BAS Speech Science Web Services – an Update on Current Developments

Thomas Kisler*, Uwe D. Reichel†, Florian Schiel*, Christoph Draxler*, Bernhard Jackl*, Nina Pörner*

*Bavarian Archive for Speech Signals, Ludwig-Maximilians-Universität München
Schellingstr. 3, 80799 München, Germany
{kisler,schiel,draxler,jackl,poerner}@bas.uni-muenchen.de

†Research Institute for Linguistics, Hungarian Academy of Sciences
Benczúr u. 33, 1068 Budapest, Hungary
uwe.reichel@nytud.mta.hu

Abstract

In 2012 the Bavarian Archive for Speech Signals started providing some of its tools from the field of spoken language in the form of Software as a Service (SaaS). This means users access the processing functionality over a web browser and therefore do not have to install complex software packages on a local computer. Amongst others, these tools include segmentation & labeling, grapheme-to-phoneme conversion, text alignment, syllabification and metadata generation, where all but the last are available for a variety of languages. Since its creation the number of available services and the web interface have changed considerably. We give an overview and a detailed description of the system architecture, the available web services and their functionality. Furthermore, we show how the number of files processed over the system developed in the last four years.

Keywords: automatic segmentation, grapheme-to-phoneme, syllabification, forced alignment, RESTful web service, web interface

1. Introduction

With the emergence of standards for modern web technologies and, at least equally important, browsers that support them, it is possible to implement even complex software as web applications that run in a browser. Since most users already have a modern browser installed on their computer, there is no need for additional software. In late 2014 the new HTML5 standard has been approved and is now supported acceptably by most modern browsers (Hickson et al., 2014; Leenheer, 2015). Software-as-a-service (SaaS) means that software is no longer purchased once by the customer, but rather payed for when accessed. In the academic world the core concept stays the same, even if software is often offered for free. This means that the user accesses a software resource on-demand running on the infrastructure of the software provider, without the need to install specialized software on her computer (Buxmann et al., 2008). Some examples for SaaS, next to ours, are the EmuLabeller, a tool for online labeling of speech signals (Winkelmann and Raess, 2014), WebLicht, an engine that enables the user to construct linguistic chains to automatically annotate text corpora (Hinrichs et al., 2010) and WebAnno, a tool for doing manual text annotation in the browser (de Castilho et al., 2014). For other available resources within the CLARIN infrastructure, please check CLARIN (2015).
The new HTML5 standard, as Anthes (2012) states, does not only have advantages for users, but also for developers. With modern browsers supporting many of the specified technologies and state-of-the-art debuggers, developing software for the web has become much easier.
In the last four years, the Bavarian Archive for Speech Signals (BAS) transformed a few of its most used applications to services. As these tools have historically been Linux command line tools, the number of users reached was quite limited. By making the core functionality of these tools accessible over the web the potential user audience is much larger. No complex software packages need to be installed, meaning that no technical background knowledge is required of the user. An additional advantage of SaaS is that the user does not have to take care himself to get and install updates of new versions and always immediately benefits from bug-fixes.

Services and front-ends that are readily available at BAS (all free for academic use):

- WebMAUS (language dependent): automatic phonetic segmentation & labeling using pronunciation prediction based on the canonic phonemic or orthographic transcription for 17 languages and language variants (Schiel, 1999)
- WebMAUS (language independent): automatic phonetic segmentation & labeling using forced-alignment based on the canonic phonemic transcription in a language-independent SAM-PA mode (for small and endangered languages) (Schiel, 1999)
- G2P: automatic grapheme-to-phoneme conversion for 18 languages and language variants (Reichel, 2012)
- WebMINNI: automatic phone recognition & segmentation for 7 languages and language variants
- Pho2Syl: automatic phonetic syllabification for 18 languages and language variants
- ChunkPreparation: preprocessing of chunk-segmented and transcribed large video recordings for a more efficient WebMAUS segmentation
- TextAlign: Automatic alignment of text sequence pairs by minimizing their edit distance. Probabilistic cost functions are intrinsically calculated or can be imported.
- COALA: automatic generation of CMDI-based metadata files to simplify the integration of speech and multimedia corpora into CLARIN repositories.

