin"/>
     <prompt>From where do you want to go?</prompt>
</field>
```

    internal

```
<grammar>[hit hold split (double down)]</grammar>
```

SRGS XML format

```
<?xml version="1.0"?>
<!DOCTYPE grammar PUBLIC "-//W3C//DTD GRAMMAR 1.0//EN"
    "http://www.speech.kth.se/~gabriel/voicexml/grammar.dtd">
<grammar xmlns="http://www.w3.org/2001/06/grammar" root="START"
xml:lang="en-US" version="1.0">

  <rule id="START" scope="public">
    i want to go to <ruleref uri="#CITY"/>
  </rule>

  <rule id="CITY">
    <one-of>
      <item>london</item>
      <item>paris</item>
    </one-of>
  </rule>

</grammar>
```

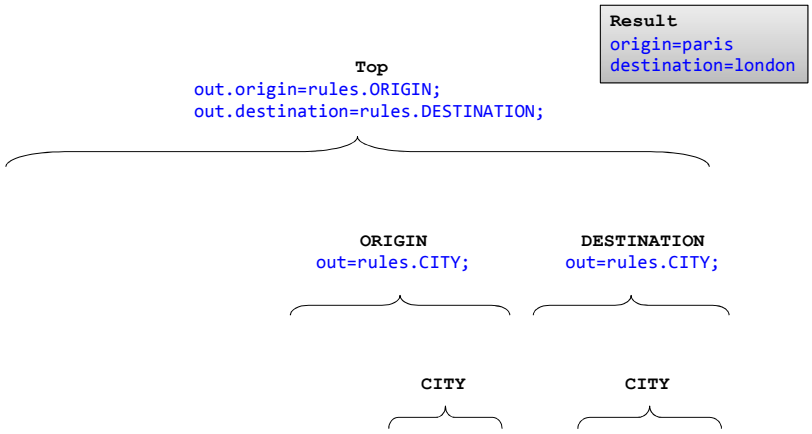
SRGS Semantics: <tag>

```
<rule id="START" scope="public">
  <item>
     <ruleref uri="#ORIGIN"/>
     <ruleref uri="#DESTINATION"/>
     <ruleref uri="#DATE"/>
     <tag>out.origin=rules.ORIGIN;
          out.destination=rules.DESTINATION;
          out.date=rules.DATE;</tag>
  </item>
</rule>

<rule id="ORIGIN" scope="public">
    <item repeat="0-1">
        <item>from</item>
    </item>
    <ruleref uri="#CITY"/>
    <tag>out=rules.CITY;</tag>
</rule>
```

SISR - Semantic Interpretation for Speech Recognition

Matching of grammar



"I want to go from paris to london"

Semantics in a field

S: From where do you want to go?
U: Paris

Result
{paris}

Grammar on the **field** level

```
<field name="origin">
  <grammar src="grammar.xml#ORIGIN"/>
  <prompt>From where do you want to go?</prompt>
</field>
```

Semantics in a form

U: I want to go from Paris to London

Result
{origin: paris
destination: london}

Grammar on the **form** level

```
<form>
  <grammar src="grammar.xml#TOP"/>
  <field name="origin">
    <grammar src="grammar.xml#ORIGIN"/>
    <prompt>From where do you want to go?</prompt>
  </field>
  <field name="destination">
    <grammar src="grammar.xml#DESTINATION"/>
    <prompt>Where do you want to go?</prompt>
  </field>
  ...
</form>
```

Mixed-initiative

Variable/Slot	**<prompt>**
Initial question: <initial name="welcome">	"Welcome to the travel booking system. You can now make your reservation."
<field name="origin">	"From where do you want to go?"
<field name="destination">	"Where do you want to go to?"
<field name="date">	"Which date do you want to go?"
<field name="time">	"At what time do you want to go?"

FIA: Form Interpretation Algorithm

- Go through each field in the form, in sequential order.
- Stop at the first field that is not filled.
- Play the prompt, start recognition.
- Fill the fields in the form that the matching grammar specifies.
- If all fields are filled:
  - Leave the form.
  - Otherwise: reiterate.

FIA example

S: Welcome to the travel booking system. You can now make your reservation
*U: I want to go to **London***
S: From where do you want to go?
*U: From **Paris***
S: Which date do you want to go?
*U: On the **13th of January***
S: At what time do you want to go?
*U: **Three** o'clock*
S: Thanks for your reservation …

Origin	Paris
Dest	London
Date	13th of January
Time	Three

FIA example

*U: I want to go from **Paris** to **London***
S: Which date do you want to go?
*U: **Three** o'clock on the **13th of January***
S: Thanks for your reservation …

Origin	Paris
Dest	London
Date	13th of January
Time	Three

Controlling the Form Interpretation Algorithm

- Go to a specific field

```
<goto nextitem="destination"/>
```

- Set a value to a field

```
<assign name="destination" expr="'London'"/>
```

- Clear fields

```
<clear/>
<clear namelist="destination origin"/>
```

<filled>

- Contains instructions to be executed when a <field> or <form> is filled, such as:

  <prompt> <goto> <if> <throw> <clear> <assign>

