Dialogue systems

Luděk Bártek

Laboratory of Searching and Dialogue, Fakulty of Informatics, Masaryk University, Brno

spring 2023

Standards for Dialogue Interfaces Development

- Objective – allow a dialogue interface transfer from a platform to another.
- Standards for Dialogue Interface Development:
  - W3C Voice Browser Activity Standards,
  - AIML,
  - implementation platforms de-facto standards.

World Wide Web

- 1876 – The patent for the phone granted to A. G. Bell
- WWW
  - 1989 – The article "HyperText and CERN" (Tim Burnes Lee) circulates around the CERN.
  - Christmas 1990 – command line web browser and editor has been demonstrated.
  - 1991 – WWW general availability on computers in CERN.
  - 1994 – The W3 Consortium first meting.

W3C Voice Browser Activity

- 1999 – the W3C Voice Browser Working Group formed.
- Objective – proposal of standards allowing web browsing and access using a voice and phone.
- Members:
  - HP
  - Nuance Communications
  - Lucent Technologies
  - Motorola
  - ScanSoft
  - IBM
  - Tellme Networks
  - Vocalocity
  - ...

W3C Voice Browser Activity Standards

- VoiceXML – a dialogue strategy description language.
- Speech Recognition Grammar Specification – the language for description of speech recognition grammars.
- Semantic Interpretation for Speech Recognition – language for semantic interpretation support.
- Speech Synthesis Mark-up Language – language for description of the sentence prosody factors for speech synthesis.

Standardy W3C Voice Browser Activity

Cont.

- Pronunciation Lexicon Specification – the pronunciation lexicon for a speech recognition and synthesis
- Call Control XML – mark-up language used to control user – dialogue system connection.
- State Chart XML – general use state machine description language.

Standardy W3C Voice Browser Activity

Processing

- The standards are mark-up languages – must be interpreted.
- There is a lot of implementation platforms:
  - Free available desktop platforms – JVoiceXML, PublicVoiceXML, ...
  - Commercial desktop – Optimtalk – there was a free available version for development and testing; LSD lab has a licence for computers in the lab.
  - Free available on-line platforms - Asterisk+VoiceGlue or OpenVXI, JVoiceXML...
  - Commercial on-line – Aspect Prophecy, Bevocal Cafe – can be used freely for development and testing purposes (2 parallels calls max)

Speech Recognition Grammar Specification

- W3C specification of a language for description of context free grammars supporting the speech recognition.
- Current version 1.0.
- The JSGF replacement.
- There are 2 two different notations:
  - XML
  - Augmented Backus-Naur Form (ABNF).
- The only difference is the notation not the power of expression.
- Support of the notations is platform depended.
  - Commonly used is the XML form.

Context Free Grammars

The Formal Languages Theory

- Grammar $G = (N, \Sigma, P, S)$
  - N – finite set of non-terminal symbols
  - Σ – finite set of terminal symbols (the language alphabet)
  - P – a set of rules
  - S – grammar root non-terminal symbol
- Context Free Grammar:
  - grammar $G = (N, \Sigma, P, S)$
  - where the rules are in the form $N \rightarrow \{N \cup \Sigma\}^*$

SRGS grammar

XML form

- Starts with an XML prologue
  - <?xml version="1.0" encoding="..."?>.
- Root element – *grammar*; contains the set of rules (*rule* elements).
- Attributes
  - *version* – used SRGS version (current 1.0).
  - *xml:lang* – grammar language code.
  - *root* – rule id corresponding the the root non-terminal symbol.
  - *mode* – the communication mode of the grammar:
    - dtmf – using the DTMF codes
    - voice – voice grammar; the implicit value.

Rule Notation

- Element *rule*:
  - attributes:
    - *id* – the rule id (corresponds to the rule left-side non-terminal symbol).
  - Content – right side of the rule:
    - textual content – a sequence of terminal symbols.
    - element *ruleref* – non-terminal symbol; referenced by *uri* attribute.
    - element *one-of* – right side variants (operator $|$).
    - element *item* – logical division of the sequence; allows to specify the the count of repeating of particular part of the speech for example.

Sequence

- Sequence of Terminal and Non-terminal Symbols.
  SAMPLE $\rightarrow$ I like *KIND* SRGS form.

```
<rule id="sample">
 I like <ruleref uri="#typ"/> SRGS form.
</rule>
```

- Right side of the rule can be divided into the logical parts:

```
<rule id="address">
 <item repeat="0-1">
  <ruleref uri="#server"/>
 </item>
 <item repeat="1-5">
   dot <ruleref uri="#domain"/>
 </item>
</rule>
```

Rule Variants

- Element *one-of*.
- It allows to specify different variants of expected inputs.
- The variants are enclosed in element *item*.
- Example:

```
<rule id="colors">
 <one-of>
  <item>red</item>
  <item>green/item>
  <item>blue</item>
 </one-of>
</rule>
```

Repeating

- It allows to specify:
  - the optional parts of the speech
  - the recurrent parts of the speech
- Notation – using the the attribute *repeat* of the element *item*.
- Repeating count possible specification:
  - n times– n:

```
<item repeat="2">repeating</item>
```

