New Standards for Speech Development

David Thomson
CTO, SpeechPhone

Voice Search 2008 (B204)
VoiceXML Tools Problems

- Non-VoiceXML applications are not portable
  (Vendor-neutral applies only to speech server)
- No standards for runtime data
- Vendor tools cannot be intermixed
- Application development is expensive
- Better tools = more, better applications
VoiceXML Forum Tools Committee

- Part of the VoiceXML Forum
- **Charter:** Make VoiceXML easier to use through better development tools.
- **Contributors:** AT&T, Chrysalis, France Telecom, Genesys, eLoyalty, Intervoice, Lumenvox, Nuance, Oracle, SpeechPhone, Voxeo, West, ….
- **1-hour calls:**
  - Wed 11am ET - Data Logging (alternate weeks)
  - Wed 12pm ET - Advanced Dialogs
  - Fri 11am ET - Grammars
VoiceXML Tools Working Groups

• Data Logging:
  A specification for logging runtime data.

• Metalanguage:
  A uniform format for application servers.

• Advanced Dialogs:
  Standard methods for non-FSM dialogues.

• Open Source Grammars:
  A set of high-quality ASR grammars.
VoiceXML Tools Working Groups

• Data Logging:
  A specification for logging runtime data.

• Metalanguage:
  A uniform format for application servers.

• Advanced Dialogs:
  Standard methods for non-FSM dialogs.

• Open Source Grammars:
  A set of high-quality ASR grammars.
Data Logging Specification Motivation

Problem
Runtime data format is proprietary, different for each platform, making analysis across vendors difficult.

Solution
Define a uniform specification with elements, structure, and syntax.
Status

• Draft specification published Aug. 20, 2007

• Draft available at www.vocexml.org/datalogging
What you’ll find at www.voicexml.org/datalogging

1. Data Logging Overview
   Introduction and high-level description

2. Session Log Annotation Markup Language Specification
   Defines the overall data logging model

3. Application Server Logging Specification (ASLS)

4. Automatic Speech Recognition Logging Specification

5. VoiceXML Browser Data Logging Specification
Sample code: Loading a grammar

```xml
<grammar-define-handler
  sl:start="1124126927613" sl:end="1124126927617"
  name="Global" rank="0"
  uri="http://server.example.com/globalgram.xml"
  format="SRGS_ABNF" lang="en-US"
  sl:handle-request="define-grammar-msg-1" outcome="success">
  <grammar-content>
    #ABNF 1.0 UTF-8;
    mode voice;
    tag-format <lumenvox/1.0>;
    root $MainMenu;
    $MainMenu = operator | customer service | main menu;
  </grammar-content>
</grammar-define-handler>
```
VoiceXML Tools Working Groups

• Data Logging:
  A specification for logging runtime data.

  • Metalanguage:
    A uniform format for application servers.

  • Advanced Dialogs:
    Standard methods for non-FSM dialogs.

  • Open Source Grammars:
    A set of high-quality ASR grammars.
Metalanguage Motivation

Problem
Applications written with Vendor X’s tools will not run on Application Server from Vendor Y.

Solution
Define a uniform format for representing a complete application
Metalanguage Vision

• Goal: Just as VoiceXML makes voice browsers interchangeable, a common metalanguage will make application servers interchangeable.
Metalanguage Roadmap

1. Database
2. To VoiceXML Browser
3. ML Code From Tools
4. Transaction Server
5. CCXML
6. IDL
7. Tag Interp. Engine
8. VXML code
9. VXML Browser
10. VXML Code From Tools
VoiceXML Tools Working Groups

• Data Logging:
  A specification for logging runtime data.

• Metalanguage:
  A uniform format for application servers.

• Advanced Dialogs:
  Standard methods for non-FSM dialogs.