```
<form>
  <field name="origin">
    <grammar src="grammar.xml#Origin"/>
    <prompt>From where do you want to go?</prompt>
    <filled>
      <prompt>
        Ok, you want to go from <value expr="origin"/>
      </prompt>
    </filled>
  </field>
  ...
  <filled>
    <prompt>
      Ok, you want to go from <value expr="origin"/> to <value expr="destination"/>
    </prompt>
  </filled>
</form>
```

Error handling

S: From where do you want to go?

A: I want to go from Paris

S: Sorry, I didn't understand. From where do you want to go?

A: I said Paris!

S: Please just say the name of the city. From where do you want to go?

A: Paris

S: You want to go from Paris.

<nomatch> <noinput>

```
<form>
  <field name="origin">
    <grammar src="grammar.xml#Top"/>
    <prompt>From where do you want to go?</prompt>
    <nomatch count="1">
      Sorry, I didn't understand.<reprompt/>
    </nomatch>
    <nomatch count="2">
      Please just say the name of the city.<reprompt/>
    </nomatch>
    <nomatch count="3">
      There seems to be a problem, I will connect you to an operator.
          <goto next="operator.vxml"/>
    </nomatch>
    <noinput count="1">
      You must say the name of a city.<reprompt/>
    </noinput>
    <noinput count="3">
      There seems to be a problem, I will connect you to an operator.
          <goto next="operator.vxml"/>
    </noinput>
  </field>
</form>
```

Variables

- Variables :
  - Definition
    ```
    <var name="city" expr="'Paris'"/>
    ```
  - Assignment
    ```
    <assign name="city" expr="'London'"/>
    ```
  - Clearing:
    ```
    <clear namelist="origin time"/>
    ```
- Fields (in a form) are also variables!

If-then-else

```
<if cond="price &gt; 1000">
  <prompt>That was very expensive!</prompt>
<elseif cond="price &gt; 700"/>
  <prompt>That was expensive!</prompt>
<elseif cond="price &gt; 500"/>
  <prompt>That was fairly expensive!</prompt>
<else/>
  <prompt>That was not so expensive!</prompt>
</if>
```

Prompts

- Text is read by a speech synthesiser that may be controlled:
  - Emphasis

    ```
    <emphasis>Welcome</emphasis> to the travel booking system.
    ```
  - "Say-as"

    ```
    <say-as type="telephone">08663995</say-as>
    ```
  - Choice of synthesis

    ```
    <voice gender="female">This is a female voice</voice>
    ```
- Speech Synthesis Markup Language (SSML)
  - W3C standard

Prompts, cont

- Recorded audio may be mixed with speech synthesis:

```
<prompt>
  A black bird sounds like this: <audio src="blackbird.wav"/>
</prompt>
```

- ”Barge-in” may be controlled:

```
<prompt bargein="false">
  Now you cannot interrupt me
</prompt>
```

Programming in VoiceXML

- The scripting functionality in VoiceXML is limited
- EcmaScript (JavaScript)
  - Evaluated at the client
  - Used for example to define own functions
- Server programming
  - JSP, PHP, Perl, etc
  - Evaluated at the server
  - Dynamic pages

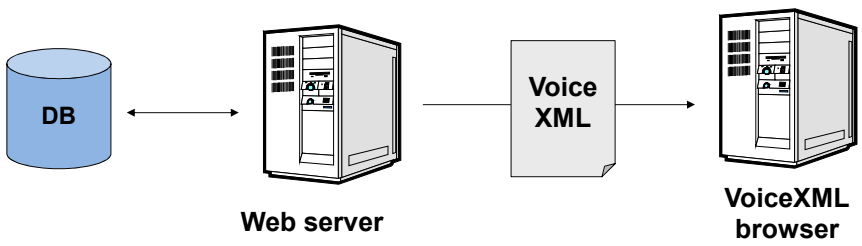
EcmaScript (JavaScript)

```
<script>
  function random() {
       return Math.floor(Math.random() * 100) + 1;
  }
</script>

<prompt>
  <value expr="random()"/> is a random number between 1 and 100.
</prompt>
```

Server programming

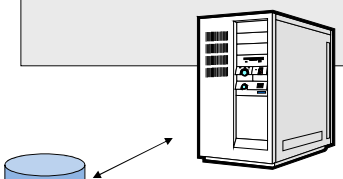
- To be able to present and store dynamic information (weather, news, user preferences) the application must use a database. There is no support for this in VoiceXML alone.



Dynamic VoiceXML pages (PHP)

weather.php - VoiceXML with embedded PHP script

```
<form>
  <block>
    The temperature for today
    will be
    <? print temp(); ?>
    degrees.
  </block>
</form>
```



VoiceXML browser

```
<form>
  <block>
    The temperature
    for today will
    be 13 degrees.
  </block>
</form>
```

DB

Webb-server
executes PHP code

ROYAL INSTITUTE OF TECHNOLOGY

THE END