LANGUAGE-INDEPENDENT GRAPHEME-PHONEME CONVERSION AND WORD STRESS ASSIGNMENT AS A WEB SERVICE

Uwe D. Reichel, Thomas Kisler

Institute of Phonetics and Speech Processing, University of Munich
{*reichelu|kisler*}*@phonetik.uni-muenchen.de*

Abstract: We introduce a new language-independent procedure for grapheme-phoneme conversion, syllabification, and word stress assignment. Grapheme-phoneme conversion and syllabification is carried out by means of fallback sequences of decision trees trained on varying context sizes. Word stress is determined within an analogy-based framework by means of a Bayes classifier. Evaluation results on six languages are presented. Furthermore, it is described, how the tool is implemented and to be accessed as a web service that is freely available for academic research.

1 Introduction

Feasible approaches for language-independent grapheme-phoneme conversion (G2P), syllabification (SYL) and word stress assignment (WSA) are necessarily constrained with respect to the method and the features from which the target values are to be inferred. Concerning the method, data-driven machine-learning approaches are preferably to be chosen, since they can much more easily be adapted to new languages than rule-based expert systems. Since in multilingual G2P processing commonly no elaborated modules for higher-level (e.g. morphological) analyzes are available for all languages, the feature choice needs to be restricted to basic features as letters for G2P and symbol-phonetic object characteristics as phonemes for SYL and syllable properties for WSA.

Grapheme-phoneme conversion Following [1] common machine-learning approaches to map graphemes to phonemes can be divided into (a) techniques based on local classification, (b) learning by analogy, and (c) probabilistic methods. Local classification techniques are trained to locally map a letter within its current environment to the corresponding target phoneme. This task is for example carried out by simple look up in tables that contain the shortest grapheme contexts for unique G2P mappings [2], neural networks [3, 4], [5, model "SIE"] or decision trees [6, 7, 8], [5, model "Pfitzinger"]. Especially decision trees allow for easy integration of higher-level morphological and part of speech information [8, 9] if available. In learning by analogy the G2P output is derived from similarities of the word to be transcribed or parts of it with entries stored in the pronunciation dictionary [10]. Among the probabilistic approaches are Hidden Markov models (HMM) treating letters as observations and phonemes as state labels [11], and joint sequence models allowing for more flexible segmentations of the letter and phone strings [1]. The G2P outcome then is given by the best path through the model. Opposed to the local classification technique in these approaches the G2P outcome for a word is optimized globally.

Syllabification Syllabification can be carried out on the orthographic or on the transcription level. As for G2P it can be divided into local classification, learning by analogy, and probabilistic methods. Local classification comprises the usage of neural networks [12] and classification trees [13]. A syllabification by analogy algorithm has been developed by [14] who adopt the syllabification of the most similar substrings in a pronunciation dictionary to a word or its transcription to be syllabified. Among probabilistic methods HMMs are employed e.g. by [15, 16].

Lang.	Dictionary	Size	Syllab.	Stress	$H(\text{Gra} \times \text{Pho})$	$I(\text{Gra};\text{Pho})$	$C(\text{Gra};\text{Pho})$
deu	Phonolex Core [21]	62.7 K	no	yes	5.85	3.45	0.59
eng	Celex [22]	52.2 K	yes	yes	5.94	2.97	0.50
hun	KORNAI corpus [23]	14.5 K	no	fixed	5.24	4.37	0.83
ita	Festival [24]	41.0 K	yes	yes	4.82	3.64	0.76
nld	Celex [22]	118.0 K	yes	yes	5.53	3.37	0.61
pol	PJWSTK [25]	31.7	no	fixed	4.98	4.29	0.86

Table 1 - Overview over the used training dictionaries. The languages are expressed in ISO 639-3. Three of the dictionaries do not contain syllabification. Word stress is fixed for Hungarian and Polish. Language dependent grapheme-phoneme pairing entropies, mutual informations of phonemes and graphemes, and G2P mapping consistencies [26] were calculated on the aligned dictionaries. These measures are explained in section 6.

