An XML Resource Definition for Spoken Document Retrieval

Contributors in Alphabetical Order
Germán Bordel, Arantza Casillas, Mikel Penagarikano, Luis J. Rodríguez-Fuentes, Amparo Varona
Grupo de Trabajo en Tecnologías Software (GTTS)
Universidad del País Vasco
{german.bordel, arantza.casillas, mikel.penagarikano, luisjavier.rodriguez, amparo.varona}@ehu.es

Abstract

In this paper, an XML resource definition is presented fitting in with the architecture of a multilingual (Spanish, English, Basque) spoken document retrieval system. The XML resource not only stores all the information extracted from the audio signal, but also adds the structure required to create an index database and retrieve information according to various criteria. The XML resource is based on the concept of *segment* and provides generic but powerful mechanisms to *characterize segments* and *group segments into sections*. Audio and video files described through this XML resource can be easily exploited in other tasks, such as topic tracking, speaker diarization, etc.

1. Introduction

Nowadays, finding multimedia (audio and video) resources is becoming as important as finding text resources. However, search engines are usually limited to adjacent texts (hand supplied transcripts or close captions) to index and classify multimedia documents. These texts are just short descriptions, shallow categorizations or partial transcriptions of the contents, so the resulting index is very coarse and the search cannot focus on specific items.

A key advantage can be taken from using Automatic Speech Recognition (ASR) and Natural Language Processing (NLP) technologies, since they allow to transcribe and enrich spoken documents, thus leading to more accurate indexes and more focused search results [1]. Some systems have been already developed in this way, most of them dealing with spoken documents in English, such as SpeechBot [2] (an experimental web-based multimedia search tool from HP Labs which was withdrawn in November 2005) and SpeechFind [3] (a spoken document retrieval system developed at the University of Texas, currently used to transcribe USA historic recordings from the last 110 years). In the last years, some systems have been developed which deal with other languages, such as the NTT system for Japanese [4] and the ASEKS system for Chinese [5].

We have designed a multilingual (Spanish, English, Basque) Spoken Document Retrieval (SDR) system that works with both audio and video resources [6]. In the case of video resources, only the audio signal is processed. The system consists of a sequence of processing agents, from the crawler that fetches the audio/video resource to the morpho-syntactic analyzer that adds information about word function and structure. The input to each agent is a resource descriptor containing all the information added by previous agents. The output is the same resource enriched with information specific to the current agent.

XML seems to be the best option to represent resource descriptors. XML document definitions related to speech have been previously proposed in several projects such as Transcriber, MATE and VoiceXML. Transcriber [7] is a tool for the transcription and annotation of speech signals which supports its own format, not specifically suited for spoken document retrieval applications. The MATE project [8] aimed to facilitate re-using language resources by addressing the problems of creating, acquiring, and maintaining language corpora. MATE designed a stand-off XML architecture to represent the information of spoken dialogue corpora at multiple levels: prosody, morpho-syntax, co-reference, dialogue acts, communicative difficulties and inter-level interaction. VoiceXML [9] was designed for creating audio dialogs that feature synthesized speech, digitized audio, recognition of spoken and DTMF key input, recording of spoken input, telephony, and mixed initiative conversations. Its major goal was to bring the advantages of web-based development and content delivery to interactive voice response applications.

The work presented in this paper involves designing an XML resource to store information from various knowledge sources, all of them relevant to the task of spoken document retrieval: URL, segmentation, audio type, language, speaker, speech transcription with spontaneous speech events, morpho-syntactic analysis, etc. A new XML resource is defined because none of the existing ones (Transcriber, MATE, VoiceXML, etc.) covers completely the requirements of our SDR system. With the aim to produce test data, we have developed a tool which translates Transcriber annotations into XML resource descriptors which can be processed by our SDR system.

The rest of the paper is organized as follows: section 2 briefly outlines the main features of our SDR system; section 3 describes the elements and attributes of the XML resource; finally, conclusions are given in section 4.

2. The system architecture

The SDR system fetches audio and video resources from internet or from local repositories, processing the audio signals and creating a collection of XML resource descriptors with information at various levels of knowledge. The SDR system also processes user queries and searches an index database to retrieve those resources matching the queries. The index database is periodically updated from the collection of XML resource descriptors. A web interface allows users to formulate queries and process the answers of the SDR system. The SDR system architecture consists of four key elements (see Figure 1): (1) the crawler/downloader; (2) the audio processing module; (3) the information retrieval module; and (4) the user interface.

