Augmented Auditory Representation of e-Texts for Text-to-Speech Systems

Gerasimos Xydas and Georgios Kouroupetroglou

University of Athens,
Department of Informatics and Telecommunications,
Division of Communication and Signal Processing,
Panepistimiopolis, Ilisia, GR-15784 Athens, Greece
{gxydas, koupe}@di.uoa.gr

Abstract. Emerging electronic text formats include hierarchical structure and visualization related information that current Text-to-Speech (TtS) systems ignore. In this paper we present a novel approach for composing detailed auditory representation of e-texts using speech and audio. Furthermore, we provide a scripting language (CAD scripts) for defining specific customizations on the operation of a TtS. CAD scripts can be assigned as well to specific text meta-data to enable their discrete auditory representation. This approach can form a mean for a detailed exchange of functionality across different TtS implementations. Moreover, it can be hosted to current TtS systems with minor (or major) modifications. Finally, we briefly present the implementation of DEMOSTHeNES Composer for augmented auditory generation of meta-text using the above methodology.

1 Introduction

In the near past, the research focus of Text-to-Speech (TtS) systems, concerning e-text handling, was mainly the extraction of linguistic information from rather plain e-texts in order to render prosody. During the last years, the format and the visual representation of e-texts have changed, introducing emerging issues in speech generation. Various e-text types (like HTML, MS-Word and XML) provide on the one hand text information in well-formed hierarchical structures and on the other optional information about the way the text will be visualized. Furthermore, in many other cases e-text has not a continuous flow with a proper syntactical structure, as the use of written telegraphic style has increased within documents. For example, an HTML document contains titles and legends in buttons and photos. Finally, a huge amount of e-text is stored in databases. Queries to these databases return different type of texts that have well-defined data structures (e.g. tables) but they lack syntactical structure.

Thus, e-text types described above carry a fair amount of text meta-data that is not related with semantics or pragmatics. These text meta-data are defined as *meta-text*. Therefore, meta-text can be format or visual oriented and lately speech oriented.

Emerging speech mark-up languages (like VoiceXML[1] and STML[2]) facilitate the transfer of speech information, by providing means (tags) for defining high-level prosodic behavior (or alternatively detailed low-level prosody in terms of pitch and time) and audio insertions.

V. Matoušek et al. (Eds.): TSD 2001, LNAI 2166, pp. 134–141, 2001.

However, the above speech related mark-up languages do not support adequately meta-text in order to fit appropriately to a TtS system. Firstly, the majority of existing e-text types are not in one of the above speech-aware formats. Secondly, even in speech mark-up languages there is not any kind of facilities to represent non-speech tags. Moreover, there is not yet a standardized method on interpreting speech tags and as a consequence this depends heavily on the view of each TtS system. And finally, the use of low-level prosodic description though detailed, is not optimal for cross-language use of the document (consider the case of the translation of an English text, formatted in VoiceXML document with detailed low-level pitch curve description, to a Czech one).

In this paper we present a novel approach for composing detailed auditory representation of e-texts using speech and audio. After the formulation of the major requirements for auditory representation of meta-text, we describe a general methodology for e-text to speech and audio composition. Next, we provide a scripting language (CAD scripts) for defining specific customizations on the operation of a TtS system. CAD scripts can be assigned as well to specific text meta-data to enable their discrete auditory representation. Finally, we briefly present the implementation of DEMOSTHeNES Composer for augmented auditory generation of meta-text using the above methodology.

2 Requirements for Auditory Representation of Meta-text

We have performed a short experiment to measure how people handle meta-text using their speech during reading out. Ten people were asked to read loudly the following document, as it is presented on a browser, to see how they will handle the structures involved using their speech.

```
<title>Speech Synthesis for Blind Persons</title>
<h1>Tools that convert text information to speech</h1>
<p>Text Readers enable users to <b>hear</b> – rather than read –
texts from a computer.</p>
<table><tr>
         <td>Institute</td><td>Description</td>
      </tr>
      <tr>
         <td>University of Athens, Department of Informatics</td>
         <td>DEMOSTHeNES Composer <a href =
"http://www.di.uoa.gr/speech/synthesis/demosthenes">Click
here.</a></td>
     </tr></table>
```

The results differed for each person, mainly because, for example, everyone tried to invent a way to describe the table structure. Some of them completely ignored the structure showed on the screen and read it without any variation in their speech. However, this document has a strongly defined hierarchical structure. If we read it as being a single line, we will hide all the structural information that it embeds. It is like showing this document in a browser without its format, but in a line. The latter additionally implies that the document will miss some of its meaning as well.