Word stress assignment Whereas for G2P and SYL the choice of the prediction targets (phonemes, syllable boundaries) is uniform across different approaches, word stress prediction can be addressed to each syllable in separation yielding a categorical output for each syllable (e.g. stress vs. unstressed; [5, model "IPS"]) or to the word as a whole returning one or more indexes of stressed syllables [17]. As for G2P and SYL also WSA approaches can be divided into local classification, learning by analogy and statistical approaches. Again neural nets [17, 3] and decision trees [5] are popular local classifiers for this purpose. Probabilistic approaches often treat WSA as a part of G2P simply by extending the set of target phonemes by stressed vowels, so that G2P additionally distinguishes between stressed and unstressed phonemes as in the HMM approach of [18]. Leaning by analogy is backed up by phonetic pseudo word production studies in which subjects take over stress patterns of "phonetically similar" words of their language. In [19] an extensive overview on studies on Dutch, English, and German is given. "Phonetic similarity" at least in these three languages generally refers to syllable weight patterns. The weight of a syllable depends on the syllable rhyme, more specific on the vowel quantity and the presence or absence of a coda. [20] propose an instance-based learning approach assigning to new words the word stress of those stored entries that have the most similar syllable pattern. This pattern derived from the last three syllables is defined on three different levels of abstraction: syllable weight, rhyme transcription, and plain transcription. The features are weighted by their information gain for word stress assignment.

Current approach Referring to the method taxonomy above our language-independent approach consists of local classification for G2P and syllabification and analogy-driven learning for word stress assignment. After an overview over the training data and its preprocessing we introduce each of the modules in greater detail, present their performances, and explain how they are integrated and accessible as a web service.

2 Preparation of the training data

After separation of transcription, syllabification and word stress information the transcription was aligned to the letters by means of the PermA aligner [8]. In short, PermA aligns the letter sequence v and its transcription w by minimizing their Levenshtein distance in terms of the minimum edit costs to transform v into w. The edit costs c for the edit operations substitution, deletion and insertion are defined in terms of conditional probabilities reflecting co-occurrences between letters and phones (including empty phones) as follows:

- **Substitution:** $c(v_i, w_j) = 1 - P(w_j|v_i)$
- **Deletion:** $c(v_i, \_) = 1 - P(\_|v_i)$
- **Insertion:** $c(\_, w_j) = 1 - P(w_j|\_)$

Syllabification is stored as a sequence of indexes of syllable final phonemes. Word stress is stored as an integer pointing to the index of the major-stressed syllable. The lexicon entry

checkerboard; ' tS e . k @ . b O: d

thus was split into:

- aligned transcription

c	h	e	c	k	e	r	b	o	a	r	d
tS	_	e	k	_	@	_	b	O:	_	_	d

- syllabification: 2, 4
- word stress: 1

3 Grapheme Phoneme Conversion

For G2P conversion we extended the decision tree approach of [8] to cover grapheme contexts and phoneme histories of variable length. For each window and history length we trained an individual C4.5 decision tree [28] with the corresponding feature vectors. For the symmetric grapheme contexts the window lengths were successively reduced at both ends from 9 to 0, the length of the phoneme histories was reduced from 1 to 0. The pruning confidence level was set to 0.25, and the trees were designed to return an "unknown" *UNK* label for all cases they cannot generalize to.

In analogy to the table look up approach of [2] these trees are than applied as a fallback sequence taking grapheme context and phoneme history of decreasing size into account. The tree sequence procedure is illustrated in the subsequent pseudo code. Starting with the most specific (largest) context (window lengths l_{wmax} and h_{hmax}, respectively), the feature windows are successively shortened until a mapping to a phoneme is found. First the phoneme history length l_h is shortened from 1 to 0, then the grapheme window length l_w is shortened at both ends (thus by stepsize 2) from 9 to 1. As soon as a tree trained on the current context size does not return *UNK*, its output is added to the conversion result and the next grapheme is processed.

function **applyTreeSequence**(l_{wmax}, l_{hmax})
  $l_w \leftarrow l_{wmax}, l_h \leftarrow l_{hmax}$, $ans \leftarrow UNK$
  while $ans = UNK \wedge l_w \geq 1$
    classifier $\leftarrow$ tree trained on features in windows of lengths l_w, l_h
    $ans \leftarrow$ **apply** *classifier*
    if $l_h > 0$: $l_h \leftarrow l_h - 1$; **else** $l_w \leftarrow l_w - 2$
  return *ans*

The G2P module outputs the corresponding transcription of the input words in the language-specific SAMPA [27] encoding.

