

11.4 Voice Browser & Multimodal Interaction

Dr. Max Froumentin, W3C



Voice Browsing & Multimodal Interaction at W3C

Outline

- New devices allow web access
- ... but they are *different*
- So far most new devices adapt themselves to the visual-page-point-click paradigm
- ... but not for long
- The web itself is going to change:
 - New browser architecture
 - New standards

New devices



New devices (2)



In more detail

- new output devices: Small screens, speech synthesizers, Braille devices
- new input devices: Electronic pens (handwriting), speech recognition, gestures
- new environments: low bandwidth networks, GPS, ambient noise, batteries
- The 3 items above are all *dynamic*
- Web pages are becoming Web applications

So what?

Each new device manufacturer builds its own browser to suit existing Web content.

- "Smart browsers"
- Server-side adaptation

Complex example

This works so far but let's look at something more complex: "google maps in my car."



...

Google maps in my car. I want to have my car navigation system use google maps.
Requirements:

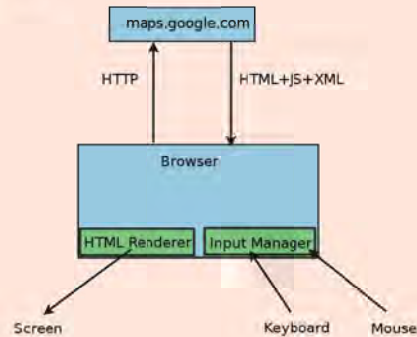
- I want to be able to click somewhere on the map (or touch the screen)
- I want to speak to it, and be able to ask it to repeat, or move to next item or previous itinerary.
- I want it to show me the screen when I'm stopped but to talk to me when I drive
- Use my car's GPS obviously, get traffic info, weather, etc.

Multimodal Interaction on the Web

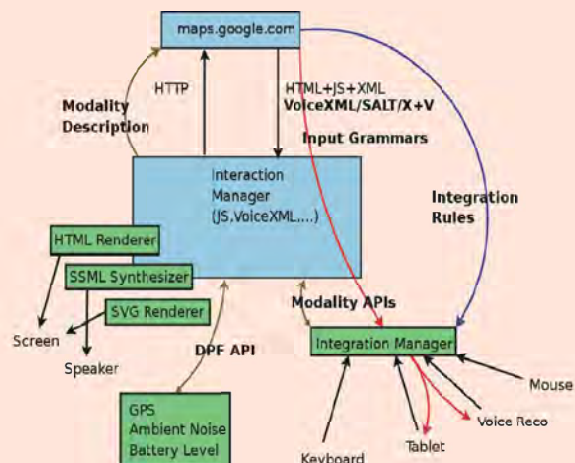
- New browsers that know what modalities they are connected to.
- New browser-server communication

Evolution of the browser architecture

Now:



The Multimodal Browser



What to standardise?

- The browser
- Web content

But why standardise things that are happening inside the browser?

Why change the Web?

The MMI framework



- the boxes don't necessarily map to devices
- Reuse of existing markup: XHTML, CSS, SVG for output, SRGS for input

Work items

- "connecting" things together: interfaces
- Input modalities: speech/text grammars (SRGS), handwriting recognition grammars: InKML
- Output modalities: reuse SVG, HTML, CSS, SSML, etc.
- Transmitting and merging input data: EMMMA, Compositing
- Interaction Manager: authoring languages
- System and Environment (now DCI)
- Sessions

Input Modality Interfaces

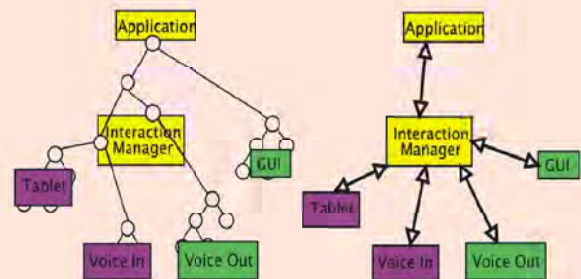
For input we need: Grammars, Integration, Interfaces

Interfaces as IDL/WSDL APIs can be used in Javascript directly, or Web Services, respectively

Generic: register, setGrammar, setModel, getData, prompt, pause, events

Specialised: sendVoiceXML

Leading to two types of architectures:



What to transmit through the pipes

- Web site to Browser: HTML, XBL, ES, VoiceXML (interaction management), PNG, SSML, SVG, CSS (output)
- Web site to Browser to Recogniser: Grammars, form information
- Recogniser to Browser to Web site: recognition results

Grammars: Speech Recognition

Speech Recognition Grammar Specification / Semantic Interpretation for Speech Recognition

```
<one-of>
  <item>Michael</item>
  <item>Yuriko</item>
  <item>Mary</item>
  <item>Duke</item>
  <item><ruleset uri="#OtherNames"/></item>
</one-of>

<one-of> <item>1</item> <item>2</item> <item>3</item> </one-of>

<one-of>
  <item weight="10">small</item>
  <item weight="2">medium</item>
  <item>large</item>
</one-of>

<one-of>
  <item weight="3.1415">pi</item>
  <item weight="1.414">root base</item>
  <item weight="1.25">scale</item>
</one-of>
```

