Deutsche Telekom Laboratories

W3C SIV Workshop (Menlo Park, March 5-6, 2009)
Ingmar Kliche, Martin Eckert
W3C SIV Workshop.
Agenda.

- SIV Architecture
- Use cases
- SIV syntax
- Conclusion
W3C SIV Workshop.
What should SIV in VoiceXML 3.0 support?

Combination of SIV with other resources (esp. ASR):
- SIV only (i.e. without ASR, standalone SIV)
- SIV in parallel to ASR (ASR and SIV are separate resources)
- SIV integrated with ASR as one (combined) resource

SIV types:
- Text independent
- Text dependent
- Text prompted

Decision control:
- Either the SIV engine or the application may control decisions (e.g. regarding acceptance/rejection)
SIV must support:

- Enrollment
- Verification
- Identification

Further basic/core functionalities for application development:

- Adaptation of voiceprints (during verification)
- Buffering of user utterances for later use
- Rollback/Undone of last turn
- Query SIV results (e.g. accept/reject information, score etc.)
- Catch SIV events (e.g. “noinput” or “nomatch” events)
- Query, copy, delete voiceprints (administration purposes) ⇒ outside of VoiceXML 3.0

Note: V3 should load/store voiceprints implicitly (without explicit markup)
W3C SIV Workshop.
SIV Architecture.

Proposed Architecture

- Standard VoiceXML architecture extended by MRCP-based SIV engine and voiceprint store
Architectural key statements

- Support MRCP v2 for integration of SIV engines
  - SIV engine should be integrated using a standardized interface to allow flexible replacement of SIV resources (product replacement).

- Extend MRCP vs. limited SIV functionalities
  - Some SIV vendors require functionalities which are not covered by MRCP v2 (e.g. COPY voiceprint, expected utterance). A decision is necessary for either using a standardized interface or to support the full set of SIV features of various vendors.

- Use EMMA for representation of SIV results
  - SIV results should be represented using EMMA standard.

- Use web protocols for voice print transport
  - Use of HTTP/HTTPS provide flexibility in deployment scenarios
Voiceprint management: load and save voiceprints via MRCP

- MRCPv2 supports voiceprint URLs only (i.e. not the voiceprint itself)
- For identification a list of voiceprint URLs or a URL identifying a group will be necessary
- Loading/storing of voiceprints should be implicitly done by V3
Voiceprint management: query/copy/delete voiceprints (Option 1)

- MRCPv2 does not provide all necessary administrative functions (e.g. COPY).
- Advantages option 1: administrative functions not executed by VoiceXML
- Disadvantage option 1: proprietary interface to voiceprint database.
Voiceprint management: query/copy/delete voiceprints (Option 2)

- MRCPv2 supports QUERY and DELETE commands
- Option 2: Reflect QUERY and DELETE at V3 syntax level
- Disadvantage option 2: admin functions executed via VoiceXML
W3C SIV Workshop.
SIV Architecture.

Embedded deployment supported by proposed architecture

- Usage of web protocols (HTTP/HTTPS) for voiceprint transport supports future deployment scenarios
W3C SIV Workshop.

Agenda.

- SIV Architecture
- Use cases
- SIV syntax
- Conclusion
Basic uses case #1: standalone SIV without ASR

Application
- Set User-ID = CLI
- Play welcome

Player resource
- "Welcome at ..."
- Welcome message

User

SIV resource
- "Say: My voice is my password"
- SIV Prompt 1
- "My voice is my password"
- Start SIV (+verif. sess.)
- Load voiceprint
- Verifying utt1

Verification session
- Start verification for "User-ID"
- Retrieve SIV results
- start second turn (if necessary)

Turn
Basic uses case #1: standalone SIV without ASR (cont’d)

- **Application:** Retrieve SIV results (accumulated), decision: accepted
- **Player resource:** "Please say it again"
- **User:** SIV prompt 2, "My voice is my password"
- **SIV resource:** Start SIV, Verifying utt2
- **Turn:** Verification session
- **Play back verification result:** "You have been successfully verified"
Basic uses case #1: standalone SIV without ASR (cont’d)

- SIV needs to implement speech detection/endpointing (like ASR)
- SIV needs to implement timeouts (like ASR)
- SIV should in this use case provide bargein functionality

- SIV may need multiple turns (within one SIV session)
- Author needs control of whether another turn is necessary or not (⇒ syntax)
Basic uses case #2: SIV + ASR

Application
- Play welcome
- Play prompt to ask for customer. no.
- Start ASR
- Retrieve ASR result and use as claimed id

Player resource
- „Welcome at...“
- „Please say your account no“
- „My account no is 1234567890“