4 Syllabification

Like G2P also syllabification is implemented as a tree sequence successively taking decreasing context into account. This time the context is formed by a symmetric phoneme window centered on the phoneme for which is to be decided whether or not it is syllable-final. l_{wmax} and l_{hmax} were set to 7 and 0, respectively (thus no boundary history was taken into account).

To ensure that exactly one syllable boundary is placed between each pair of subsequent syllable nuclei, first these nuclei are identified in the transcription output of the G2P module (example: *coherence* – / *k* **@U** *h* **I@** *r* @ *n s* /). For each consecutive nucleus pair (e.g. *@U* and *I@*) moving from left to right the tree sequence is applied on the phoneme window centered on the respective phoneme (first *@U*, then *h*) until a boundary is reported or the right nucleus is reached. If no boundary has been signaled between the two nuclei, a fallback boundary is placed in front of the sonority minimum in the transcription substring (in this case *h*). If the sonority minimum is a plateau, the boundary is shifted rightwards towards the last element of this plateau. This heuristic serves to avoid violations of phonotactics.

For German, Hungarian, and Polish only the fallback boundary placement was carried out, since the used dictionaries did not contain syllabification information.

5 Word stress assignment

For word stress assignment we have developed a Bayes classifier within a learning by analogy framework. In the training phase the transcriptions in the lexicon are mapped onto codes of different levels of abstraction most of them preserving the words' syllable weight pattern. For the resulting codes the probability distribution of word stress positions is estimated. Furthermore, similar to [20], for each code mapping operation the mutual information between its code outputs and the word stress location is calculated.

In application an incoming word is mapped onto the same abstract representations as described for the training phase. A Bayes classifier predicts the most probably stress position based on the stress distributions for each of the abstractions and the mutual information of the corresponding code operation.

5.1 Code operations

The syllabified input transcription is successively transformed into five increasingly abstract codes. The transformations are ordered, and each transformation operates on the output of the preceding one. The transformations are illustrated given the syllabified input transcription / k @U . h I@ . r @ n s / (*coherence*) returned by the G2P and syllable module.

- t_1 replace all consonants by %C: / %C @U . %C I@ . %C @ %C %C /
- t_2 reduce %C-sequences within a syllable to single %C: / %C @U . %C I@ . %C @ %C /
- t_3 remove all onset consonants: / @U . I@ . @ %C /
- t_4 replace all syllable nuclei; reduced vowels and syllabic consonants by %R, long vowels and diphthongs by %VV, all others by %V: / %VV . %VV . %R %C /
- t_5 remove all coda consonants: / %VV . %VV . %R /

Operations t_1 to t_4 preserve the syllable weight pattern of the transcription.

5.2 Bayes classification

5.2.1 Probability and weight estimation

In the training phase each transcription of the lexicon was transformed into five keys by the operations $t_{1...5}$ to calculate the co-occurrence frequencies of these keys k with the word stress position s of the original transcription. From these co-occurrence frequencies smoothed by absolute discounting the conditional probabilities $P(k|s)$ were estimated. Furthermore, the word stress prior probability $P(s)$ was calculated.

In order to weight the contributions of the code operations, for each operation t_i the mutual information I_i between its outputs T_i and the word stress positions S was calculated. The mutual information values were then transformed into weights u_i by normalizing them to sum up to 1.

$$\begin{aligned} I_i &= H(S) - H(S|T_i) \\ u_i &= \frac{I_i}{\sum_j I_j} \end{aligned}$$

$H(S)$ is the entropy of word stress in the training dictionary, and $H(S|T_i)$ the word stress entropy after application of the i-th transformation in the lexicon. The higher I_i the more the word stress is determined by the outcomes of transformation t_i.