Visualization can provide some further information to support the vocalization of the text (e.g. bold letters usually imply emphasis). For enhancing the audible representation of e-texts, meta-text requires a different manipulation than ordinary text. Several works

have looked into ways to enhance the presentation of structures, such as tables, lists, etc., in cases of auditory-only interfaces [3]. They focus on the insertion of non-speech audio signals [4] such as beeps and on prosodic variations and speaker-style change [5]. Especially, the latter work shows how non-lexical cues inserted in a lexical string can enhance presentations with limited or no visual support.

An efficient audible representation of e-text requires the combination of synthetic speech, appropriately formed prosodic events and non-speech sounds. In order for a TtS system to accommodate these requirements, it should provide:

1. A mechanism for identifying meta-text in a document.
2. The appropriate data types and services for storing and manipulating the meta-text.
3. An efficient prosody and audio generator closely coupled with the meta-text.

Though the first requirement can be hosted in current TtS synthesizers in the form of an add-in module, the second and the third need major restructuring to be done, as there is not any provision for meta-text.

In order to describe the variety of information carried by meta-text that take place during speech generation and also the insertion of other non-speech sounds, we introduce the notion of *composing* rather than *synthesizing* when referring to such systems. Our approach essentially suggests a sub-system that enables the conversion to speech and audio from e-text. We call it *e-TSA composer*, or simply "composer" for the purposes of this paper. It applies to almost any open and modular TtS system with minor or major modifications (e.g. [6] and [7]). The e-TSA composer cooperates with the underlying TtS system (simply referred from now on as "TtS").

3 E-text to Speech & Audio Composer

Figure 1 presents the architecture and the flow of information in the e-TSA composer we propose. It consists of five major components (the "E-text Adapter", the "Transformer", the "Clusters Auditory Definition", the "Modules properties" and the "Composer") and a set of modules (for example the "HTML" and "Word" e-text adaptation modules) that customize its functionality. Black shapes ("Composer" and the text, speech and audio related modules) indicate strong coupling with the underlying TtS. The reasons for choosing this scheme accompanied by XML were mainly two: (a) the ability to hierarchically represent any type of meta-text and (b) the flexibility in writing scripts to customize the TtS functionality and vocalize e-texts.

The e-TSA composer is based on the eXtended Markup Language (XML) and its transformation capabilities. XML is an emerging document type that offers a text-based format for storing and transferring data.

The three main stages (namely "E-text to cXML", "cXML to ciXML" and "ciXML to S&A") will be described in the next paragraphs.

In [8] and [9], the XML has been used as a mean for internal representation and processing of hierarchical linguistic structures. However, they deal neither with non-linguistic structures nor with the interference of speech and audio signals. Furthermore, in the present work we use XML to organize and index meta-text under a hierarchical representation so that it is further processed.

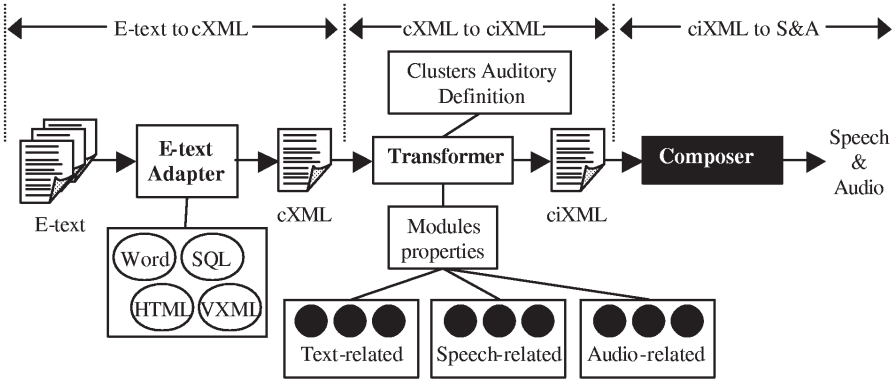

Fig. 1. The architecture of the e-TSA composer.