- MaryTTS: a free German synthesis with 4 voices based on the MARY TTS system (Schröder and Trouvain, 2001)

Earlier versions of these services have been described in Kisler et al. (2012) and Kisler et al. (2015), here we give a summary of recent updates and extensions.

2. Architecture

2.1. Architectural Overview

The BAS web services are wrappers around already existing tools. They are usually Linux command line tools without graphical user interface (GUI). These web services, including a few helper services, can then be accessed using the web front-end or other programs and tools, allowing access to the above-mentioned core functionalities. By using a traditional client-server architecture that separates the GUI (client) and the data storing and processing back-end (server) through a web service interface, we achieve high decoupling of system components. Exchanging the front-end could therefore be done without the need of changing the back-end services, which has been done recently to accommodate for better extensibility (as described in section 2.3.). A depiction of the system architecture can be found in Figure 1.

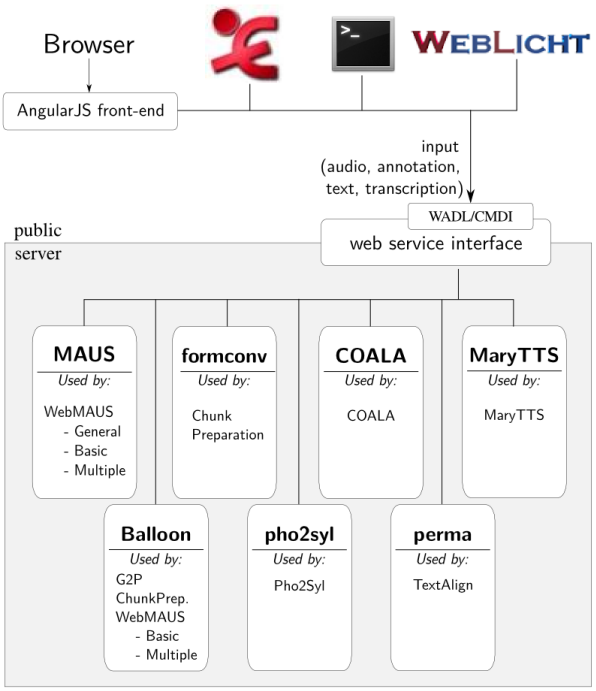

Figure 1: Architectural overview of the BAS speech science web services.

2.2. Web Services

The back-end services wrap the underlying functionality provided by the Linux command line tools. This is achieved using Java and JAX-RS based web services, running in a Tomcat Servlet container. To hide the Tomcat server location, we are using an Apache HTTP server as a proxy. We implemented our services in a RESTful way, but as we have a process-oriented view, our services are best described as RESTful remote procedure calls (RPC), where we use the HTTP methods GET and POST as envelopes for our RPCs. The web service response is then returned within an XML envelope. This complicates the handling of the server response to some extent, especially if called from the command line, but was necessary to successfully communicate warnings that occurred in the back-end to both the front-end and the user.

The back-end web services can be used much like traditional Linux command line tools if needed. Through tools like curl[1] these services can easily be called from within a script or a program. To enable users and developers to do so successfully, the web services are described in two metadata formats. These are the Web Application Description Language (WADL) and the Component Metadata Infrastructure (CMDI). For the latter we use the CMDI CLARIN core data model for web services (Windhouwer et al., 2012). In our case the WADL description is used to determine the necessary envelope to call the services and the CMDI description describes the necessary parameters and their meaning (links to both descriptions can be found in section 6.). For automatic service invocation it is necessary to look at both files. For example, a special helper service (a link can be found in section 6.) that shows information about all the available services uses them to generate example curl calls.

This metadata description allows for automatic checks whether the passed parameters are valid. Many options have a clear parameter range like closed vocabularies (only a set of fixed strings that is accepted as input, e.g. languages) or specific types that can be checked, e.g. numbers or Booleans. In case the user is calling the web services with wrong parameters, it is possible to notify him about the wrong parameter and suggest possible options that would be valid.

An example for the successful integration of one of our services to an application is ELAN (Wittenburg et al., 2006). Using ELANs plug-in mechanism, WebMAUS can be called directly within the application. The user has to select the correct signal file, the tier containing the orthographic transcript and the correct language, and ELAN sends it to the WebMAUS service. After successful service execution, the result is directly integrated as a new transcription layer. This saves the user the trouble of exporting the files and later re-importing them (Kisler et al., 2012).