5.2.2 Application

The location of the word stress $\hat{s}$ in a transcription *trs* results from maximizing the following term:

$$\hat{s} \quad = \quad \arg\max_{s}[\prod_{i=1}^{5}(u_i \cdot P(t_i(\text{trs})|s)) \cdot P(s)]$$

$t_i(\text{trs})$ stands for the i-th transformation of the input transcription. $P(t_i(\text{trs})|s)$ is the probability that the output of t_i is an abstract pattern of words with stress location s. u_i is the weight associated to the i-th transformation operation, and $P(s)$ is the stress location prior.

6 Results and Discussion

We evaluated our modules by 10-fold cross validation. Figures 1 and 2 display the word error rates (A) for G2P, SYL, and WSA taken together, (B) for G2P only, (C) for SYL only, and (D) for WSA only.

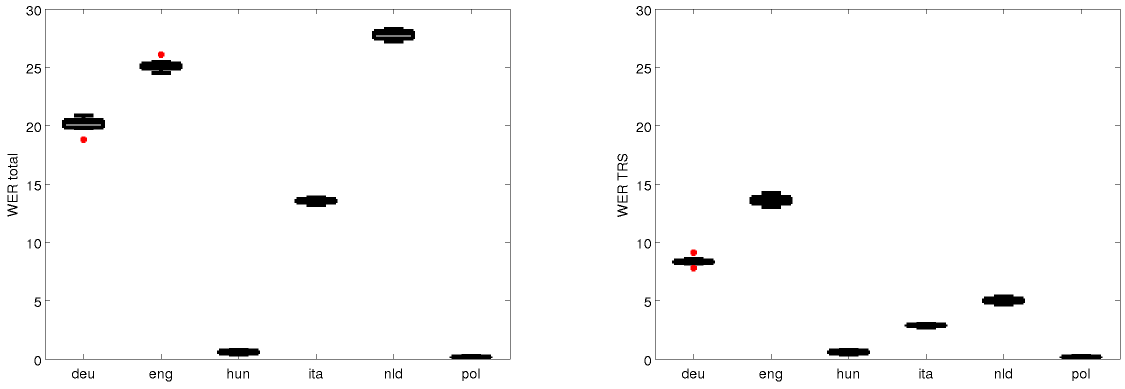

Figure 1 - **A (left):** Word error rate for G2P, syllabification, and stress assignment taken together. **B (right):** Word error rate for transcription only.

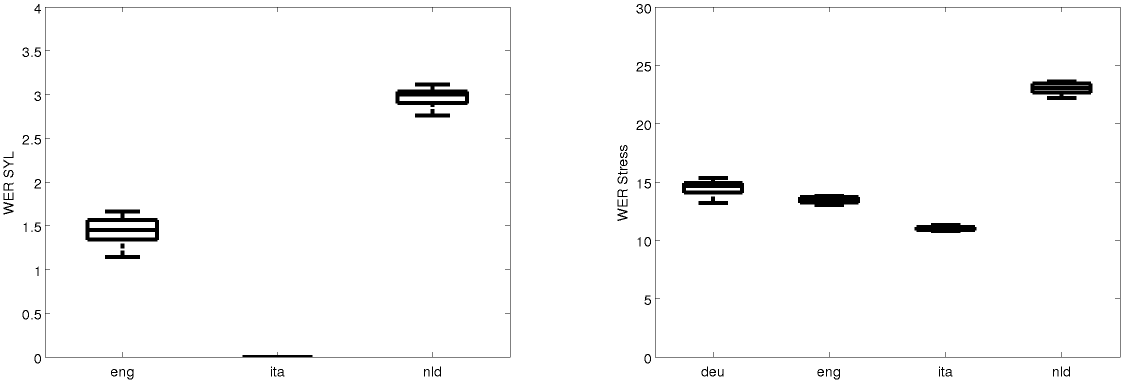

Figure 2 - **C (left):** Word error rate for syllabification. **D (right):** Word error rate for stress assignment.

