MASH IT UP!

A speech mashup framework for multimodal mobile services

Giuseppe (Pino) Di Fabbrizio
pino@research.att.com

AT&T Labs – Research, Inc.

© 2009 AT&T Intellectual Property. All rights reserved. AT&T and the AT&T logo are trademarks of AT&T Knowledge Ventures.
Outline

• Background
• Migrating Speech to the Web
• Speech Mashups
• Prototypes/Demos
Background
Challenge

- Existing business models in the speech industry limit scale of development community
  - Companies need to buy/integrated/support speech technology software, hardware, and network connectivity

- **Our Goals:**
  - Reduce barrier-of-entry (e.g., cost, ease of access to core speech technologies) for innovators
  - Grow innovators pool – create new service opportunities
  - Grow demand for data services
Resources – Ubiquitous and Pervasive

- Network Connectivity
- Bandwidth
- Computing
Speech Mashup Vision

Grow development community; Enable Developers To Easily Create Voice-Enabled Services

Enable Best in Breed: Provide speech technologies and APIs, thus reducing high barrier-of-entry and enabling a new generation of service innovators

Benefit from Best in Breed Service Innovators: Many new services, many new vendors, increase customer base, new business models

Speech Mashup Clients

Speech API

Speech Mashup Manager

WATSON ASR

Natural Voices TTS
Application Enabling

- **Mobile**
  - Local business search ([http://www.speak4it.com](http://www.speak4it.com))
  - General internet voice search
  - Restaurant & hotel recommendations
  - Online ordering / transaction fulfillment

- **IPTV**
  - Movie on demand search
  - Media content search
  - Voice controlled DVR
  - Multimodal remote control

- **Multimedia Search/Indexing**
  - TV shows, movies, other video content

- **Security**
  - Voice Signature
Migrating Speech to the Web
Web 2.0 memes cloud as constructed by Markus Angermeier on 11/11/2005
Web 2.0 Main Concepts

• Web as Platform
  – Web services (SOAP, REST), web hybrids (mashups)

• Dynamic User Interfaces
  – Rich internet applications (AJAX)

• Social Networks
  – blogs, wikis, cms (content management systems)
Web Services

Definition
- Software system designed to provide interoperable service machine-to-machine interaction over the network

Technologies
- XML, SOAP, WSDL
  - SOAP (Service Oriented Architecture Protocol): a protocol for exchanging XML-based messages over the network (usually over HTTP/HTTPS)
  - WSDL (Web Services Description Language): an XML-based language for describing web service interfaces

Features
- **Discovery**: the requester contacts a service broker to locate the service provider (UDDI Universal Description Discovery and Integration – describes the service registry)
- **Description**: the requested retrieves the service description (WSDL Web Services Description Language) from the provides and typically creates a local API stub
- **Messaging**: Communication between requester and provider using SOAP messages
Representational State Transfer (REST)

Definition
• REST is a design pattern for implementing networked systems
• “Representational State Transfer is intended to evoke an image of how a well-designed Web application behaves: a network of web pages (a virtual state-machine), where the user progresses through an application by selecting links (state transitions), resulting in the next page (representing the next state of the application) being transferred to the user and rendered for their use.” (Roy Fielding, 2000[1])

Technologies
• HTTP and relative methods GET, POST, PUT, and DELETE

Features
• **Resource** - identified by a URL, and accessible through a resource representation
• **Resource representation** - XML/HTML/GIF/JPEG/etc.
• **Resource Types** - text/xml, text/html, image/gif, image/jpeg, etc.
• **Transport Mechanism** – XML, JSON

Asynchronous JavaScript and XML (AJAX)

- Allows to send an HTTP request in background and dynamically update the DOM without reloading the page
- Supported in most of the more popular browsers

Mashups (web hybrids)

- Mashups are **web applications** that leverage the compositional nature of public web services.
- A mashup is created when several **data** sources and **services** are "mashed up" (combined) to create a new services.
- Common **technologies** used in a mashup environment: SOAP, REST, AJAX, JavaScript, JSON (e.g., JavaScript Object Notation: lightweight data-interchange format), and various public web services (Google, Yahoo, Amazon, etc.).
- Service and data **aggregation** is typically done at the server level, but there is an increasing interest in providing web-based **composition engines** (Yahoo! Pipes, Microsoft Popfly, etc.).
Client-Side Mashups

“Mashup Styles: Client-Side Mashups” - Ed Ort, Sean Brydon, and Mark Basler, August 2007
Server-Side Mashups