  - $< m, n >$ times– m-n

```
<item repeat="0-1">
  I'd like to
</item>
```

  - $< n, \infty)$ times– m-

```
<item repeat="1-">Hi</item>
```

Special Rule

- Can be used to input:
  - the unspecified utterance – *GARBAGE*
  - the unspeakable rule (prohibition of particular utterance) – *VOID*
  - ever valid rule (even empty) – *NULL*
- They are used as special non-terminal symbols:

```
<ruleref special="GARBAGE"/>
```

- Usage example:

```
<rule id="connection">
 <ruleref special="GARBAGE"/>
 from <ruleref uri="#place"/>
 to <ruleref uri="#place"/>
 using
 <ruleref uri="#transport"/>
</rule>
```

The SRGS ABNF Form

- Pure plain-text grammar form based on the BNF form.

```
<spojeni> ::= I want to go from <from> to <wh
                  at <when>"."
<when> ::= <day> <time>
<what> ::= train | bus
...
```

- BNF based form is used by JSGF for example.

ABNF Form of SRGS

- Grammar header – may contain:
  - grammar natural language specification
  - grammar mode – voice/dtmf
  - root non-terminal
  - ...
- Grammar rules
  - form – $\$\mathit{non\text{-}terminal} = (\mathit{non\text{-}terminal}|\mathit{terminal})^*$
  - *non-terminal* = XML form rule id.

SRGS Grammar ABNF form Header Structure

- Starts with document type specification.
  - #ABNF *SRGS_version grammar_encoding*

```
#ABNF 1.0 ISO-8859-2
```

- Followed by:
  - the root non-terminal symbol specification – root $*non-terminal*;
  - grammar native language – language *language code*;

```
language en-US;
```

  - grammar usage mode – mode (voice|dtmf);

SRGS Grammar ABNF Notation

- Sequence – sequence of white-space separated terminal and non-terminal symbols:

```
$greating = Good day;
$date = $day th of $month $year;
```

- Variants – corresponding sequences of terminal and non-terminal symbols separated by symbol '|':

```
$transport = bus| train;
```

- Repeating:
  - optional parts of the speech – enclosed to '[ ]'
  - m—n – <m-n>

ABNF Form of SRGS Grammar Example


```
#ABNF 1.0 UTF-8;
root $url;
language en-US;
mode voice;

$url = [$protocol][$server] dot
       ($domain dot)<1-3>
       $tld[$path];
$protocol = http | ftp | telnet | gopher | ...;
$path = (/ $directory ) <1-> /[$file];
...
```

Semantic Interpretation for Speech Recognition

- Semantic – assigns the meaning to the expressions/sentences.
- Dialogue system and semantic:
  - assigns the interpretation to the utterances and to their parts
  - allows to extract relevant data from speech.
- SISR – W3C Voice Browser Activity Standard
  - serves for semantic interpretation of utterances
  - published on April 2007
  - current version 1.0.
  - is closely related to standards:
    - ECMA Script – the semantic interpretation uses the ECMAScript expressions
    - SRGS – the semantic evaluation is assigned to the SRGS rules using the rules attributes.
    - JSON – the interpretation is internally represented using a JSON objects.

Assignment of Interpretation to the Part of the Speech

- Semantic interpretation tends to be part of the SRGS rules.
- Assigned to the rule using either the tag or the element *tag*:
  - SRGS XML Form:
    - element tag:

```
<item>
  <ruleref uri="aggrement"/>
  <tag>{out ='yes'}</tag>
</item>
```

    - attribute tag:

```
<item tag="yes">Yeah</item>
```

  - SRGS ABNF Form:
    - interpretation follows the interpreted part of the speech.
    - from: {*interpretation*}

```
$claim = $aggre {yes} | $disagree {no}
```

Deriving the Interpretation of Whole Using Interpretations of Parts

- Derivation notation – using ECMAScript expressions.
- Assignment of derivation to the rules using either element of attribute *tag*.
- Resulting interpretation is represented as JSON object.
- Utterances evaluation:
  - partial interpretation access – rule right side symbols meaning access:
    - shadow variable *rules* attributes
    - the attribute *N* corresponds to the non-terminal *N*.
  - to return the interpretation into to the left-side non-terminal – object *out*.
  - to return interpretation to the dialogue:
    - *out* object attributes
    - to the input field *N* corresponds attribute *N*.

Utterances Evaluation

XML form


```
<rule id="ownership">
<item>
 Mám
 <item repeat="0-1">
  <ruleref uri="#color"/>
 </item>
 <ruleref uri="#transport">
 <tag>
 {
  out= rules.color+ ";" + rules.transport;
 }
 </tag>
</item>
</rule>
...
```

Utterances Evaluation

ABNF Form


```
$ownership = I own  $color <0-1> $transport
{
    out = rules.color+ ";" + rules.transport;
};
```

Interpretation Assignment to the Input Fields

XML Form


```
<rule id="ownership">
 <item>
 I own
  <item repeat="0-1"><ruleref uri="#color"/></ite
  <ruleref uri="#transpor"/>
  <tag>
   {
     out.color = rules.color;
     out.transport = rules.transport;
   }
  </tag>
 </item>
</rule>
```

Interpretation Assignment to the Input Fields

ABNF form


```
$ownership = I own $color <0-1> $transport
 {
  out.color = rules.color;
  out.transport = rules.transport;
 };
```

Literature

- SRGS Specification
- SISR Specification
- ECMAScript Specification
- JSON Specification