Defining handwritten gestures and grammar: InkML

hello

```
<ink>
  <trace>
    10 0 9 14 8 28 7 42 6 56 6 70 8 84 8 98 8 112 9 126 10 140
    11 154 14 168 17 182 18 196 21 174 30 160 38 147 49 135
    58 124 72 121 77 135 80 149 82 163 84 177 87 191 93 205
  </trace>
  <trace>
    110 155 144 159 158 160 170 154 179 143 179 129 166 125
    152 128 140 136 131 149 126 163 124 177 128 190 137 200
    140 208 143 210 178 208 142 201 205 142 214 180
  </trace>
  <trace>
    217 50 226 64 225 78 227 92 228 106 228 120 229 134
  </trace>
  230 148 234 162 235 176 238 190 241 204
</ink>

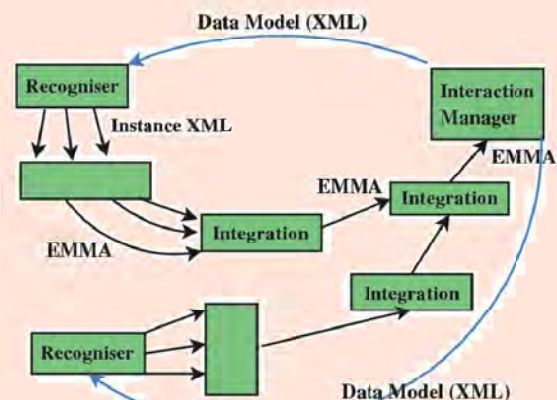
<trace>
  282 45 281 59 284 71 285 87 287 101 288 115 290 129
  291 143 294 157 294 171 294 185 296 199 300 213
</trace>

<trace>
  366 130 355 143 354 157 349 171 352 185 359 197
  371 204 385 205 398 202 408 191 413 177 413 163
  405 150 392 143 378 141 365 150
</trace>
</ink>
```

EMMA: representing and annotating input

Goals:

- define a markup language to combine user input information from recognisers to the interaction manager
- intermediaries add metadata to recogniser output



EMMA: example

```
<emma:emma emma:version="1.0"
  xmlns:emma="http://www.w3.org/2003/04/emma#">
  <emma:one-of emma:id="r1"
    emma:start="2003-03-26T0:00:00.15"
    emma:end="2003-03-26T0:00:00.2">
    <emma:interpretation emma:id="int1" emma:confidence="0.75">
      <origin>Boston</origin>
      <destination>Denver</destination>
      <date>
        <emma:absolute-timestamp
          emma:start="2003-03-26T0:00:00.15"
          emma:end="2003-03-26T0:00:00.2"/>
        03112003
      </date>
    </emma:interpretation>
    <emma:interpretation emma:id="int2" emma:confidence="0.68">
      <origin>Austin</origin>
      <destination>Denver</destination>
      <date>03112003</date>
    </emma:interpretation>
  </emma:one-of>
</emma:emma>
```

Compositing

Making one emma file out of two

- programming language (e.g. Java+DOM)
- XSLT

```
<xsl:if test="$emma:confidence > 40">
  <xsl:copy-of select="."/>
</xsl:if>
```

- Declarative: SMIL

```
<PAR>
  <SPOT>
    <ref id="input1" node="ink" grammar="select.ink" begin="activateEvent"/>
    <ref id="timeout1" dur="2s" begin="input1.activateEvent"/>
  </SPOT>
  <excl end="timeout2.end">
    <priorityClass peer="pause">
      <ref id="timeout2" end="timeout1.end"/>
    <ref id="speech1" node="speech" grammar="print.grm" begin="activateEvent"/>
  </priorityClass>
</excl>
</PAR>
```

The Dynamic Properties Framework



The DPF specification defines an API to access system properties. E.g.

- battery level
- signal strength
- latitude/longitude from GPS
- ambient noise
- user preferences

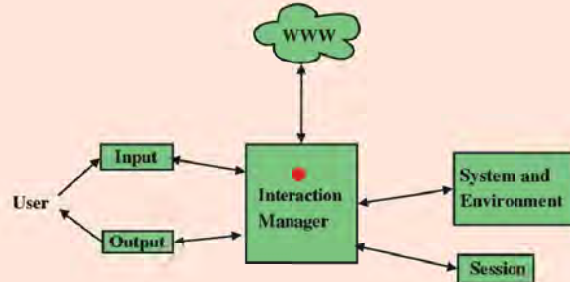
DPF: example

```
<html>
  <head>
    <title>GPS location example</title>
    <script type="text/javascript">
      <![CDATA|
        SystemEnvironment.location.format="zip code";
        SystemEnvironment.location.updateFrequency="20s";
      ]>
    </script>
    <script defer="defer" type="text/javascript">
      ev:event="se:locationUpdate">
        <![CDATA|
          var field = document.getElementById("location");
          var zipcode = SystemEnvironment.location;
          field.childNodes[0].nodeValue = zipcode;
        ]>
      </script>
    </head>
    <body>
      <h1>Track your location as you walk</h1>
      <p>Your current zip code is: <span id="location">(please wait)</span></p>
    </body>
  </html>
```

Interaction Manager

The manager...

