

Creation of Speech Corpora for the Multilingual Bonn Open Synthesis System

Esther Klabbers[†], Karlheinz Stöber^{*}

[†]IPO, Center for User-System Interaction, Eindhoven, The Netherlands

E.A.M.Klabbers@tue.nl

^{*}IKP, University of Bonn, Germany

kst@ikp.uni-bonn.de

Abstract

In this paper we present the procedure for creating a new speech corpus for the Bonn Open Synthesis System (BOSS). BOSS has several advantages which make this procedure particularly straightforward and fast. BOSS is open source, allowing flexible use of components and corpora. It shows a clear separation between data and architecture, which means that a change in corpus does not require a change in the architecture. The data formats are strictly defined, making it a very transparent system. The implementation of a small Dutch corpus is used as a case study.

1. Introduction

The Bonn Open Synthesis System (BOSS) is a new open source architecture for unit-selection-based synthesis [6]. It is developed in C++ for the Linux platform. BOSS is a complete redevelopment of the Verbmobil synthesis software [9], allowing for the on-line selection of variable-sized units such as phonemes, diphones, syllables and words.

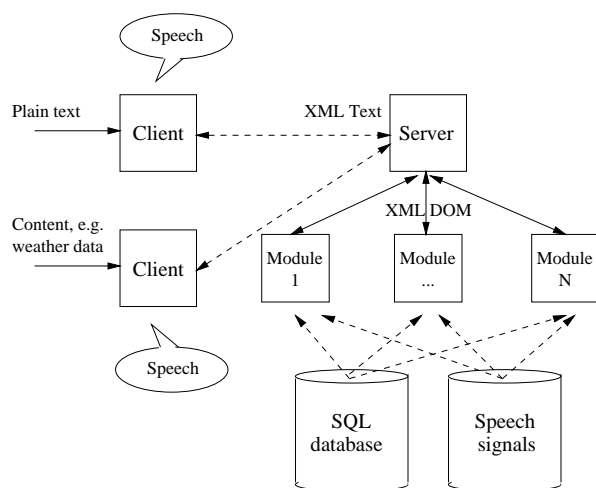


Figure 1: Overview of the architecture of BOSS.

There are several unit-selection systems available today [1, 3, 10]. We can identify several problems in existing unit-selection systems. Firstly, some systems have spe-

cific requirements concerning hardware or operating system, which can hamper their use in small devices. BOSS employs a small client program which can be developed for any operating system on any platform to contact the BOSS server remotely. An additional feature of the client is that it can not only send plain text to the server, but also provide additional information about the text when available. As an example, consider the case where the synthesis is coupled to a natural language generation module, which can generate accent and phrase boundary locations much more reliably than when these are computed from plain text [4, 8].

Secondly, most commercial systems provide only one voice per language and make you pay for additional languages should you require them. BOSS allows the easy creation of as many corpora as you like, either for different languages, different application domains or just different voices.

Thirdly, some systems are not very transparent, making the process of creating a new corpus a very troublesome one. In BOSS, the data and architecture are clearly separated and the data formats are well defined. Even inexperienced users of speech synthesis software can integrate new corpora easily without having to understand the underlying synthesis software. Thus, BOSS can be used for commercial purposes as well. In this paper we will show that the implementation of a new corpus is very straightforward and fast.

Figure 1 depicts the architecture of BOSS. The architecture is split up into a client and a server part. The client sends input to the server and receives the speech signal. The server contains the actual synthesis software. It consists of separate modules, which are independent of each other. Each module realises a synthesis step (e.g., transcription, unit selection, synthesis). The communication between server and modules takes place in a fixed format. Thus, modules can be developed and tested separately.

For BOSS to work with a new corpus, only the speech files (Section 2) and the SQL data (Section 3) have to be changed. The SQL data bases are created from the label files that accompany the speech data. In Figure 2 this process is explained. The boxes contain the utilities that were

