A NEW W3C MARKUP STANDARD FOR TEXT-TO-SPEECH SYNTHESIS

Mark R. Walker, Jim Larson
Intel Corporation

Andrew Hunt
SpeechWorks International

ABSTRACT

A new set of XML-based markup standards developed for the purpose of enabling voice browsing of the Internet will begin emerging in 2001 from the Voice Browser working group, recently organized under the auspices of the W3C. Among the first in this series of soon-to-be-released specifications is the speech synthesis text markup standard. The Speech Synthesis Markup Language (SSML) Specification is largely based on JSML [1], but also incorporates elements and concepts from SABLE [2], a previously published text markup standards, and from VoiceXML [3], which is itself based on JSML and SABLE. SSML also includes new elements designed to optimize the capabilities of contemporary speech synthesis engines in the task of converting text into speech. This paper summarizes the markup element design philosophy and includes descriptions of each of the speech synthesis markup elements.

1. INTRODUCTION

The Voice Browser working group has utilized the open processes of the W3C for the purpose of developing standards that enable access to the web using spoken interaction. This paper describes the nearly completed Speech Synthesis Markup Language (SSML) Specification [4].

The SSML specification is part of a new set of markup specifications for voice browsers, and is designed to provide a rich, XML-based markup language for assisting the generation of synthetic speech in web and other applications. The essential role of the markup language is to give authors of synthesizable content a standard way to control aspects of speech output such as pronunciation, volume, pitch, rate and etc. across different synthesis-capable platforms.

Previous work [1] has contrasted the use of XML as a structured markup language with traditional in-line control markup used in specifications such as SAPI 4.0 [5]. This paper summarizes the W3C Voice Browser SSML design philosophy and includes descriptions of each of the speech synthesis markup elements

2. APPLICATIONS AND AUTHORING OF SSML

SSML will enable a large number of applications by virtue of the fact that XML documents will be able to simultaneously support viewable and audio output forms. Email messages would potentially contain SSML elements automatically inserted by a synthesis-enabled, mail-editing tool that rendered the messages into speech when no text display was present. Web sites designed for sight-impaired Internet users would likely acquire a standard form, and would be accessible with a potentially larger variety of Internet access devices. Finally, SSML has also been designed to integrate with the Voice Dialogue markup standard in the creation of text-based dialogue prompts.

It is anticipated that authors of synthesizable documents will initially possess differing amounts of expertise. The impact of such differences may diminish as high-level tools for generating SSML content eventually appear. Some authors with little expertise will rely on choices made by the SSML processor at render time. Some document creators with higher levels of expertise will make considerable effort to mark as many details of the document to ensure consistent speech quality across platforms and to more precisely specify output qualities. Other document authors may demand the highest possible control over the rendered speech and will utilize synthesis-knowledgeable tools to produce “low-level” synthesis markup sequences composed of phoneme, pitch and timing information for segments of documents or for entire documents.

3. DESIGN CRITERIA AND GUIDING CONCEPTS

The following items were key design criteria during the design and standardization process:

Consistency: provide predictable control of voice output across platforms and across speech synthesis implementations.

Interoperability: support use along with other Speech Interface Framework specifications including (but not limited to) the Voice Browser Dialog Markup Language and Audio Cascading Style Sheets.

Generality: support speech output for a wide range of applications with varied speech content.

Internationalization: enable speech output in a large number of languages within or across documents.

Generation and Readability: support automatic generation and insertion of markup elements as well as hand authoring of documents. The resulting documents should be human-readable.

Ease of implementation: the markup specification should be easily implemented with existing, generally available technology. The number of optional features should be minimal.

4. SUMMARY OF MARKUP ELEMENTS

This section the SSML elements and attributes as they appear in the November 2000 working draft of the specification.