3.1 E-text to cXML

The first step involves adaptation of the source e-text to a hierarchical format that represents the structures of the source in a meaningful way for the composer, called composer-XML (named *cXML*). Thus, we can have a complete view of the elements that constitute the e-text (e.g. the tag `<table>`). We call these elements *clusters*. A cluster in the cXML format has a discrete XPath. This approach enables the user to decide how each cluster should be vocalized. Details are given in paragraph 3.2.

The "E-text Adapter" component offers services for importing, storing and interlinking text and meta-text from a number of e-text sources. Because of the variety of e-text types, several modules are provided to deal with each specific type (see Figure 1). For example, when dealing with MS-Word documents, the Word's instruction for "bolding" text can be represented in cXML with a `<bold>` tag that starts in an appropriate position and ends in its corresponding one. An appropriate module called "Word" implements this manipulation. If the document is already in any XML-like format (e.g. VoiceXML), it still needs some adaptation to map the source tags to user defined ones, according to the configuration that follows. Suitable modules handle this case as well.

3.2 cXML to ciXML

The second step in the composer is the production of a complete instruction set for the TtS. The target here is to have a scheme that can be recursively applied to any instance of a cluster, in order to generate the appropriate speech and audio. For example, one would like to apply a "staccato" pattern when reading the `<title>` in HTML pages or a "flat" pitch when reading the `<headline>`s in an e-newspaper.

To facilitate this we need a mechanism that will associate a customized speech and audio behavior to each cluster. This constitutes the Cluster Auditory Definition (CAD). A CAD consists of a collection of parameters throughout the modules of the TtS, an XPath description of the corresponding cluster and an ID of the instance (or the type) of the e-text it refers to. By having access to the functionality of the modules, we can

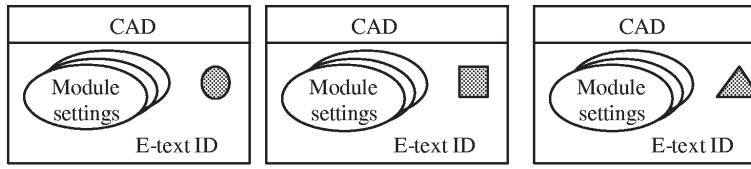

Fig. 2. A Cluster Auditory Definition (CAD) refers to dedicated meta-text with a customized auditory behavior.

apply different configurations to the specified cluster and optionally include non-speech sounds as well.

To make possible the implementation of a CAD, each module in the underlying TtS should provide a Document Type Definition (DTD), where it defines a unique namespace and the parameters that customize it [10]. For example, assume a module that handles rhythm in a TtS, providing options for applying several tempo patterns. This will define a `VMOD_RHYTHM` namespace and its DTD should contain:

```
<!ELEMENT Tempo EMPTY>
<!ATTLIST Tempo speed (lento|andante|allegro)>
<!ATTLIST Tempo accentual_variation (legato|staccato)>
```

CAD Scripts A script in a CAD must be able to make use of any service and data available in the TtS. It actually acts as a link between the cluster and a detailed programmed behavior of the TtS when dealing with this cluster.

We will demonstrate this by an example: assume the HTML document of paragraph 2 and consider what you would expect from a TtS. We can define seven different CADs. The auditory representation we describe lies on personal preferences (we will refer to them by their XPath):

1. `/title`: apply a “staccato” rhythm pattern upon vocalization,
2. `/h1`: form a single accent group during prosody generation,
3. `/p`: apply natural prosody,
4. `/table`: insert specific start/end beeps to indicate the table,
5. `/table/tr`: insert the word “row” to be pronounced in low-pitch,
6. `/table/tr/td`: insert word “column” in high-pitch and form a comma-ending phrase,
7. `/table/tr/td/a`: spell it loudly.