GeoTracker

Server-Side Mashup Aggregators
Speech Mashups
Speech Mashups

- **Public API**
  - Speech resources accessible through simple **REST** interface (SOAP works as well) – no telephony
  - **Input**
    - HTTP header – User ID, ASR parameters / TTS parameters
    - HTTP body – User’s speech / SSML
  - **Output**
    - XML/JSON/EMMA documents describing the recognition results, confidence scores, etc. / Chunked speech with bookmarks

- **Transport**
  - **Server** side support for receiving/sending **speech packets over HTTP** *(progressive upload/download based on HTTP 1.1 chunking)*
    - similar to Real Time Streaming Protocol (RTSP), a session ID could be used to keep track of the session when needed
  - **Client** side support for sending/receiving **speech packets over HTTP**
    - using AJAX pseudo-threading in the browser, or any other HTTP client technology

- **Audio capture**
  - Fat browser clients (full DOM & Javascript support) – IE, Safari, Firefox
  - Thin clients - Java ME, .NET, Perl, Python, Cocoa

- **Speech Mashup Portal**
  - User management, resources control (disk quota, processing usage), logging, transcriptions
AT&T Speech Mashup Architecture

Mashup Server / Application Web Server

Mobile Clients

Speech Management Portal

Yahoo Maps

Movies Info

Yellow Pages

Speech Service Portal

Servlet Container

REST API

Client Mgr

Web Service Broker

User Account Management & Administration

Web Container

Watson Servers

Voice Icons Dictionaries

Natural Voices TTS Servers

OS/S Systems

• Acoustic Models Dictionaries
• Built-in Grammars
• Shared and Private Grammars

• Accounts
• Logging
• Speech Data
• Resource Usage
AT&T Speech Mashup Architecture

Mashup Server / Application Web Server

Yahoo Maps
Movies Info
Yellow Pages

Mobile Clients

Speech Management Portal

Speech Service Portal

Servlet Container

REST API
Client Mgr
User Account Management & Administration
Web Service Broker

Web Container

Watson Servers
Voice Icons Dictionaries
Natural Voices TTS Servers
OSS Systems

• Acoustic Models Dictionaries
• Built-in Grammars
• Shared and Private Grammars

Pages

Accounts
Logging
Speech Data
Resource Usage
AT&T Speech Mashup Architecture – Mobile Client Examples

Java ME

Mobile App

Audio Manager

REST Client

AT&T Provided

GUI Manager

Authentication & Security Manager

Captures the audio from the device in the native coding format

Communicates with the Watson ASR web service and other web applications

Manages the synchronization with Watson, audio capture and the GUI (DHTML)

Abstracts the device graphical interface. Includes maps rendering and caching

Safari

JavaScript/DOM

WebKit Audio Manager Plug-in

AJAX Client

JS Watson

AT&T Provided

Abstracts the device graphical interface. Includes maps rendering and caching
AT&T Speech Mashup Architecture – Natural Voices TTS
AT&T Speech Mashup Portal

Welcome to the Speech Mashup Manager web site. This is where you can sign up to use AT&T’s speech recognition technology in web and mobile applications.

You are logged in as: 
Currently selected application name: (change)

- Manage application
- Manage user info
- Web log
- Web/HTTP transactions
- Mashup (developer’s name)
- Sample Code (to accompany Developer’s Guide)
- Save the speech Mashup status, delete
- Update account information
- Login

Speech Mashup Manager

Speech Mashup (AT&T network)
AT&T Speech Mashup Portal

- User accounts management 3rd party application developers
- Direct access to speech mashups API for registered users
- Quota control (disk space and number of hits/day/account)
- Built-in set of basic grammars
- Web-based grammar compiler
- Grammars and language model sharing (public/private)
- Real-time data logger
- Utterance recorder and online transcription tools
- Offline APIs
DIY Speech Mashups
Three Simple Steps

1. Register for a portal account.
   http://service.research.att.com/smm/
   You’ll receive back an UUID for logging onto the portal and enabling the client to access WATSON servers.

2. Create and upload one or more grammars or use a built-in or shared grammar.
   Three grammar types are supported: the SRGS XML standard, WBNF (WATSON’s BNF version), and SLMs. Uploading the grammar (or a zipped file of multiple grammars) to the portal automatically compiles it.