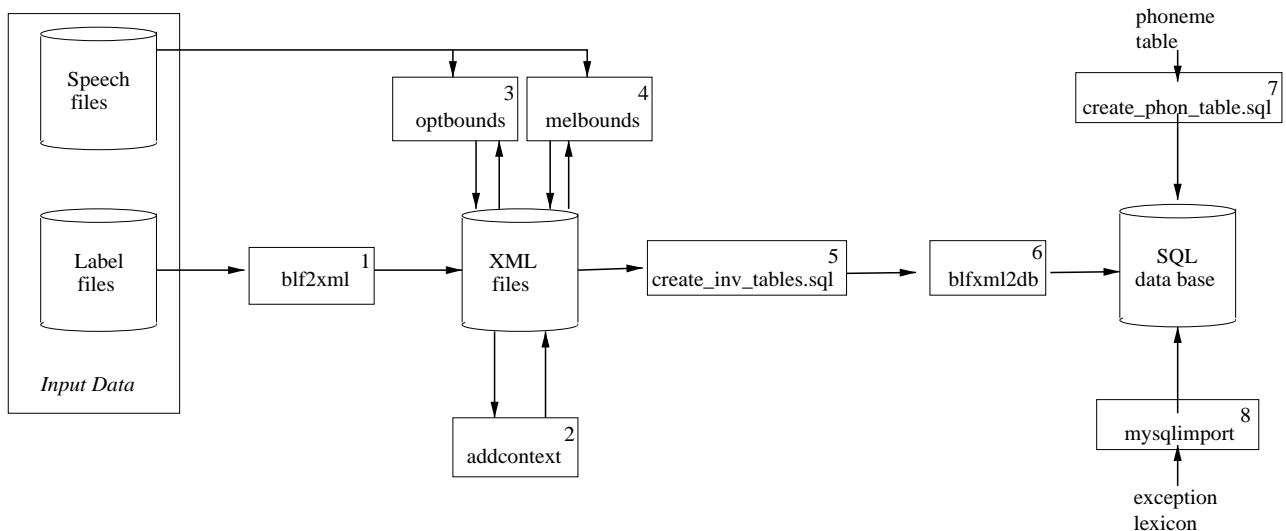


Figure 2: Overview of steps in creating a new corpus.

created to automatically transform the data.

2. Speech data

The speech data base is a collection of speech files in raw 16 KHz format. Each sentence is stored in a separate file. The original German Verbmobil corpus contains approximately 4500 sentences in the domain of hotel bookings and meeting scheduling.

In this paper, the implementation of a Dutch corpus is presented. This corpus was not specifically recorded for unit-selection-based synthesis. In fact, it was collected for the purpose of duration modelling [5]. It is only meant as

a demonstration of the ease of development of synthesis for new languages. At that time no larger corpus for Dutch with segmentation information was available. The corpus consists of 297 sentences containing all Dutch phonemes in a wide variety of phonetic and prosodic contexts. The sentences are taken from news items and are all declarative.

3. SQL data

The SQL data bases are derived from the label files that accompany each speech file. A number of steps have to be performed to integrate the label files into the SQL data bases (Figure 2). The steps will be discussed below starting with the format of the label files.

3.1. Label files

Each speech file has a corresponding label file (see Figure 3). The structure of a label file is line oriented and very similar to the Phondat II format, except that there is no header. We call this format BLF (BOSS Label File). The first column lists the start position of each phoneme label. The second column lists the phoneme label plus additional information. The additional information is used to mark word boundaries (.), syllable boundaries (.), syllable stress (“), phrase boundaries of different strength (1-5), and sentence type (declarative (.) or interrogative (?)) behind the phrase boundary indicator. For the Dutch corpus, the speech files were labelled manually using Waves+. Afterwards, a small tool was written to convert the label file into the desired format.

Pronunciation variations can also be coded in the label file. Deletions are marked by a minus sign (-) behind the phoneme label, substitutions are marked by the minus sign followed by the substitute phoneme. Insertions are marked by a plus sign (+) behind the label.

```

2056      _z
3109      @
4172      _x
5200      a
7470      n
7965      _?E
10079     r
10500     _"?O
12759     n
13889     .d
14344     @
15069     r
15414     .z
17803     u
19063     k
20063     _d 5,.
21159     u
22735     n
25197     -