The crawler/downloader fetches multimedia resources and creates the corresponding XML resource descriptors. Two different kinds of resources are considered: multimedia files obtained from internet and multimedia files stored at local repositories. Upon fetching, resources are uniquely identified by their SHA1 hash, which allows to discard copies of the same resource at different locations and avoid redundant processing. Audio

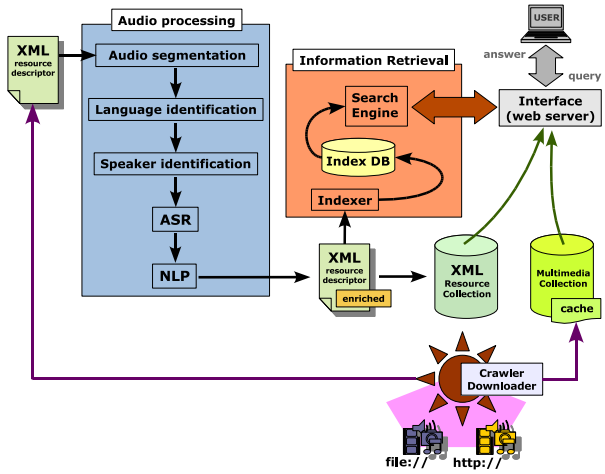

Figure 1: The spoken document retrieval system is organized around a collection of XML resource descriptors and consists of four key elements: a crawler/downloader, a sequence of audio processing modules, an information retrieval module (including an index database) and a web interface.

signals in PCM format are extracted from multimedia files. For presentation purposes, cached copies of the original multimedia resources are saved in Flash video format.

Audio processing is performed in several steps. At each step, the XML resource descriptor is enriched with information specific to a knowledge level. The identification of speech and non-speech regions in spoken documents is a key step: if non-speech segments are excluded from recognition, not only computation time is saved, but also better transcriptions are obtained. So, before applying the ASR engine, the audio input is divided into acoustically homogeneous regions called *segments*, which are further classified as speech and non-speech segments. Additionally, language identification is performed for speech segments. Identifying the target language is critical for the ASR engine to use adequate acoustic and syntactic models, and for the NLP module to apply the linguistic knowledge specific to that language. In particular, our SDR system deals with three languages: Spanish, English and Basque. Optionally, speaker identification may be performed. Speaker identification has always a positive impact on the accuracy of ASR, since it allows to apply speaker adaptation techniques. Once the speech segments, the language and (optionally) the speaker are identified, the ASR engine is applied to get the most likely word transcription according to previously estimated acoustic and language models. Finally, lemmatization and morpho-syntactic analysis of word sequences are applied, which allows to know the lemma, number, gender and case of each word. This information helps to index and search inflected forms and also to disambiguate between homonyms.

The information retrieval module takes the collection of enriched XML resource descriptors as input to create a hierarchized structure of word references which makes the search process easier and faster. The index structure, which contains location information for each word in the automatically generated transcriptions, is dynamically updated each time a new XML resource is added to the system. The information retrieval process begins when the user formulates a query. NLP tools are then applied to preprocess the query, yielding a list of query items (relevant words, topics or even speakers) which are searched within the index structure. The system retrieves those segments (or sequences of segments) matching the query items in the index database, and ranks them according to some predefined metric, which takes into account various measures.

The user interface is a web application based on Java Server Pages (JSP) accepting queries, sending them to the search engine, and receiving the list of matching items. This way, users can interact with the SDR system by using a standard web browser. The web application composes and presents successive HTML pages showing the list of matching items, with information about the resource name, location and size, segment boundaries (time stamps), links (thumbnails) to cached copies of the original multimedia resources and transcription excerpts which link to the full recognized transcriptions.

3. The XML document structure

The XML document structure was designed by means of XML Schema [10], taking into account the kind of data to be indexed and retrieved and the modules operating on them. The XML definition [11] is based on the concept of *segment* and provides generic but powerful mechanisms to: (a) *characterize segments*, and (b) *group segments into sections*. Each segment is characterized by a set of features and consists of a sequence of words, multi-words and acoustic events. Words and multi-words may also include phonetic, lexical and morpho-syntactic information. The root element, named <*resource*>, consists of a sequence of four consecutive elements: <*processors*>, <*source*>, <*segmentation*> and <*sectioning*>, which are described in detail in the following paragraphs.