Two further advantages of services for users are the installation/updating and processing power. As mentioned before users access the software that is installed on the server and therefore they do not have to install software locally to their computer (assuming they already have a recent browser installed). This does not only save time, but also ensures using the latest version (regarding new features and bugfixes), without even checking if there is an update available.

2.3. Web Interface

As it became obvious that adding services to the web interface was a reoccurring task, but was too complicated in the old interface, described in Kisler et al. (2012), it was replaced by a new one. The new interface was designed

[1] https://curl.haxx.se/

with extensibility in mind and we tried to incorporate the experiences of implementing the old interface. The new interface is written using the JavaScript framework AngularJS[2], which is based on a modified Model View Controller (MVC) pattern for separating display and logic (speaking about display and logic in the front-end) (Google, 2015). Unlike the name suggests, we are using the scripting language TypeScript, which is a typed programming language that translates to JavaScript files.

AngularJS provides a good basis for our web interfaces, as it allows for the creation of custom HTML tags (called directives in AngularJS), which ensure encapsulation of common functionality and therefore easier creation of new interfaces. This means that certain parts, like uploading files to the server, displaying a preview, etc. are implemented as a directive and a final service page is then created by using those basic building blocks.

Through the CMDI metadata description of the back-end, some parts of the front-end that were subject to regular changes are automatically generated from that description. One example of this are the options that are available for each interface. The CMDI XML description is parsed and added to the front-end on-the-fly. This means for adding or changing options (like adding languages), the web application does not have to be modified in the source code, only the metadata description has to be adapted. This has the benefit that only one place has to be modified and people without programming skills can perform the task.

2.4. Processing Power

The actual processing of the files is done on the server. This has the advantage that processing speed is independent of the client computer's performance. At the moment (March 2016) we are running the web services on a computer with two Intel®[3] Xeon® (X5650) processors, which means a total of 12 cores and up to 24 simultaneous threads (via Intel® Hyper Threading). This allows access for multiple users at the same time without slowing down the processing for other users. All web interfaces that allow to upload multiple files per drag and drop make use of this feature and process, for one user, multiple files at the same time.

3. Web Services in Detail

3.1. WebMAUS – automatic segmentation

The service is based on MAUS, which produces a segmentation and labeling of a recorded speech signal based on the orthographic or phonological transcript (Schiel, 1999). It combines an HMM-based acoustic model of phone segments with a stochastic predictor model of pronunciation, both being language dependent. Phonetic transcriptions created by MAUS have been evaluated against human transcriptions on a German benchmark derived from the Verbmobil corpus. When comparing inter-labeler transcription agreements among three human labelers with the pairwise transcription agreements of MAUS against the human transcribers, it turns out that MAUS transcriptions on average reach about 97% transcription accuracy of that of human labelers (Kipp et al., 1997). The time performance of MAUS depends on language, length of signal file and other factors, but in general MAUS segments and labels a signal faster than its playback time. In our experience human labelers performing a similar task need a factor of between 1:50 to 1:150 (playback time:time needed for segmentation and labeling). Recent advances of the MAUS development are the extension to now 17 languages, the handling of a hierarchical annotation format based on the Emu Database system (Winkelmann and Raess, 2014), the modeling of four different varieties of English (British, American, Australian and New Zealand), new output encodings in IPA, phonetic place of articulation and manner of articulation, and the modeling of Swiss German based on the Dieth encoding standard. Web applications based on the MAUS web service allow the instant display of segmentation results in the browser using the JavaScript-based Emu labeler EMU-webApp[4]. For the future we plan to allow the upload of customized stochastic pronunciation models, the extension to Spanish, Czech, Japanese and Cantonese, and an improved embedding into the Emu Database framework.

3.2. WebMINNI – automatic phone recognition

A new web application called 'WebMINNI' has been launched recently to the BAS web services page. WebMINNI – as the small girlfriend of WebMAUS – allows the phonetic transcription and segmentation of a speech signal into IPA or SAMPA, or broader classes such as manner or place of articulation. In contrast to WebMAUS this service does not require any orthographic or phonological transcript. The basic idea behind WebMINNI is to replace the pronunciation model of MAUS by a phonotactic bi-gram model of the chosen language. Currently the following languages are supported: German, Swiss German, Estonian, American English, Australian English, Hungarian and Italian.