```

Figure 3: Example label file.

```

<SENTENCE Type=".">
  <WORD TKey="z@" TReal="z@" PInt="" PMode="" First="2056" Last="4172">
    <SYLLABLE TKey="z@" TReal="z@" Stress="0" First="2056" Last="4172">
      <PHONEME TKey="z" TReal="z" Stress="0" First="2056" Last="3109"></PHONEME>
      <PHONEME TKey="@" TReal="@" Stress="0" First="3109" Last="4172"></PHONEME>
    </SYLLABLE></WORD>
  <WORD TKey="xan" TReal="xan" PInt="" PMode="" First="4172" Last="7965">
    <SYLLABLE TKey="xan" TReal="xan" Stress="0" First="4172" Last="7965">
      <PHONEME TKey="x" TReal="x" Stress="0" First="4172" Last="5200"></PHONEME>
      <PHONEME TKey="a" TReal="a" Stress="0" First="5200" Last="7470"></PHONEME>
      <PHONEME TKey="n" TReal="n" Stress="0" First="7470" Last="7965"></PHONEME>
    </SYLLABLE></WORD>
  ...
</SENTENCE>

```

Figure 4: Part of the XML file corresponding to the label file in Figure 3.

The label files are then converted into XML format using the *blf2xml* utility. Additional information is then added to these XML files by the different utilities.

3.2. XML files

Part of the XML file corresponding to the label file in Figure 3 is provided in Figure 4. Pronunciation variation is included by making a distinction between the fields *TKey* and *TReal*, where *TKey* represents the intended transcription and *TReal* the actual pronounced transcription. The fields *PInt* and *PMode* depict the boundary strength and sentence type. *First* and *Last* represent the segment positions in the speech file.

Before these XML files are converted into SQL data bases, the following additional information is added or modified:

- Context information: For each phoneme, syllable and word, the phoneme occurring on the left and on the right is included in the fields *CLeft* and *CRight*. This is done using the *addcontext* utility.
- The segment positions are adjusted to fall on the nearest (positive going) zero crossing. The *optbounds* utility performs this action.
- Twenty Mel-cepstral coefficients are added for each segment boundary to allow computation of spectral similarity in the unit selection process. This is done by the *melbounds* utility.

3.3. SQL data bases

Now the XML files are ready to be transformed into SQL data bases. These data bases are queried by the BOSS server during the unit selection process. Three data bases are created: a *word data* table containing all *WORD* entries in the XML files, a *syllable data* table containing all *SYLLABLE* entries in the XML files, and a *phoneme data* table containing all *PHONEME* entries.

An additional table is needed containing the list of phonemes for the language in question. The phoneme table is read into the SQL data base using the *create_phon_table.sql* option. This action only needs to be performed when a new language is introduced. For new voices or application domains the same phoneme table can be used. Transcription is currently performed by means of lexical lookup. The lexical lookup table is also stored as an SQL data base, by using the *mysqlimport* command. For Dutch, we have included the CELEX lexicon, containing approximately 120,000 entries. In the future, the transcription module has to be extended to handle words not present in the lexicon. This lookup table is language-specific, but it may be desirable to include application-specific lexicons.

It is possible to add any type of information, simply by adding a new attribute to the XML tags contained in the existing XML files. For their integration the SQL data base structure has to be extended and must have a column with the same name as the XML attribute. The new items can then be used in the unit selection without changing the communication architecture but the cost functions have to be changed to use the new features

4. Unit selection

So how is the data used in unit selection? The unit selection algorithm consists of two stages. In the first stage, available units are selected on the basis of the target segmental structure, the phonetic context and the unit location within the phrase (*candidate selection*). In the second stage, a final selection is made (*unit selection*). The basic building block in BOSS is the *word*. The incoming transcription is matched against the word data in the SQL data base using the following SQL query:

```

SELECT * FROM <inventory>_word_data WHERE
TKey="trans_key" AND PInt="pros_int";

```

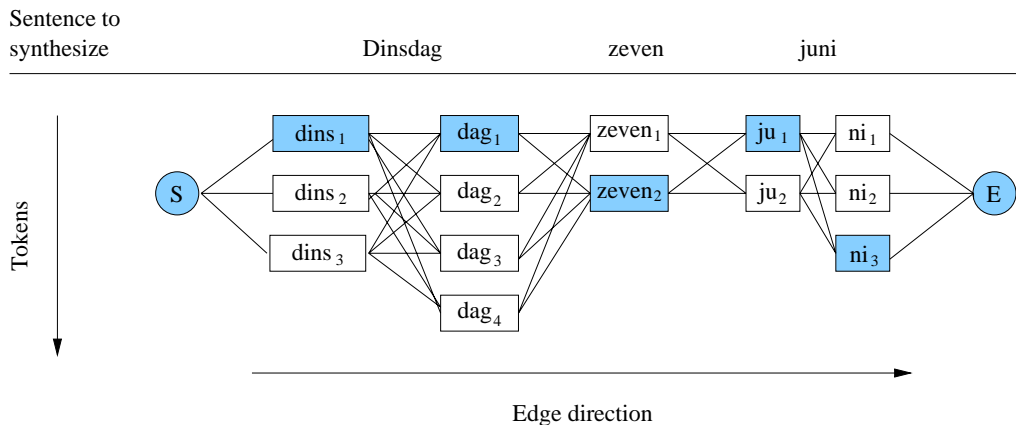


Figure 5: Example graph for the sentence “dinsdag zeven juni” *Tuesday June 7th*.