4.1 "speak" Root Element

The Speech Synthesis Markup Language is an XML application. The root element is "speak"

```
<?xml version="1.0"?>
<speak>
  SSML body ...
</speak>
```

4.2 "xml:lang"

Following the XML convention, languages are indicated by an "xml:lang" attribute on the enclosing element with the value following to define language and country codes.

```
<speak xml:lang="en-US">
 <paragraph>I don’t speak Japanese.</paragraph>
 <paragraph xml:lang="ja">Nihongo-ga wakarimasen.
 </paragraph>
</speak>
```

Language information is inherited down the document hierarchy. The speech output platform determines behavior in the case that a document requires speech output in a language not supported by the speech output platform according to common text formatting patterns of the language.

4.3 "paragraph" and "sentence" Elements

The "paragraph" element represents the paragraph structure in text. A "sentence" element represents the sentence structure in text. A paragraph contains zero or more sentences.

```
<paragraph>
 <sentence>This is the first sentence of the paragraph.
  </sentence>
  <sentence>Here’s another sentence.</sentence>
</paragraph>
```

The use of paragraph and sentence elements is optional. Where text occurs without an enclosing paragraph or sentence elements, the SSML processor should attempt to determine the structure.

4.4 "say-as" Element

The "say-as" element indicates the type of text construct contained within the element. This information is used to help disambiguate the pronunciation of the contained text. In any case, it is assumed that pronunciations generated by the use of explicit text markup always take precedence over pronunciations produced by a lexicon.

The **"type"** attribute is a required attribute that indicates the contained text construct. The base set of enumerated type values includes *acronym*, (contained text is pronounced as individual characters), *number*, *date*, *time* (time of day), *duration*, (temporal duration), *currency*, *measure*, (measurement), *telephone* (telephone number), *name*, *net*, (internet identifier), and *address*, (indicates a postal address).

```
<say-as type="acronym">
  USA </say-as>
<!-- U. S. A. -->

Rocky  <say-as type="number">XIII</say-as>
<!-- Rocky thirteen -->

Pope John the
  <say-as type="number:ordinal">VI</say-as>
<!-- Pope John the sixth -->

Deliver to
   <say-as type="number:digits">123 </say-as>
Brookwood.
<!-- Deliver to one two three Brookwood-->

<say-as type="date:ymd"> 2000/1/20 </say-as>
<!-- January 20th two thousand -->

Proposals are due in
<say-as type="date:my"> 5/2001 </say-as>
<!-- Proposals are due in May two thousand and one -->

The total is <say-as type="currency">$20.45</say-as>
<!-- The total is twenty dollars and forty-five cents -->

<say-as type="net:email">
  road.runner@acme.com
</say-as>
```

The **"sub"** attribute is a say-as attribute employed to indicate that the specified text replaces the contained text for pronunciation. This allows a document to contain both a spoken and written form.

```
<say-as sub="World Wide Web Consortium">
  W3C  </say-as>
<!-- World Wide Web Consortium -->
```

4.5 "phoneme" Element

The "phoneme" element provides a phonetic pronunciation for the contained text. The "phoneme" element may be empty. However, it is recommended that the element contain human-readable text that can be used for non-spoken rendering of the document. The **"ph"** attribute is a required attribute that specifies the phoneme string itself. The **"alphabet"** attribute is an optional attribute that specifies the phonetic alphabet. Phoneme alphabets currently supported by SSML include International Phonetic Alphabet (IPA), WorldBet, and X-SAMPA.

```
Well
<phoneme alphabet="worldbet" ph="h;&amp;l;ou>
  hello
</phoneme>
there!
```

4.6 "voice" Element

The "voice" element is a production element that requests a change in speaking voice. Optional attributes include **"gender"**, (gender of the voice to speak the contained text) with enumerated values *male*, *female*, *neutral*, **"age"**, taking on *(integer)* values, **"category"**, (indicates preferred age category of the voice) with enumerated values *child*, *teenager*, *adult*, *elder*, **"variant"**, (indicates a preferred variant of the other voice) which takes on value *(integer)*, and **"name"**, (a platform-specific voice name). The value of "name" may be a space-separated list of names ordered from top preference down.

```
<voice gender="female" category="child">
  Mary had a little lamb
</voice>

<!-- now request a different female child’s voice -->
<voice gender="female" category="child" variant="2">
  It’s fleece was white as snow.
</voice>

<!-- platform-specific voice selection -->
<voice name="Mike">I want to be like Mike.</voice>
```