It can be seen that the performance of our modules is highly language-dependent. While G2P performs very well for Hungarian, Italian, and Polish, results are poorer for German, English, and Dutch. These differences are due to much more systematic grapheme-phoneme correspondences in the first three languages. For Polish for example orthography follows the *phonetic principle* (*zasada fonetyczna*) [29, p.

85], that requires a stable fixed relation between sound and grapheme. [26] proposed a dictionary consistency measure $C(\text{Gra};\text{Pho})$ for the mapping from set of grapheme units *Gra* to the set of phoneme units *Pho* as follows:

$$C(\text{Gra};\text{Pho}) \quad = \quad \frac{I(\text{Gra};\text{Pho})}{H(\text{Gra} \times \text{Pho})}$$

It's the quotient of the mutual information $I(\text{Gra};\text{Pho}) = \sum_{x \in \text{Gra}} \sum_{y \in \text{Pho}} p(x,y) \log_2 \frac{p(x,y)}{p(x)p(y)}$ between grapheme and phoneme units resulting from dictionary alignment.
$H(\text{Gra} \times \text{Pho}) = -\sum_{x \in \text{Gra}, y \in \text{Pho}} p(x,y) \log_2 p(x,y)$ is the entropy of the *graphone* alphabet, i.e. the set of grapheme-phoneme unit pairings. In line with the observed performance differences, the last three columns in table 1 report higher consistency values (resp. lower entropy and higher mutual information values) for languages with easy-to-learn stable G2P relations.
While syllabification performs reasonably well there is still room for improvement for word stress assignment. A major shortcoming of our language-independent approach ignoring higher level morphological features is, that it does not allow for identifying compounds and thus stressed parts within these compounds. While in several languages in simplex forms stress is rather to be located within the range of the last three syllables of a word [19], it commonly occurs earlier in compounds when non-final compound parts receive the major stress. As can be seen in Figure 2 this is a major issue in Dutch and German with highly productive compounding tendencies.
In order to increase the performance of our converter in application we extended it by language-dependent G2P postprocessing modules, that account for example for regressive voice and place of articulation assimilation processes in Hungarian [30].

7 Web service

As a contribution to the 'Common Language Resources and Technology Infrastructure" (CLARIN) project, we developed web services that provide the functionality of tools like the converter introduced in this paper. Those web services wrap different functionality of command line tools, and allow for easy access to this software, without the need of installing it locally on the users machine. In this scenario the "user" might be a human or another web service that wants to access a particular piece of software.
In the CLARIN context the view on web services is rather process oriented, i.e. major efforts are put into implementing work flow engines, that allow for chaining different services [31]. We therefore implemented our web services as RESTful remote procedure calls (RPC). The reference implementation "Java API for RESTful Web Services" (JAX-RS) was used to implement the web service wrapper around the underlying grapheme-to-phoneme command line tool. We overload the POST operation, a standard HTML operation, that serves as an envelope for the data that has to be passed to the web service while remaining compliant with the RESTful idea [32].
To allow the automatic call or easy manual integration of our RESTful web services, we describe all web services both technically in the Web Application Description Language (WADL) and semantically in the Common Meta Data Infrastructure CMDI [33] format. The WADL format is a machine readable format describing the technical aspects of a certain web service. This allows programs or chaining engines to generate the call to the WADL described web service automatically on the technical side. The description in the CMDI format, based on components that are registered in the public component registry [34], allows for further automatic interpretation of the possible parameters that a web service takes. Those two descriptions can then be used to automatically integrate our G2P tool to any infrastructure or program that understands these descriptions or use it to integrate them manually in existing software. Examples of workflow engines in which our web services can easily be integrated are WebLicht [35] and Taverna [36]. Further information how to access the web services *runG2P* for the introduced converter can be found at: *http://clarin.phonetik.uni-muenchen.de/BASWebServices/services/help*

8 Acknowledgments

The work of the authors has been carried out within the CLARIN-D project [37] (BMBF-funded). We very much thank all institutions and projects referred to in Table 1 for providing the research community with their huge and very valuable pronunciation dictionaries.