User

SIV resource

ASR resource
- Load grammar
- Start ASR
- Recognize utt
### Basic uses case #2: SIV + ASR (cont’d)

<table>
<thead>
<tr>
<th>Application</th>
<th>Player resource</th>
<th>User</th>
<th>SIV resource</th>
<th>ASR resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start verification using claimed id</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Play prompt</td>
<td>Please say: My voice is my password</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start ASR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Retrieve ASR/SIV results, continue (if necessary)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Now say your personal phrase</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Verifying utterance 1</td>
<td>Start SIV (+verif. sess.)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Load voiceprint 1</td>
<td>Verify SIV utt1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Start SIV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Verifying utterance 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Load grammar 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Start ASR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Recognize utterance 2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Verification Session
- Turn
- "My voice is my password"
- "My dogs name is pfiffi"

**Deutsche Telekom Laboratories**
Basic uses case #2: SIV + ASR (cont’d)

- SIV may run in parallel to ASR (difference to use case #1)
- Idea: use ASR to make sure that the user repeated the correct (prompted) utterance
- Both ASR and SIV can return events like noinput etc. ⇒ application has to catch them

Issues:

- What if user repeated wrong utterance and ASR is used to check if SIV is not successful? ⇒ conclusion: undone/rollback functions necessary to remove latest utterance from cumulated result
- Problem if engine ended session by itself ⇒ conclusion: session has to be ended by app only
- Same problem if adaptation was enabled ⇒ rollback for adaptation necessary (supported by MRCP thru abort header for end-session method)
Basic uses case #3: ASR + SIV from buffer

Application
- Play welcome
- Play prompt to ask for customer. no.
- Start ASR (incl. buffering of user utt.)
- Retrieve ASR result
- Start verification from buffer using claimed id
- Play back verification result

Player resource
- "Welcome at ..."
- "Please say your account no"

User
- "My account no is 1234567890"

SIV resource
- Start SIV (+verif. sess.)
- Verifying utt from buffer

ASR resource
- Load grammar
- Recognize utt
- Buffering utt

Time
- Turn
- Verification session

Deutsche Telekom Laboratories
Basic uses case #3: ASR + SIV from buffer (cont’d)

- ASR must be able to buffer one (or more?) utterances for later verification
- Requires new ASR functionality (e.g. new attribute siv_buffer)
Basic uses case #4: ASR + SIV from file

- **Application**: Play welcome
- **Player resource**: Play prompt to ask for customer. no.
  - Start ASR
  - Start Recorder
- **User**: Retrieve ASR result
  - Start verification from file using claimed id
- **SIV resource**: Play back verification result
  - "You have been successfully verified"
- **ASR resource**: Start SIV (+verif. sess.)
  - Verifying utt
    - Load voiceprint from file
  - "My account no is 1234567890"
- **Recorder resource**: Start SIV (+verif. sess.)
  - Load voiceprint
  - "My account no is 1234567890"
- **Recorder resource**: Start Recorder
  - Record utt
- **Player resource**: Load grammar
  - Start ASR
  - Recognize utt
- **User**: "Please say your account no"
- **User**: "Welcome at ... "
- **Welcome message**: Welcome message
- **Recorder resource**: Start Recorder
  - Play prompt to ask for customer. no.
  - Start ASR
  - Start Recorder
- **User**: "My account no is 1234567890"
- **Recorder resource**: Start SIV (+verif. sess.)
  - Verifying utt
    - Load voiceprint from file
  - "You have been successfully verified"
Basic uses case #4: ASR + SIV from file

- Recorder resource running in parallel to ASR to record user utterance
- Verification of recorded utterance requires special parameter (WAV file reference for verification from file)
- Which audio-formats are supported?
W3C SIV Workshop.

Agenda.

- SIV Architecture
- Use cases
- SIV syntax
- Conclusion
W3C SIV Workshop.
SIV vs. ASR.

ASR
- ASR dialogs consists of one or more independent turns

```
field 1
I want a pizza

field 2
with cheese

field 3
Yes, onions too
```

turn 1   turn 2   turn 3

dialog

SIV
- SIV dialogs consists of one or more turns that are part of an enrollment/verification session

```
field 1
My account is ...

field 2
Yes, thats true

SIV 1
My voice is my pass.

SIV 2
My voice is my pass.

field 3
Transfer $2000 to...
```

turn 1   turn 2   turn 3   turn 4   turn 5

verification session
dialog
SIV sessions:

- Enrollment and verification IDENTIFICATION can be session based
- SIV engines often compute (internally) cumulative results when verifying several utterances (turns)

- MRCP provides Start-Session and End-Session methods
- Voiceprint-ID (given when session is started) defines which voiceprint to be trained or matched during the enrollment/verification session
W3C SIV Workshop.
SIV syntax.