To create a script, we use transformation rules in XSLT [11] template formats. So, the template for the case of the CAD number 7 above (`/table/tr/td/a`) would be as follows:

```
<template match="table">
   <VMOD_SOUND:Insert file="hyperlink.wav"/>
   <VMOD_ENERGY:Loud>
   <VMOD_RHYTHM:Tempo accentual_variation="staccato">
```

```
      <apply-template select="//*/a"/>
   </VMOD_ENERGY:Loud></VMOD_RHYTHM:Tempo>
</template>
```

After applying the above template, the produced file embeds instructions for customizing the functionality of the TtS. The result is formatted in composer-interface XML type (named *ciXML*) and looks like the follow:

```
<VMOD_SOUND:Insert file="hyperlink.wav"></VMOD_SOUND:Insert>
<VMOD_ENERGY:Loud>
<VMOD_RHYTHM:Tempo accentual_variation="staccato">
Click here.
</VMOD_ENERGY:Loud></VMOD_RHYTHM:Tempo>
```

This template implies that whenever a link (`/a`) cluster is provided in the source HTML file, under a `/table` cluster, it will be vocalized using the behavior described in the template. In this way we implement the recursive feature described at the beginning of paragraph 3.2.

A template can host more complicated instructions concerning the available underlying modules. For example, we can define and parameterize the intonation model to be followed, how to handle different types of sentences presented in a CAD, etc. Actually, this approach constitutes a fully programming interface for the composer that can be shared among different TtS architectures, which adopt the e-TSA approach.

3.3 ciXML to Speech and Audio

The ciXML produced above needs to be further checked to ensure that it is well formed. This procedure confirms the validity of the referred namespaces and it is taking place during run-time. The interpretation of ciXML is left to the underlying TtS, though it offers a detailed description of the generation process. However, the TtS should provide means to parse it and use its own modules to further generate speech and audio. So, the basic modifications that are needed for a TtS to host the composer are:

1. To create DTDs for each module
2. To create an interface to parse the ciXML document

Another flexible feature of the composer's architecture is that a TtS can be configured to ignore any instructions which refer to modules it does not have, or to map the namespaces to local ones.

4 DEMOSTHeNES: A Complete Composer Implementation

Based on the requirements we set in paragraph 2 and the approach described above, we implemented DEMOSTHeNES Composer to accommodate classical TtS applications, but furthermore to facilitate the auditory representation translation of meta-text using the e-TSA composer architecture. We will briefly present some aspects of its implementation on hosting the composer [12].

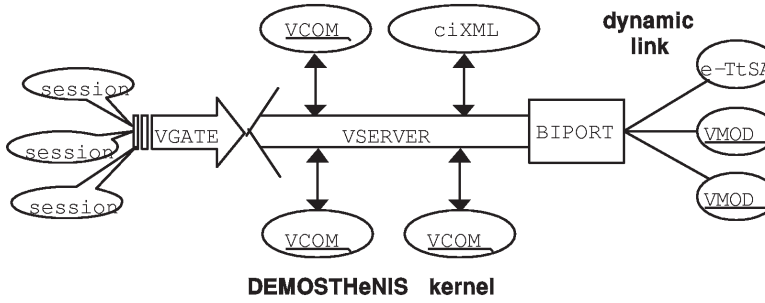

Fig. 3. DEMOSTHeNES architecture.

Its architecture (Figure 3) mainly benefits from its Component Based Technology (CBT) design. It comprises of a set of Vocal COMponents (VCOM) that provides services concerning basic text, speech and audio related tasks (e.g. manipulation of segmental duration, manipulation of linguistic information, etc.) and a Vocal SERVER (VSERVER). Dynamically linked Vocal MODules (VMOD) customize and expand the functionality of the kernel and the provided services. VMODs are connected to the VSERVER by a bi-directional port (BIPORT) that allows them to push and pull information from any VCOM.