3. Build a speech mashup client from existing examples.
   The client can be written in any suitable programming language (Java, JavaScript, or any other language) depending on the device. Three sample clients can be downloaded: a Java-based client for any Java ME mobile, iPhone native application client, and a Safari Mac OSX plugin.
# What do You Need to Know

<table>
<thead>
<tr>
<th>Speech mashup specifications</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Speech mashup portal</strong></td>
<td><a href="http://service.research.att.com/smm/">http://service.research.att.com/smm/</a></td>
</tr>
<tr>
<td><strong>Device support</strong></td>
<td>Java ME devices, iPhone, Safari browser (Mac®)</td>
</tr>
<tr>
<td><strong>Audio format</strong></td>
<td>AMR (Adaptive multi-rate coding)</td>
</tr>
<tr>
<td>(speech mashup portal)</td>
<td>AU (μ-law, 16-bit linear)</td>
</tr>
<tr>
<td></td>
<td>WAV (μ-law, linear)</td>
</tr>
<tr>
<td></td>
<td>CAF (μ-law, linear)</td>
</tr>
<tr>
<td></td>
<td>Raw audio (μ-law and 16-bit linear) is also supported, but requires you to provide the sample rate if other than 8000 Hz. (See page 32.)</td>
</tr>
<tr>
<td><strong>Output format</strong></td>
<td>XML (default), JSON, EMMA (recommended)</td>
</tr>
<tr>
<td><strong>Supported languages</strong></td>
<td>US English (en-us, the default) and the US dialect of Spanish (es-us).</td>
</tr>
<tr>
<td><strong>Grammar types</strong></td>
<td>XML (SRGS), WBNF (WATSON BNF), and SLMs (statistical language models)</td>
</tr>
<tr>
<td><strong>User ID (UUID)</strong></td>
<td>Allows access to portal and enables the speech mashup client to access WATSON servers. Obtained by registering at the portal.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Value</td>
</tr>
<tr>
<td>----------------</td>
<td>----------</td>
</tr>
<tr>
<td>uuid</td>
<td>string</td>
</tr>
<tr>
<td>resultFormat</td>
<td>string</td>
</tr>
<tr>
<td>appid</td>
<td>string</td>
</tr>
<tr>
<td>cmd</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>audioFormat</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>control</td>
<td>string</td>
</tr>
<tr>
<td>grammar</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>sampleRate</td>
<td>integer</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| platform       | string   | Optional (but recommended). Some text to identify the make, model, and version of the client’s hardware platform, e.g. “BlackBerry 8800”.
| client         | string   | Optional. Some text that identifies the client software version, e.g. “Hotel Finder 2.0”.                                                 |
| imei           | string   | Optional. If the client is a mobile device, clients may set this to the device’s International Mobile Equipment Identity code, to allow application |
|                |          | developers to distinguish individual users. For production applications, this parameter should not be used, or at least not without the user’s explicit |
|                |          | consent, because of privacy considerations.                                                                                                  |
Step by step example
0) service registration

Username:
Password:
Password (Confirm):
First name:
Middle initial:
Last name:
E-mail:

Please type the text in this image:

Submit
iPizza – step by step example

1) speech grammar

<START> = [hes] [preamble] (<pizza> | <appetizers> | <wings> | <drinks> | <desserts> ) | <topmenu>

hes = uh | um | ahh

preamble = hi | please | ([I want] to [order]) | I would like [a | to order | place an order]

<pizzaname> = <pizzaname> [pizza];

topmenu =
  _( "$='FeaturedItems'"  ) (featured items) _
  _( "$='Specialties'"   ) (specialties) _
  _( "$='SignatureCrust'" ) (signature crust) _
  _( "$='AppetizersSauces'" ) (appetizer and sauces|apetizers|sauces) _
  _( "$='WingStreetWings'" ) (wingstreet wings|wings) _
  _( "$='WingStreetWings'" ) (wingstreet sides|sides) _
  _( "$='DessertDrinks'"   ) (desserts and drinks| desserts | drinks) _

<pizzaname> =
  _( "$='spcPizzaMiasm'"  ) (pizza mia | [a] [pizza] [with] mozzarella cheese and cheddar )
  _( "$='specChesselovers'" ) ([a] [pizza] [with] cheese [lover's] )
  _( "$='specMeatLovers'"  ) (meat [lover's] | [a] [pizza] ([with] italian sausages|sausage|bacon) [topping] )
  _( "$='specPepperoni'"   ) ([a] [pizza] [with] pepperoni ([lover's ]|topping))
  _( "$='specVegieLovers'" ) (veggie [lover's] |...
iPizza – step by step example

2) upload and compile

The controls below allow for the manipulation of grammar files for your application. Files may be viewed, renamed, deleted or uploaded from your workstation. To upload multiple files, ZIP the files together and upload the zip file. The files will automatically be unZIPPed in the destination directory. Uploaded grammar files are automatically compiled, and this compilation log may also be viewed. Files listed in red have failed grammar compilation. You can also share or remove sharing on your grammar files, and provide descriptions for grammar files. Shared grammar files are listed in blue.