Inputs for VoiceXML 3.0 SIV elements:
- Mode (enroll/verify/identify)
- SIV-ASR (SIV only, SIV+ASR)
- Adaptation (bool)
- Buffering (for <field>) and “useBuffer” for <siv>
- Req. phrase
- Decision threshold
- Timeouts, like ASR
- ID (voiceprint URL), WAV file reference for verification from file (file URL)
- Rollback

Administrative functions:
- Query/copy/delete function
W3C SIV Workshop.
SIV syntax.

Syntax option 1: Extend existing `<field ...>` element

- Example:

```
<field name="utt1" siv_type="verify" ...>
   <voiceprint src="voiceprint_url"/>
   <grammar src="speech_grammar"/>
</field>
```

- Advantage:
  - reuse of existing element

- Disadvantages:
  - increased complexity of `<field>` element
  - control of begin and end of SIV session not sufficient

- Comment
  - multiple fields may belong to a single SIV session and hence use the same voiceprint. Referencing the same voiceprint URL in subsequent `<field>` is redundant.
W3C SIV Workshop.
SIV syntax.

Syntax option 2: Create one new <siv> element

- Example:

```xml
<par>
  <siv name="utt1" type="enroll / verify / identify" ...>
    <voiceprint src="voiceprint_url"/>
  </siv>
  <field>
    <grammar src="speech_grammar"/>
  </field>
</par>
```

- Advantage:
  - no increased complexity of <field> element
  - clear separation of SIV and ASR syntax

- Disadvantages:
  - additional element necessary
  - control of begin and end of SIV session not sufficient
W3C SIV Workshop.
SIV syntax.

Syntax option 3: Create a new element for each of the 3 basic functions:

- Example:

<table>
<thead>
<tr>
<th>function</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>enrollment</td>
<td>&lt;enroll ...&gt;</td>
</tr>
<tr>
<td>verification</td>
<td>&lt;verify ...&gt;</td>
</tr>
<tr>
<td>identification</td>
<td>&lt;identify ...&gt;</td>
</tr>
</tbody>
</table>

- Advantage:
  - better control of meaningful combinations of attribute values
  - example: `<siv type="enroll" adaptation="true"...>` is not meaningful, whereas `<enroll>` would not have a adaptation attribute
W3C SIV Workshop.
SIV syntax.

Open issues:
- Control of begin/end of SIV session
- Session needs to be closed by application (to allow control of rollback)
- How to execute a rollback? Separate `<rollback>` element?
W3C SIV Workshop.
SIV results.

Training:
- more_data_needed [true, false]
- decision [accepted, rejected, undecided]
- score (0 ... 100, 50 = decision threshold)

Verification:
- more_data_needed [true, false]
- decision [accepted, rejected, undecided], cumulative and local
- score (0 ... 100, 50 = decision threshold), cumulative and local
- adapted [true, false]

Identification:
- more_data_needed and adapted like for verification
- array of decision, score and voiceprint-ID

⇒ These are core results, should be mandatory within VoiceXML 3.0
W3C SIV Workshop.
SIV results.

Additional results:
- Various vendors provide more results. Most of them are nice-to-have.
  ⇒ Could be optional within VoiceXML 3.0

Examples:
- valid [true, false] (is the utterance valid?)
- device [cellular phone, electret phone, carbon button phone]
- gender [male, female]
- matched (is gender and device type same as in training?)
- num_utterances (number of utterances)
- ...

⇒ Proposal: Collect list of results of existing technologies and generate list of mandatory results. Decide on whether optional results should be allowed
W3C SIV Workshop.
Agenda.

- SIV Architecture
- Use cases
- SIV syntax
- Conclusion
W3C SIV Workshop.
Other open issues.

The following issues have not been addressed here:

- Events: SIV might generate a “noinput” event, a combination of SIV and ASR leads to doubled or conflicting events
- Timeout parameters: Should SIV and ASR always use the same timeouts? Different resources (e.g. from different vendors) may behave inconsistently on the same timeouts.
Similarities and differences between ASR and SIV

- SIV and ASR share some similarities, but do also have a lot of differences (e.g. SIV session)

Detailed requirements / use case description necessary:

- VoiceXML 3.0 requirements document contains a very generic set of SIV requirements
- For a further discussion, a common understanding regarding use cases is necessary

Proposed next steps:

- Collect and describe use cases in detail, to achieve a common understanding
- Decide which use cases to support in VoiceXML 3.0 (and which not)
- Collect list of (mandatory) results and decide whether optional results will be allowed
- Compare with MRCP and decide what functionality from MRCP also to support in VoiceXML 3.0