The precision of WebMINNI is of course below that of MAUS, but nevertheless the tool can be very useful to obtain a rough segmentation of non-transcribed signals into phonetic or broader phonetic classes (e.g. voiced vs. unvoiced etc.).

3.3. G2P – automatic text-to-phoneme conversion

Our G2P service provides an automatic phonetic transcription for 18 languages and language variants including syllabification and word stress assignment. Transcription and syllabification are carried out by means of decision trees, and word stress assignment by a Bayes classifier within a learning by analogy framework. Details of the underlying algorithms can be found in Reichel (2012) and Reichel and Kisler (2014). The G2P service supports five input and nine output formats, among them the BAS partitur format (BPF) (Schiel et al., 1998) and Praat TextGrids (Boersma, 2001). For all languages and variants a basic text normalization in terms of tokenization and cardinal number expansion is provided. For English and German the text normalization additionally covers the classification and expansion of 22 non-standard word classes by means of pattern matching and local grammars, among them different number types

[2] https://www.angularjs.org

[3] www.intel.com

[4] https://github.com/IPS-LMU/EMU-webApp

(e.g. ordinal, phone number, postal code), acronyms, abbreviations, and URLs. The non-standard word taxonomy was adopted from Sproat et al. (2001).

3.4. Pho2Syl – phonetic syllabification

Our Pho2Syl service provides an automatic syllabification of phonetic transcriptions in 18 languages and language variants. As in combination with grapheme-phoneme conversion the standalone version of syllabification is also based on decision trees and a sonority-based fallback. The service supports the BAS partitur format as input and additionally the Praat TextGrid format as output. The user can choose between a word-synchronous syllabification and a syllabification irrespective of word boundaries. The former is of use when processing canonic transcriptions, the latter for connected speech transcriptions.

3.5. TextAlign – automatic text alignment

This service aligns text sequence pairs by minimizing their edit distance. Edit operations are substitution, insertion, and deletion. Next to a naive cost function penalizing any edit operation but null substitution by 1, cost functions can be imported, or estimated probabilistically from the input data, or can be chosen from prestored examples. The cost function estimation is described in detail in Reichel (2012). Typical use cases are the alignment of letters and phonemes in pronunciation dictionaries, and the alignment of canonic and spontaneous speech transcriptions in order to infer or verify phonological rules. The service takes a CSV file with each row containing a string pair to be aligned. It outputs a two-column CSV file with the aligned result. If the cost function is estimated from the input data this function is additionally returned as a CSV file. It can then be re-used for further applications of the aligner.

3.6. ChunkPreparation – preprocessing of chunk-segmented recordings

The ChunkPreparation service enables the segmentation of certain parts of the signal. One use case would be a scenario where an interviewer and an informant are recorded, but only the informant's speech is of interest. Similarly, ChunkPreparation is useful when only a short stretch of a long recording has been transcribed for analysis.

As input we take a specification of these chunks in either TextGrid, ELAN or CSV format (containing the three columns time on-, offset and the text). The service internally executes the G2P service to generate the necessary canonic information of the specified parts and returns a BAS Partitur File. This file in turn enables the user to feed the chunk segmentation information into WebMAUS, to generate the final segmentation & labeling.

3.7. COALA – automatic generation of CMDI-based metadata

In contrast to the other services in this paper, COALA does not analyze primary or secondary data but deals with metadata. It facilitates the creation of CMDI files for multimodal corpora, especially for large ones. As input files, five regular CSV tables are needed, one for each metadata category: media files, written resources (annotation to media files), bundles (grouping media files together that were recorded in parallel, as well as their derived annotations), actors (signers/speakers) and sessions, the collection of all recordings (here bundles) done in a consecutive order and with the same actor(s). COALA parses these tables and transforms them into one CMDI file for every session and additionally one file for the corpus itself. The session files generated are already valid metadata files and ready for further processing, whereas the corpus file is generated as a template file and needs to be edited by hand. The resulting files can then be easily integrated into a CLARIN repository. Adding a corpus to a repository does not only make it accessible and visible to a broader audience, but also ensures access to and persistence of the data and prevents the data from being forgotten, long after the corresponding project has finished.