3.1. The <processors> and <source> elements

These two mandatory elements include metadata describing where the audio and video resources were taken from and how they were processed, and key features that allow to play their contents (see Example 1). The <*processors*> element consists of a sequence of <*processor*> elements, each containing information about one of the agents that enriched the XML resource. The <*source*> element includes attributes specifying resource location and format-related features: URL, URI, cached file, cached audio, creation/download date and time, file size, audio length (in seconds), audio format, sampling rate, number of channels, coding (alaw, mulaw, linear) and resolution (bytes per sample: 8, 16, 32, 48).

3.2. The <segmentation> element

Audio segmentation can be done keeping in mind different objectives: discriminating speech from non-speech segments, identifying speaker turns, etc. So, our XML structure allows for the existence of one or more segmentations, each uniquely identified within the same XML resource. A segmentation consists of a sequence of one or more segments.

3.2.1. The element.

This element is characterized by: (a) three mandatory attributes: *id* (identifier), *offset* (start point, in seconds) and *length* (duration, in seconds); (b) a sequence of zero or more instances of the <*feature*> element; and (c) a sequence of at least one of the elements <*word*>, <*multiword*> and <*event*>, in any order (see Example 2).

3.2.2. The <feature> element.

This element provides a generic and flexible way to characterize segments. It has three attributes, all of them mandatory: *name* (feature name), *value* (feature value), *likelihood* (the likelihood of the feature value, a real value in the range [0,1]) and *processorName* (name or description of the agent extracting that feature from the audio signal). In this way, an arbitrary number

Example 1: The <*processors*> element includes information about the modules that have processed the multimedia resource. The <*source*> element stores metadata (location, format, size, etc.) useful for playing the audio and video contents.

```
<resource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://gtts.ehu.es/Ehiztari/erd.xsd">
  <processors>
      <processor name="downloader" date="Mon Jul 13 19:44:41 CET 2009"/>
      <processor name="Segmentator" date="Mon Jul 13 19:44:41 CET 2009" info="version: 0.6"/>
      <processor name="AudioTypeDetector" date="Mon Jul 13 19:44:42 CET 2009"/>
  </processors>
  <source uri="magnet:?xt=urn:sha1:20652dd426110f6c7e6d5a1b64"
          cache="/EhiztariSystemLocalSupervised/Source/20652dd426110f6c7e6d5a1b64.flv"
          pcm_cache="/EhiztariSystemLocalSupervised/pcmCache/20652dd426110f6c7e6d5a1b64.wav"
          date="1173116054" format="video/x-msvideo" size="1070916" length="12141"
          sampling_rate="48000" channels="2" code="linear" resolution="16">
      <url> http://gtts.ehu.es/Iberian_SLTech_2009_Example.avi </url>
  </source>
  ............
</resource>
```

of features can be extracted and assigned to audio segments: speaker, language, topic, etc. (see Fig. 2). Note that segments could be also characterized by defining the corresponding attributes (e.g. *speaker_id*, *language_id*, *topic*, etc.), but such an approach has two disadvantages: (1) the number and type of attributes is fixed, and (2) it does not provide any mechanism to specify how each feature was extracted.

3.2.3. The <word> element.

The <*word*> element is designed to represent all the information that can be extracted from a recognized word, through eight attributes: *transcription* (ortographic transcription), *pronunciation* (phonetic transcription), *confidence* (acoustic confidence, a real value in the range [0,1]), *offset* (start time, in seconds), *length* (duration, in seconds), *realization* (taking three possible values: regular, mispronounced and cutoff), *lemma* (canonical form of the lexeme) and *case* (grammatical function). Only the ortographic transcription is mandatory. However, three other attributes: *pronunciation*, *offset* and *length* can be easily obtained as a by-product from the recognizer. The recognizer should also be able to provide a confidence value for each hypothesized word, since confidence could play a fundamental role in ranking the search results. The default confidence is 1.0. Detecting cutoff or mispronounced words is not so easy. In fact, the attribute *realization* is there just for the case we were able to detect non-regular word realizations reliably. Its default value is *"regular"*. Finally, lemma and case, extracted by the NLP module (which takes into account both word transcriptions and context), play an important role in normalizing documents and queries before searching for matching segments.