- runs the web page or web application
- knows what modalities are available
- gets information from the DPF and Session components



Interaction Manager (2)

...and shapes the interaction accordingly:

- with visual media: shows the page on the screen
- with audio media: presents an application as a dialogue (à la VoiceXML)

Could be code: JavaScript using the APIs mentioned above, or declarative interaction markup like VoiceXML, with a mapping to API calls (with XBL)

Writing multimodal web content

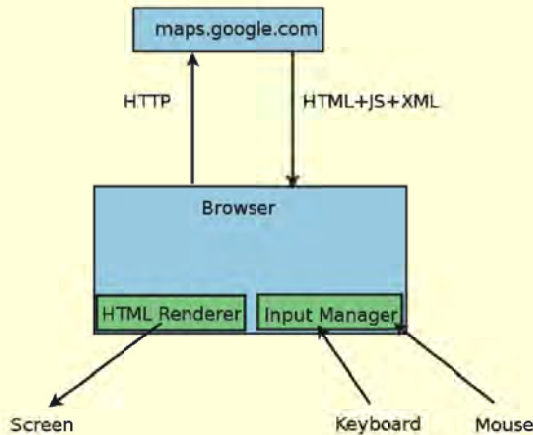
Existing web pages and applications will still work but won't provide:

- modality dependent: content
- modality dependent: interaction

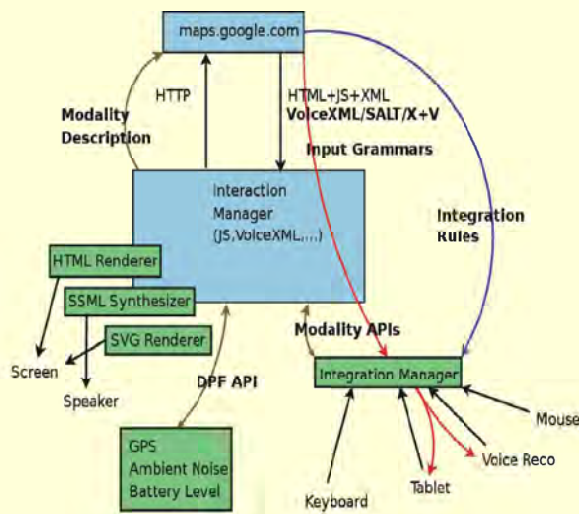
So extensions will be useful.

- You can already do it in HTML+JavaScript+MID+DPF (See above)
- Declarative markup (better): VoiceXML, SCXML, CCXML

Summary 1



Summary 2



The Voice Browser Activity

Historically precedes the MMI Framework

A specific framework

Now Integrates into MMI

VoiceXML

VoiceXML2: one of W3C's most successful specifications

Simple form-filling applications on the phone



```
<field name="adjustment amount">
  <grammar type="application/srgs+xml" src="/grammars/currency.grxml"/>
  <prompt>
    What is the value of your account adjustment?
  </prompt>
  <filled>
    <submit next="/cgi-bin/updateaccount"/>
  </filled>
</field>
</form>
```

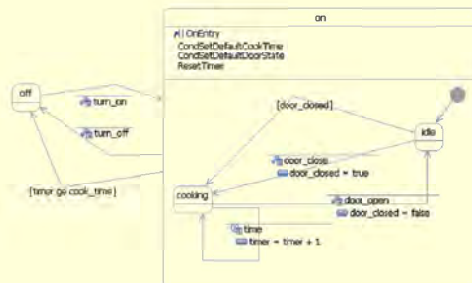
CCXML

a standard for telephony platforms

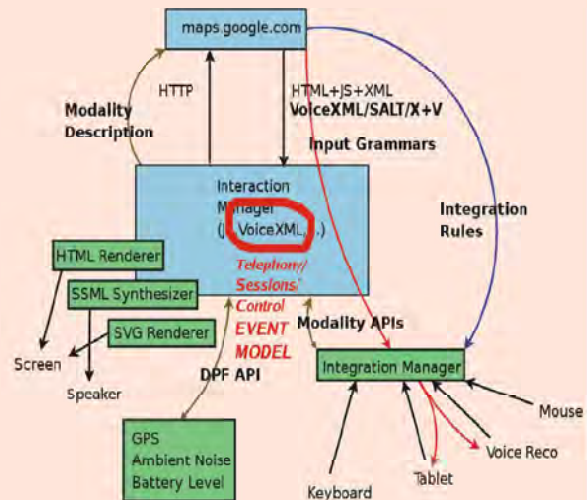
handles events (e.g. incoming calls)

makes outgoing calls, conference calls, start (VoiceXML) dialogues

Can plug in to CCXML, or drive VoiceXML dialogs or MMI interaction



- VoiceXML CCXML and SCXML provide: interaction manager and telephony capabilities
- But are mainly focused on voice applications, with telephony controls
- They integrate in the MMI framework and add new components



- Recognition
- Interaction
- New components: environment, events, integration, etc.