4. Intended Users

The BAS is associated with the chair of Phonetics and speech processing. Many of the tools behind the web services were implemented to support research by assisting in the preprocessing of speech databases. The tasks involved are often tedious and repetitive and can be sped up greatly by (semi-)automatic methods. The tools are traditionally Linux command line tools which limited the possible number of users. This is because the installation of software packages requires a certain level of familiarity with the process and even if the software is available, in our experience the command line rarely is the preferred way of accessing a piece of software.

Within the CLARIN initiative (CLARIN, 2015) we implemented the aforementioned web interfaces to allow access to our services to a broader audience. Doing that, extending the number of possible users of an existing tool, is one of the aims of the CLARIN project, which should be achieved by making them publicly available or online accessible. As argued before, especially the latter is a promising way of doing so, as many technical details are hidden from the end user and, therefore, access to the software is made easier.

Our aim is to enable all researchers to use our tools, regardless of their technical background. Since we started providing the web interfaces we got feedback from researchers from various technical and non-technical fields and disciplines such as ethnology, anthropology, sociophonetics, historical linguistics, dialectology, speech technology, computer science, and computer assisted learning, which all successfully used our tools.

5. Usage Statistics

During the last four years the usage of the web services at the BAS shows a slight but steady increase over time. The trend can be seen by the blue, dashed linear regression line in Fig. 2. Two months with extraordinary usage have been October and November 2015 where combined over 767 thousand files have been processed using various services, both via front-end and back-end calls. These numbers seem very high and to be sure that they do not solely influence the regression line towards a positive increase, the red, dotted line shows the regression line where those two

outliers are set to the average calls over the last four years (the average was calculated by excluding the two months). The increase in usage can still clearly be seen in Fig. 2. This second line is shown to illustrate that the increase in usage is not only accounted to those two months, though they have, as expected, a big impact on the overall trend. Nevertheless, we want to emphasize that these calls, despite being outliers, did actually happen.

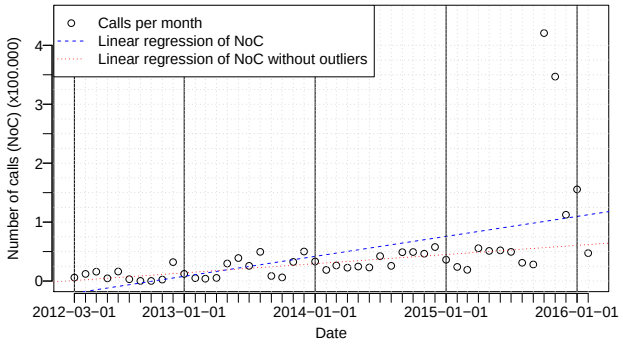

Figure 2: Number of calls to the web services per month over the last four years.

6. URLs

- **BAS web interface (all services):** `http://clarin.phonetik.uni-muenchen.de/BASWebServices`
- **BAS web services metadata description (CMDI):** `https://clarin.phonetik.uni-muenchen.de/BASRepository/WebServices/BAS_Webservices.cmdi.xml`
- **BAS web services WADL description** `http://clarin.phonetik.uni-muenchen.de/BASWebServices/services/application.wadl`
- **BAS web services help page:** `http://clarin.phonetik.uni-muenchen.de/BASWebServices/services/help`
- **CLARIN Component Metadata model (CMD):** `http://www.clarin.eu/content/component-metadata`
- **CLARIN ERIC:** `http://clarin.eu/`
- **CLARIN-D:** `http://de.clarin.eu/`

7. Acknowledgements

We thank the European CLARIN ERIC and the German CLARIN-D consortium funded by the German Ministry of Science and Education for establishing the infrastructure, as well as hundreds of users' valuable feedback to improve our services.

8. Call for Resources

We are always looking for language resources to extend our services. Examples for those resources are

- MAUS: manually segmented and labeled speech corpora of languages, that we do not provide yet. Necessary are at least 50 speakers with combined around 2h of phonetically rich material
- G2P: pronunciation lexica with orthographic transcript and some form of consistent phonemic transcription

Furthermore, we are happy to receive proposals for new services, if they lie within the scope of our work.