Grammar Context:

<table>
<thead>
<tr>
<th>File Name</th>
<th>Title</th>
<th>Shared Grammars</th>
<th>Built-in Grammars</th>
<th>Size (Bytes)</th>
<th>Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>pizza.wbnf</td>
<td>pizza</td>
<td></td>
<td></td>
<td>4995</td>
<td>2008-04-23 10:12:48</td>
</tr>
<tr>
<td>imet.wbnf</td>
<td>imet</td>
<td></td>
<td></td>
<td>1026</td>
<td>2008-04-28 13:27:40</td>
</tr>
</tbody>
</table>

Upload File: [Browse] (maximum allowed size is 50 Mbytes)
iPizza – step by step example

3) web frontend

```javascript
function onClick() {
    var speakButton = document.getElementById("speakButton");
    if (speakButton.innerHTML == 'Speak') {
        start();
        speakButton.innerHTML = 'Stop';
    }
    else
        if (speakButton.innerHTML == 'Stop') {
            stop();
            speakButton.innerHTML = 'Speak';
            var results = asr();
            // alert(results);
            var interpretation = processAsrResponse(results);
            processUserChoice(interpretation);
        }
}
</script>
</head>

<body onload="onLoad();">
    <embed name="audio" id="audio" type="audio/watson" width=1 height=1>
    <div id="info" name="info">TEST</div>
    <div class="toolbar">
```
Response Example

Extensible MultiModal Annotation (EMMA) Format

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <emma:grammar id="graml" ref="sma:grammar=en-us-date&amp;UID=[your own UUID]"/>
  <emma:model id="modell" ref="sma:file=en-us-date.xsd&amp;UID=[your own UUID]"/>
  <emma:info>
    <uuid>[your own UUID]</uuid>
    <watson>
      <version>watson-6.1.3655</version>
      <time>2008-08-31 13:01:32.338</time>
      <session_id>20080807-130131-00000652</session_id>
      <hostname>ss-2</hostname>
    </watson>
  </emma:info>
    <![CDATA["July thirty first 2 thousand 8"]]>  
  </emma:one-of>
</emma:emma>
```
### iPizza – step by step example

4) service usage / utterance transcription

<table>
<thead>
<tr>
<th>id</th>
<th>timestamp</th>
<th>filename</th>
<th>appname</th>
<th>field</th>
<th>grammar</th>
<th>reco</th>
<th>transcription</th>
</tr>
</thead>
<tbody>
<tr>
<td>14616</td>
<td>2008-06-25 17:13:07.000</td>
<td>20080625/audio-14616.amr</td>
<td>def001</td>
<td>business</td>
<td>pizza</td>
<td>pizza italian sausage</td>
<td></td>
</tr>
<tr>
<td>14617</td>
<td>2008-06-25 17:15:14.000</td>
<td>20080625/audio-14617.amr</td>
<td>def001</td>
<td>business</td>
<td>pizza</td>
<td>uh veggie pizza</td>
<td></td>
</tr>
<tr>
<td>14618</td>
<td>2008-06-25 17:15:23.000</td>
<td>20080625/audio-14618.amr</td>
<td>def001</td>
<td>business</td>
<td>pizza</td>
<td>Diet Pepsi</td>
<td></td>
</tr>
<tr>
<td>14619</td>
<td>2008-06-25 17:15:30.000</td>
<td>20080625/audio-14619.amr</td>
<td>def001</td>
<td>business</td>
<td>pizza</td>
<td>buffalo wings</td>
<td></td>
</tr>
</tbody>
</table>
Prototypes
Speak4it! - [video]

http://www.youtube.com/watch?v=OURZpqh-35A
iPizza – Retail Ordering - video

http://www.youtube.com/watch?v=I_HxZJg4fzw
iMOD – Multimodal Movies on Demand
Mobile Local Search
Summary

Speech Mashup – What is it

- Paradigm shift in technology/business model by integrating web services and speech technologies
- Publicly available web API for AT&T WATSON automatic speech recognition (speech to text) and AT&T Natural Voices (text to speech) synthesis
- “Voice-enabler” for mobile devices, including iPhones, Smartphones, or any network-enabled device with audio input (including PCs)
- Advanced multimodal interaction support by combining gestures and voice inputs
Questions