When there is no voice available that exactly matches the attributes specified in the document, the results of inserting the voice selection element may be platform-specific. Voice attributes are inherited down the tree, including elements that change the language:

```
<voice gender="female">
  Any female voice here.
  <voice category="child">
    A female child voice here.
    <paragraph xml:lang="ja">
      <!-- A female child voice in Japanese. -->
    </paragraph>
  </voice>
</voice>
```

A change in voice resets the prosodic parameters since different voices have different natural pitch and speaking rates. The "xml:lang" attribute may also be used to request usage of a voice with a specific dialect or other variant of the enclosing language.

4.7 "emphasis" Element

The "emphasis" element requests that the contained text be spoken with emphasis (also referred to as prominence or stress). The synthesizer essentially determines how to render emphasis, since the nature of emphasis differs between languages, dialects or even voices. The optional **"level"** attribute indicates the strength of emphasis to be applied.

```
That is a <emphasis> big </emphasis> car!
That is a <emphasis level="strong">
  huge
</emphasis>bank account!
```

4.8 "break" Element

The "break" element is an empty element that controls the pausing or other prosodic boundaries between words. If the element is not defined, the speech synthesizer is expected to automatically determine a break based on the linguistic context. Optional attributes include **"size"** and **"time"**.

```
Take a deep breath <break/>  then continue.
Press 1 or wait for the tone. <break time="3s"/>
I didn’t hear you!
```

In practice, the "break" element is most often used to override the typical automatic behavior of a speech synthesizer.

4.9 "prosody" Element

The "prosody" element permits control of the pitch, speaking rate and volume of the speech output. The attributes include **"pitch"**, the baseline pitch for the contained text in Hertz, **"contour"**, which sets the actual pitch contour for the contained text, **"rate"**, the speaking rate for the contained text, **"duration"**, the time to take to read the element contents, and **"volume"**, the volume for the contained text in the range 0.0 to 100.0. Relative changes for any of these attributes above are specified as floating-point values. For the pitch and range attributes, relative changes in semitones are permitted: "+5st", "-2st". Since speech synthesizers are not able to apply arbitrary prosodic values, conforming speech synthesis processors may set platform-specific limits on the values.

```
The price of the  package is
<prosody rate="-10%">
  <say-as type="currency">$45</say-as>
</prosody>
```

The **"contour"** attribute is used to define a set of pitch targets at specified intervals in the speech output. The algorithm for interpolating between the targets is platform-specific. In each pair of the form (interval,target), the first value is a percentage of the period of the contained text and the second value is the value of the **"pitch"** attribute.

```
<prosody contour="(0%,+20)(10%,+30%)(40%,+10)">
  good morning
</prosody>
```

4.10 "audio" Element

The "audio" element supports the insertion of recorded audio files and the insertion of other audio formats in conjunction with synthesized speech output. The audio element may be empty. If the audio element is not empty, the contents correspond to the marked-up text to be spoken if the audio document is not available. The required attribute is **"src"**, which is the URI of a document with an appropriate mime-type.

```
<!-- Empty element -->
Please say your name after the tone.
  <audio src="beep.wav"/>
<!-- Container element with alternative text -->
<audio src="prompt.au">
  What city do you want to fly from?</audio>
```

The "audio" element is not intended to be a complete mechanism for synchronizing synthetic speech output with other audio output or other output media (video etc.).

4.11 "mark" Element

A "mark" element is an empty element that places a marker into the output stream for asynchronous notification. When audio output of the TTS document reaches the mark, the speech synthesizer issues an event that includes the required **"name"** attribute of the element. The platform defines the destination of the event. The "mark" element does not affect the speech output process.

```
Go from <mark name="here"/>
  here, to
<mark name="there"/> there!
```

When supported by the implementation, requests can be made to pause and resume at document locations specified by the mark values.