In order to host the composer, we added a registration phase where each VMOD passes an appropriately formed DTD describing itself to the VSERVER. The VSERVER uses a Directory Service to register VMOD and to further locate any functionality described in a ciXML document. We built a VCOM that can translate ciXML documents to the internal structures of DEMOSTHeNES. We further configured it to ignore any unknown instruction in the ciXML. Finally, the “e-TSA” VMOD is dynamically linked to the kernel and implements the composer. For the purposes of this work we further implemented a module in the composer that translates HTML texts to cXML using the “E-Text Adapter” services. The composer, which uses the MBROLA synthesizer [13], is currently implemented in MS-Win32 platform and can offer vocal services to other applications.

5 Conclusions

In this paper we have introduced an approach for handling meta-text information. Meta-text is stored in an appropriate hierarchical structure and it is combined with customized functionality (described in XML) of the underlying TtS system, using XML-based scripts. The produced document forms a set of instructions that drive the TtS to generate the auditory representation of the meta-text. Finally, we showed how we exploit the above methodology to build a complete e-text to speech and audio composer.

Acknowledgements

The work described in this paper has been partially supported by the M-PIRO project of the IST Programme of the European Union under contract no IST-1999-10982.

References

1. Voice eXtensible Markup Language (VoiceXML^{TM}) version 1.0, W3C Note 05 May 2000 (2000), http://www.w3.org/TR/voicexml/
2. Sproat, R., Taylor, P., Tanenblatt, M. and Isard, A.: A markup language for text-to-speech synthesis, In Proceedings of Eurospeech97, Rhodes, Greece (1997) 1747-1750
3. Mitsopoulos, E.: A Principled Approach to the Design of Auditory Interaction in the Non-Visual User Interface, Submitted for the degree of Doctor of Philosophy, University of York, UK (2000)
4. Hakulinen, J., Turunen, M. and Raiha, K.: The Use of Prosodic Features to Help Users Extract Information from Structured Elements in Spoken Dialogue Systems, In Proceedings of ESCA Tutorial and Research Workshop on Dialogue and Prosody, Eindhoven, The Netherlands, (1999) 65-70
5. Shriver, S., Black, A. and Rosenfeld, R.: Audio Signals in Speech Interfaces, In Proceedings of International Conference on Spoken Language Processing (ICLSP-2000), Beijing, China (2000)
6. Taylor, P., Black, A. and Caley, R.: The architecture of the Festival Speech Synthesis System, 3rd ESCA Workshop on Speech Synthesis, Jenolan Caves, Australia (1998) 147-151
7. Dutoit, T., Bagein, M., Malfrere, F., Pagel, V., Ruelle, A., Tounsi, N. and Wynsberghe, D.: EULER : an Open, Generic, Multi-lingual and Multi-Platform Text-To-Speech System, In Proceedings of LREC'00, Athens, Greece (2000) 563-566.
8. Huckvale, M.: Presentation and Processing of Linguistic Structures for an All-Prosodic Systhesis System Using XML, In Proceedings of Eurospeech99, Budapest, Hungary (1999) 1847-1850
9. Horlock, J.: How Information is Extracted at Edinburgh, TeSTIA-2000, 8th ELSNET Eupopean Summer School on Languge & Speech Communication, Chios, Greece (2000)
10. Xydas, G. and Kouroupetroglou, G.: Text-to-Speech Scripting Interface for Appropriate Vocalisation of e-Texts, In Proceedings of Eurospeech2001, Aalborg, Denmark (2001)
11. XSL Transformations (XSLT), Version 1.0, W3C Recommendation 16 November 1999, (1999) http://www.w3.org/TR/xslt
12. Xydas, G. and Kouroupetroglou, G.: DEMOSTHeNES Composer, Technical Report, University of Athens, Athens (2001)
13. Dutoit, T., Pagel, V., Pierret, N., Bataille, F., Van Der Vreken, O.: The MBROLA Project: Towards a Set of High-Quality Speech Synthesizers Free of Use for Non-Commercial Purposes, In Proceedings of ICSLP'96, Philadelphia, vol. 3, (1996) 1393-1396