3.2.4. The <multiword> element.

The <*multiword*> element allows us to handle expressions made up of several words, whose meaning cannot be derived from the meanings of the member words. Each multiword consists of a sequence of two or more <*word*> elements, plus additional information stored in five optional attributes. Three of them: *confidence*, *offset* and *length* can be derived from member words. The other two: *lemma* and *case*, are extracted by NLP tools.

3.2.5. The <event> element.

The <*event*> element may represent two kinds of phenomena: (1) those related to spontaneous speech and (2) those coming from external non-linguistic sources (environmental/channel noises). Two mandatory attributes are defined to specify the type of event: *category*, which can take four possible values: *noise*, *filled_pause*, *silent_pause* and *OOV_word*; and *subcategory*, which can take ten values: six noises (*breath*, *puff*, *cough*, *laugh*, *click* and *saturation*), three filled pauses (*e_long*, *a_long* and *m_long*) and the default value *other*. Three optional attributes allow to specify the start time (*offset*), the duration (*length*) and the acoustic confidence (*confidence*; default value: 1.0) of the event.

3.3. The <sectioning> element

Finally, the <*sectioning*> element provides a generic and flexible way to group segments into sections, according to any given criterion, specified by the attribute *criterion*. Since various segmentations might be available, the attribute *segmentation_id* tells what segmentation the segments are taken from. Besides these attributes, <*sectioning*> contains a sequence of one or more <*section*> elements. There can be various *sectioning* elements in an XML resource, corresponding to different clustering criteria: topic, language, speaker, etc. (see Fig. 2). Moreover, since there can also be $n > 1$ segmentations, the same criterion can be applied n times to different segmentations, yielding n different <*sectioning*> elements. This is an efficient way to access the same information from various points of view.

3.3.1. The <section> element.

Each <*section*> element is uniquely identified by the attribute *id* (mandatory). The attributes *seg_start* and *seg_end* (also mandatory) point to the first and last segments in the section, respectively. Optionally, the start time and the duration can be specified by the attributes *offset* and *length*. For any given criterion, a set of classes can be defined, segments tagged with the most likely class (by means of <*feature*> elements), and sections computed and characterized through class likelihoods. So, besides the above mentioned attributes, the <*section*> element contains a sequence of <*class*> elements.

3.3.2. The <class> element.

The <*class*> element has two attributes: *descriptor* (class descriptor) and *likelihood* (class likelihood). It tells how well any given section matches a predefined class.

4. Conclusion

In this paper, an XML resource has been presented fitting in with the architecture of a multilingual spoken document retrieval system. The XML resource not only stores information extracted from the audio signal (segmentation, transcription, etc.) but also adds structure which helps to create an index database and to retrieve information according to various criteria (keywords, topic, speaker, language, etc.). Around a

Example 2: The element consists of a sequence of features followed by a sequence of words, multiwords and/or events.

```
<segment id="34" offset="76.34" length="2.89">
    <feature processorName="langId" name="language" value="english" likelihood="0.95"/>
    <feature processorName="spkrId" name="speaker" value="s17" likelihood="0.41"/>
    <event category="noise" subcategory="breath"/>
    <word transcription="next" lemma="next"  case="IN"/>
    <word transcription="week" lemma="" case="NN"/>
    <word transcription="Michael" lemma="michael" case="NP"/>
    <word transcription="is" lemma="be" case="VBZ"/>
    <word transcription="going" lemma="go" case="VBG"/>
    <word transcription="to" lemma="to" case="IN"/>
    <event category="filled_pause" subcategory="m_long"/>
    <multiword lemma="new_york" case="NP">
        <word transcription="New"/>
        <word transcription="York"/>
    </multiword>
</segment>
```