5. FUTURE STUDY

Several element types and issues are proposed for inclusion in future versions of the Speech Synthesis Markup Language. Readers are referred to the specification itself for the entire listing.

5.1 Other Phoneme Alphabets

All of the phoneme alphabets currently supported by SSML suffer from the same defect in that they contain phonemic symbols not specifically designed for expression within XML documents. The design of a new, XML-optimal phoneme alphabet is currently under study.

5.2 "lowlevel" Elements: Fine-Grained Acoustic-Prosodic Control

The "lowlevel" element has been proposed as a container for a sequence of phoneme and pitch controls. A lowlevel sequence is composed of **"ph"** (phoneme with duration) and **"f0"** (timed pitch target) elements. A lowlevel element may contain a sequence of zero or more "ph" and "f0" elements. Both the "ph" and "f0" elements are empty. The elements may be interleaved or placed in separate sequences. The "ph" element is specified with attributes that designate the phoneme symbol and phoneme duration. The "f0" attribute is specified with attributes that designate a pitch value target and a time offset from the previous pitch target.

It is anticipated that synthesis content authoring tools could automatically generate documents composed only of lowlevel sequences. These documents would be rendered into audio by low-complexity waveform generators. For this reason, compactness of the individual elements has been given priority over readability.

5.3 Intonational Controls

The existing specification supports many ways by which a document author can affect the intonational rendering of speech output. In part, this reflects the broad communicative role of intonation in spoken language: it reflects document structure, paragraph and sentence elements, prominence, and prosodic boundaries. Intonation also reflects emotion and many less definable characteristics. The specification could be enhanced to provide specific intonational controls at boundaries, and at points of emphasis. In both cases there are existing elements to which intonational attributes could possibly be added.

6. STANDARD CONFORMANCE CRITERIA

The conformance criteria in the SSML specification are derived from the general XML 1.0 specification [5], and are designed to ensure consistency and portability of SSML documents across disparate platforms. XML 1.0 defines the properties required of *well-formed* and *valid* XML documents. These criteria also apply to conforming SSML documents.

The other conformance criteria are less strictly specified. It is recommended, for example, that the SSML processor inform its hosting environment if an unsupported element or element form is encountered within the SSML document. A conforming processor also should produce some tangible output in response to each output-altering markup element present in an SSML document. The output should be generated in a manner that complies with the functional description of the element in the specification. Exceptions are allowed when a non-supported language is specified within an element, or a parameter is specified that exceeds local computing or rendering capabilities. In these cases, the behavior of the SSML processor is platform dependent.

7. SUMMARY

Widespread use of SSML may energize the development of new classes of speech-enabled applications, as well as new tools for authoring synthesizable content. It is also anticipated that the SSML specification will undergo changes in the future to reflect the practices of SSML content developers.

8. REFERENCES

[1] *Java Speech Markup Language Specification*, Version 0.5, Sun Microsystems Inc., August 28, 1997.

[2] “SABLE: A Standard for TTS Markup”, R. Sproat, A. Hunt, M. Ostendorf, P. Taylor, A. Black, K. Lenzo, M. Edgington**,** *Proceedings Intl. Conf. Spoken Language Processing*, Sydney, November, 1998.

[3] *Voice eXtensible Markup Language* (VoiceXML^{TM}), version 1.0, May 2000. http://www.voicexml.org/

[4] *Speech Synthesis Markup Language Specification*, M. Walker, A. Hunt, W3C Working Draft, Nov 2000. http://www.w3.org/TR/speech-synthesis

[5] *Microsoft Speech API*, version 5.0, Microsoft Inc., 1999.

[6] *Extensible Markup Language* (XML) 1.0, October, 2000. http://www.w3.org/TR/REC-xml