9. References

Anthes, G. (2012). HTML5 Leads a Web Revolution. *Commun. ACM*, 55(7):16–17, July.

Boersma, P. (2001). Praat, a system for doing phonetics by computer. *Glot International*, 5(9/10):341–345.

Buxmann, P., Hess, T., and Lehmann, S. (2008). Software as a service. *WIRTSCHAFTSINFORMATIK*, 50(6):500–503.

CLARIN. (2015). Overview of available clarin web services. http://clarin.eu/content/web-services, last accessed: 2016-03-09.

de Castilho, R. E., Biemann, C., Gurevych, I., and Yimam, S. M. (2014). Webanno: a flexible, web-based annotation tool for clarin. In *Proceedings of the CLARIN Annual Conference (CAC) 2014*, page online, Utrecht, Netherlands, October. CLARIN ERIC. Extended abstract.

Google. (2015). AngularJS. last accessed: 2016-03-09.

Hickson, I., Berjon, R., Faulkner, S., Leithead, T., Navara, E. D., O'Connor, E., and Pfeiffer, S. (2014). HTML5. W3C recommendation, W3C, October. http://www.w3.org/TR/2014/REC-html5-20141028/, last accessed: 2016-03-10.

Hinrichs, E., Hinrichs, M., and Zastrow, T. (2010). Weblicht: Web-based LRT services for german. In *Proceedings of the ACL 2010 System Demonstrations*, ACLDemos '10, pages 25–29, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kipp, A., Wesenick, B., and Schiel, F. (1997). Pronunciation Modeling Applied to Automatic Segmentation of Spontaneous Speech. In *Proceedings of the EUROSPEECH*, pages 1023–1026, Rhodos, Greece.

Kisler, T., Schiel, F., and Sloetjes, H. (2012). Signal processing via web services: the use case webmaus. In *Proceedings Digital Humanities 2012*, pages 30–34, Hamburg.

Kisler, T., Schiel, F., Reichel, U. D., and Draxler, C. (2015). Phonetic/linguistic web services at BAS. In *Proc. Interspeech*, page paper 2609, Dresden, Germany.

Leenheer, N. (2015). HTML5TEST – how well does your browser support html5? https://html5test.com/, last accessed: 2016-03-09.

Reichel, U. and Kisler, T. (2014). Language-independent grapheme-phoneme conversion and word stress assignment as a web service. In R. Hoffmann, editor, *Elektronische Sprachverarbeitung 2014*, volume 71 of *Studientexte zur Sprachkommunikation*, pages 42–49. TUDpress, Dresden, Germany.

Reichel, U. (2012). PermA and Balloon: Tools for string alignment and text processing. In *Proc. Interspeech*, page paper no. 346, Portland, Oregon, USA.

Schiel, F., Burger, S., Geumann, A., and Weilhammer, K. (1998). The Partitur Format at BAS. In *Proc. of the 1st Int. Conf. on Language Resources and Evaluation*, pages 1295–1301, Granada, Spain.

Schiel, F. (1999). Automatic Phonetic Transcription of Non-Prompted Speech. In *Proc. of the ICPhS*, pages 607–610, San Francisco, August.

Schröder, M. and Trouvain, J. (2001). The German text-to-speech synthesis system MARY: A tool for research, development and teaching. In *Proceedings of the 4th ISCA Tutorial and Research Workshop on Speech Synthesis, August 29–September 1*, pages 131–136, Perthshire, Scotland.

Sproat, R., Black, A., Chen, S., Kumar, S., Ostendorf, M., and Richards, C. (2001). Normalization of non-standard words. *Computer Speech & Language*, 15(3):287–333.

Windhouwer, M., Broeder, D., and Van Uytvanck, D. (2012). A CMD core model for CLARIN web services. In *Proceedings of LREC 2012: 8th International Conference on Language Resources and Evaluation*, pages 41–48.

Winkelmann, R. and Raess, G. (2014). Introducing a web application for labeling, visualizing speech and correcting derived speech signals. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland.

Wittenburg, P., Brugman, H., Russel, A., Klassmann, A., and Sloetjes, H. (2006). Elan: a professional framework for multimodality research. In *Proceedings of Language Resources and Evaluation Conference (LREC)*.