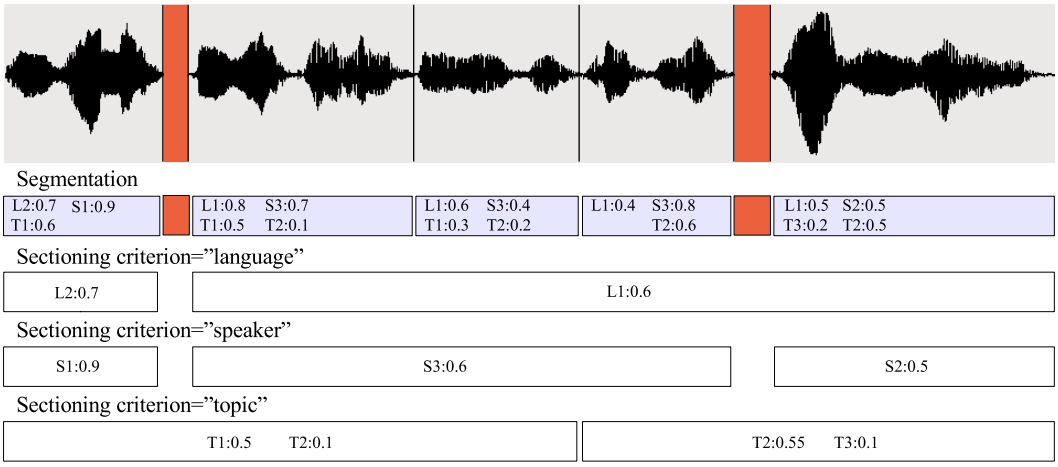

Figure 2: The <segmentation> element consists of a sequence of consecutive segments defined according to their acoustic contents (non-speech segments are shaded darker). Each segment can be characterized by an arbitrary number of features. For instance, the first segment in the example above is assigned the language L2, the speaker S1 and the topic T1, with likelihoods 0.7, 0.9 and 0.6, respectively. Segments can be grouped into sections according to various criteria. In the example above, three grouping criteria are showed: language, speaker and topic. Each section is characterized by a set of class likelihoods. For instance, in the grouping according to topics, the first section includes topics T1 and T2, with likelihoods 0.5 and 0.1, respectively.

core element called <*segment*>, two generic elements are defined: <*feature*>, to characterize segments, and <*sectioning*>, to group segments. The XML resource allows to handle an arbitrary number of segmentations and sectionings. At a lower level, each segment consists of an arbitrary number of words, multiwords and events, in any order. These latter elements account for phenomena related to spontaneous speech and external non-linguistic sources. Audio and video files described through this XML resource can be easily exploited in other tasks, such as topic tracking, speaker diarization, etc.

5. Acknowledgements

This work has been jointly funded by the University of the Basque Country, under project INFO09/29, and the Spanish MICINN, under project TIN2009-07446.

6. References

[1] J. Makhoul, F. Kubala, T. Leek, D. Liu, L. Nguyen, R. Schwartz, and A. Srivastava, "Speech and Language Technologies for Audio Indexing and Retrieval," *Proceedings of the IEEE*, vol. 88, no. 8, pp. 1338–1353, 2000.

[2] J. V. Thong, P. Moreno, B. Logan, B. Fidler, K. Maffey, and M. Moores, "SpeechBot: An Experimental Speech-Based Search Engine for Multimedia Content in the Web," *IEEE Transactions on Multimedia*, vol. 4, no. 1, pp. 88–96, 2002.

[3] J. H. Hansen, R. Huang, B. Zhou, M. Seadle, J. R. Deller, A. R. Gurijala, M. Kurimo, and P. Angkititrakul, "SpeechFind: Advances in Spoken Document Retrieval for a National Gallery of the Spoken Word," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 5, pp. 712–730, 2005.

[4] K. Ohtsuki, K. Bessho, Y. Matsuo, S. Matsunaga, and Y. Hayashi, "Automatic Multimedia Indexing," *IEEE Signal Processing Magazine*, vol. 23, no. 2, pp. 69–78, March 2006.

[5] R. Ye, Y. Yang, Z. Shan, Y. Liu, and S. Zhou, "ASEKS: A P2P Audio Search Engine Based on Keyword Spotting," in *ISM'06: Proceedings of the Eighth IEEE International Symposium on Multimedia*, San Diego, CA, USA, 2006, pp. 615–620.

[6] Hearch: http://gtts.ehu.es/Hearch/.

[7] C. Barras, E. Geoffrois, Z. Wu, and M. Liberman, "Transcriber: Development and use of a tool for assisting speech corpora production," *Speech Communication*, vol. 33, no. 1-2, pp. 5–22, January 2001.

[8] The MATE Project: http://mate.nis.sdu.dk.

[9] Voice Extensible Markup Language (VoiceXML) Version 2.0, http://www.w3.org/TR/voicexml20.

[10] W3C XML Schema: http://www.w3.org/XML/Schema.

[11] The EHIZTARI Resource Definition (ERD): http://gtts.ehu.es/Ehiztari/erd.xsd.