If a word is not available in the corpus, units are searched at a lower level, in this case the *syllable* level:

```
SELECT * FROM <inventory>_syllable_data
WHERE TReal="trans_key" AND CLeft="cont_left"
AND CRight="cont_right";
```

If no matches are returned the search criterion is loosened using:

```
SELECT * FROM <inventory>_syllable_data
WHERE TReal="trans_key" AND (CLeft="cont_left"
OR CRight="cont_right");
```

If a syllable is not available, search continues at the *phoneme* level, the lowest level. The queries are similar to those for syllable selection, except that the `<inventory>_phoneme_data` table is searched.

Then, in the unit selection stage, a final selection is made. For each possible unit, a set of possible candidates is present in the corpus. These are represented as nodes in a graph (see Figure 5). The edges represent the possible transitions between units. Following the edges, a path through the graph is searched with the lowest cost to find the most appropriate units. Currently, only one cost function is implemented using the Euclidean MFCC distance to measure spectral similarity. However, further development of better cost functions is necessary. Previous research [7] has shown that the Euclidean distance between MFCCs does not correspond very well to human perception of spectral discontinuities. The best performing measure from this study is the symmetrized Kullback-Leibler measure (SKL), which is currently being implemented in BOSS. Additional research concerning cost functions and smoothing is reported on in [9].

This strategy makes unit selection very efficient and allows for the use of very large corpora. The procedure is independent of the speech corpus used, and thus no changes to the software are necessary.

5. Discussion

Given the small size of the Dutch speech corpus, it is not difficult to imagine why the synthesis quality is still far from perfect. The question is what a corpus for a particular language should ideally contain. There is no cut-and-clear answer to that. A number of important questions remain.

How large should a corpus be? It is known that the quality improves with a larger corpus. A corpus should contain many different segments in diverse contexts.

So what prosodic variables are important? We know that syllable stress, word accent and position relative to the phrase boundary have an influence on phoneme durations and pitch contours. Also surrounding phonemes can cause coarticulation phenomena which make some concatenations at segment boundaries more problematic than others. These prosodic variables are not only important in the corpus construction but also in the selection process. One can see a trade-off between the number of selection criteria and the size of the corpus. Obviously, the more selection criteria, the more precise the units will match the target context, but with a smaller corpus the chances are that no appropriate candidates can be selected. Because there is no signal manipulation facility implemented in BOSS yet, there is no way to repair duration, F_0 or energy mismatches on the fly.

If there were a possibility to modify F_0 or duration, then what would the trade-off be with corpus size? It could well be that a word is found in an accented condition, where an unaccented condition would be required. In this case, that word could be selected and its pitch modified without much degradation of the signal quality.

What kind of sentences should be recorded? This is a difficult issue, especially for unrestricted text-to-speech. For limited domains, this is easier to determine because they contain less variation. In limited domain synthesis one can identify a number of frequently occurring concepts, such as the pronunciation of names, dates and numbers. [2] give examples of limited domain synthesis systems im-

plemented in the unit selection mode of Festival [10].

5.1. Directory assistance

As an example consider the pronunciation of phone numbers, for instance in an automatic directory assistance service. Figure 6 shows a pitch contour for an example Dutch phone number (032-40654, “nul twee en dertig / “vier nul / “zes vijf vier ///). Accents are indicated by a double quote, phrase boundaries by a slash. By analysing different instances of phone numbers, it was possible to come up with a prosody grammar for this type of concept. In order to have them pronounced correctly by the synthesis system, all numbers have to be recorded in five versions:

- A neutral low version.
- A neutral high version for numbers at the start of a phrase.
- An accented version.
- An accented version with a continuation rise for numbers before a phrase boundary.
- A version with a final fall for numbers at the end.