• Open Source Grammars:
  A set of high-quality ASR grammars.
Advanced Dialogs Overview

- **Objective (initial):** Provide input to the metalanguage group on language structures needed for advanced dialogs
- **Advanced Dialog:** A *voice user interface specification that cannot efficiently be represented by a finite state machine.*
- **Our third Workshop on Advanced Dialogs is this Thursday at Voice Search.**
Advanced Dialog Criteria Examples

1. Branching point is determined by multivariate statistical computation
   - Fraud detection entering PIN
2. System requires sophisticated rules or massive amounts of logic
   - Plays chess
3. If the dialog is guided by goals, not by states.
4. User input or system prompts cannot be anticipated in advance
   - Online psychologist - no canned phrases
5. System learns through experience – e.g. rewrites the state machine
   - “Personal help” on IRS line
6. System tries to model what's in the user's head
   - Movie line uses demographic to select 1408, not Hairspray
VoiceXML Tools Working Groups

• Data Logging:
  A specification for logging runtime data.

• Metalanguage:
  A uniform format for application servers.

• Advanced Dialogs:
  Standard methods for non-FSM dialogs.

• Open Source Grammars:
  A set of high-quality ASR grammars.
Open Source Grammars Goals

**Objective:** Create a set of open-source, standard grammars.

**Motivation:**
1. Make it easier to develop speech applications.
2. Unify the human-machine interface so applications from different vendors have familiar behaviors.
3. VoiceXML 2.0 built-in grammar specification is incomplete.
Project Guidelines

1. Try to create a single grammar for each purpose.
2. Grammars are modular.
3. Use SRGS & SISR.
4. Start with contributions from industry.
5. Grammars are maintained under Sourceforge.
6. Grammars are free and available to everyone.
7. We do not specify the prompt.
8. We define reference grammars that embody the specification and are also useful for developers.
Grammars in Initial Release

- Boolean (yes/no)
- Time
- Date
- Digits (1-16 digits)
- Number (1-16 digits)
- Phone (10 digits)
- Currency (dollars)
- Credit Card (card number, expiration date)
  - With and without Luhn check?
- Social Security Number

VoiceXML built-in grammars
Existing Grammars

• VoiceXML 2.0
• Jakarta RDC Tag Library
• ASR vendors (Nuance, Lumenvox, etc.)
• Platform vendors (Voxeo, etc.)
• eLoyalty – probably first version
• Now: Open Source Grammars:
  
  sourceforge.net/projects/grammars
Open Issues

• How do we handle multiple versions?
  – (Languages, CPU, SRGS updates, application, improvements)
• How do we handle recognizer differences?
• How do we measure compliance?
• How do we handle platform dependence?
• SRGS/SISR have limitations.
For More Information

Questions:
David Thomson <david@speechphone.com>

VoiceXML Forum site
www.voicexml.org/dialogs
www.voicexml.org/datalogging
www.voicexml.org/grammars
Metalanguage Processing Flow

Meta language:
- SCXML (for control)
- Presentation tags
- XML (for data)
- Other resources

Offline Development Tools

Application Server
- Tag Library
- Metalanguage Interpreter
- SCXML
- Other stuff

VoiceXML w/SCXML
- <scxml>
  - <vxml>
    - <tagX>
    - </tagX>
  - </vxml>
- </scxml>

Tag Interp. Engine
- VXML Code
- <scxml>
  - <vxml>
    - </vxml>
  - </scxml>

VDXML + SCXML
- VoiceXML Browser

Database
- Transaction servers, web pages, humans, etc.

IDL=Interface Definition Language
Typical VoiceXML System

- Developer
- Data Log
- Metalanguage
- VoiceXML
- VoiceXML Browser
- Application Server
- Devel. Tools
- ASR/TTS Servers
- MRCP
- Grammars

The SpeechGroup
VoiceXML Forum
Metalanguage with VoiceXML tags

The diagram illustrates the flow of information within an application server. The process starts with ML code from tools, which is then interpreted by a Tag Interp. Engine. The interpreted code is then sent to the VoiceXML Browser. The application server also interacts with a database and transaction servers, web pages, humans, etc.
Case-Based Reasoning for Diagnostics

* Script Selection chooses a Solution Script if a case is activated. If no cases are activated, Script Selection takes a script from the Disambiguation Rules box.