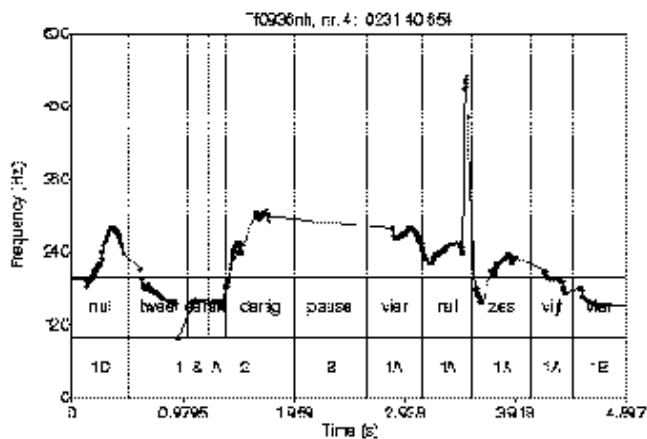


Figure 6: Example pitch contour of a Dutch phone number.

Other concepts can be tackled in the same way, ensuring at least a good quality output for these frequently returning items. A similar approach was taken to obtain a good speech quality from phrase concatenation in limited domains such as automatic train timetable systems [4]. For unrestricted text-to-speech, all we can say is that it is wise to collect as much speech as possible, containing both declarative sentences and different types of questions, and attempting to cover a wide variety of phonetic and prosodic contexts.

6. Conclusion

This paper has shown the addition of a new speech corpus to BOSS in a new language to be very straightforward. For the Dutch corpus it took little over a week to have a working version. Most of the time went into learning about the

structure of the data bases and writing a utility to convert the Waves+ label files to the BLF format. In the future we hope to obtain more contributions from the speech synthesis field both in terms of improvements to the synthesis system as in the contribution of more different corpora.

7. References

- [1] M. Beutnagel, A. Conkie, J. Schroeter, Y. Stylianou and A. Syrdal. The AT&T NextGen TTS System. *Proceedings of the Joint Meeting of ASA, EAA and DAGA, Berlin, Germany, 1999.*
- [2] A. Black and K. Lenzo. Limited Domain Synthesis. *Proceedings of the 6th Int. Conf. on Spoken Language Processing, Beijing, China, 2000*, vol. II, p411-414.
- [3] N. Campbell and A. Black. Prosody and the selection of source units for concatenative synthesis. In: J. van Santen, R. Sproat, J. Olive and J. Hirschberg (eds.), *Progress in Speech Synthesis, 1997*, p279-292. New York: Springer Verlag.
- [4] E. Klabbbers. Segmental and Prosodic Improvements to Speech Generation. *Ph.D. Thesis, Technische Universiteit Eindhoven, 2000.*
- [5] E. Klabbbers and J. van Santen. Predicting segmental durations for Dutch using the sums-of-products approach. *Proceedings of the 6th Int. Conf. on Spoken Language Processing, Beijing, China, 2000*, vol. III, p670-673.
- [6] E. Klabbbers, K. Stöber, R. Veldhuis, P. Wagner and S. Breuer. Speech synthesis development made easy: The Bonn Open Synthesis System. *Submitted to EUROSPEECH'01, Aalborg, Denmark.*
- [7] E. Klabbbers and R. Veldhuis. Reducing Audible Spectral Discontinuities. *IEEE Transactions on Speech and Audio Processing*, vol. 9, nr. 1, January 2001, p39-51.
- [8] K. Stöber, P. Wagner, J. Helbig, S. Köster, D. Stall, M. Thomae, J. Blauert, W. Hess, R. Hoffmann and H. Mangold. Speech Synthesis by Multilevel Selection and Concatenation of Units from Large Speech Corpora. In: *W. Wahlster (ed.), VerbMobil: Foundations of Speech-to-Speech Translation*, p521-538, Symbolic Computation, Springer, Berlin, 2000.
- [9] K. Stöber, P. Wagner, E. Klabbbers and W. Hess. Definition of a Training Set for Unit Selection-Based Speech Synthesis. *Submitted to 4th ISCA Tutorial and Research Workshop on Speech Synthesis, Pitlochry, Scotland.*
- [10] P. Taylor, A. Black and R. Caley. The architecture of the Festival speech synthesis system. *Proceedings of the 3rd ESCA/COCOSDA Workshop on Speech Synthesis, Jenolan Caves, Australia, 1998*, p147-152.