References

[1] M. Bisani and H. Ney, "Joint-sequence models for grapheme-to-phoneme conversion," *Speech Communication*, vol. 50, pp. 434–451, 2008.

[2] A. van den Bosch and W. Daelemans, "Data-oriented methods for grapheme-to-phoneme conversion," ITK Research Report, Tilburg, Tech. Rep. 42, 1993.

[3] H.-U. Hain, "Automation of the training procedures for neural networks performing multi-lingual grapheme to phoneme conversion," in *Proc. Eurospeech*, Budapest, Hungary, 1999, pp. 2087–2090.

[4] J. Häkkinen, J. Suontausta, S. Riis, and K. Jensen, "Assessing text-to-phoneme mapping strategies in speaker independent isolated word recognition," *Speech Communication*, vol. 41, no. 2, pp. 455–467, 2003.

[5] U. Reichel, H. Pfitzinger, and H. Hain, "English grapheme-to-phoneme conversion and evaluation," *Speech and Language Technology*, vol. 11, pp. 159–166, 2008.

[6] K. Torkkola, "An efficient way to learn English grapheme-to-phoneme rules automatically," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Minneapolis (MN), USA, 1993, pp. 199–202.

[7] W. Daelemans and A. van den Bosch, "Language-Independent Data-Oriented Grapheme-to-Phoneme Conversion," in *Progress in Speech Synthesis*. New York: Springer, 1997, pp. 77–89.

[8] U. Reichel, "Perma and balloon: Tools for string alignment and text processing," in *Proc. Interspeech*, Portland, Oregon, 2012, p. paper no. 346.

[9] J. Suontausta and J. Häkkinen, "Decision tree based text-to-phoneme mapping for speech recognition," in *Proc. ICSLP*, Beijing, China, 2000, pp. 831–834.

[10] Y. Marchand and R. Damper, "Multistrategy approach to improving pronunciation by analogy," *Computational Linguistics*, vol. 26, no. 2, pp. 195–219, 2000.

[11] P. Taylor, "Hidden Markov Models for grapheme to phoneme conversion," in *Proc. Interspeech*, Lisbon, 2005, pp. 1973–1976.

[12] W. Daelemans and A. van den Bosch, "Generalization performance of backpropagaion learning on a syllabification task," in *Proc. 3rd Twente Workshop on Language Technology*, 1992, pp. 27–38.

[13] S. Pearson, R. Kuhn, S. Fincke, and N. Kibre, "Automatic methods for lexical stress assignment and syllabification," in *Proc. Interspeech*, Pittsburgh, Pennsylvania, 2000, pp. 423–426.

[14] Y. Marchand, C. Adsett, and R. Damper, "Automatic syllabification in English: A comparison of different algorithms," *Language and Speech*, vol. 52, no. 1, pp. 1–27, 2009.

[15] B. Krenn, "Tagging syllables," in *Proc. Eurospeech*, Rhodes, Greece, 2007, pp. 991–994.

[16] H. Schmid, B. Möbius, and J. Weidenkaff, "Tagging syllable boundaries with joint N-gram models," in *Proc. Interspeech*, Antwerp, Belgium, 2007, pp. 2857–2860.

[17] P. Gupta and D. Touretzky, "Connectionist models and linguistic theory: Investigations of stress systems in language," *Cognitive Science*, vol. 18, no. 1, pp. 1–50, 1994.

[18] H. Che, J. Tao, and S. Pan, "Letter-to-sound conversion using coupled Hidden Markov Models for lexicon compression," in *Proc. Speech Database and Assessments*, Macau, 2012, pp. 141–144.

[19] U. Dohmas, I. Plag, and R. Carroll, "Word stress assignment in German, English and Dutch: Quantitysensitivity and extrametricality revisited," *Comparative Germanic Linguistics*, p. 53 pages, 2013.

[20] W. Daelemans, S. Gillis, and G. Durieux, "The acquisition of stress: A data-oriented approach," *Computational Linguistics*, vol. 20, no. 3, pp. 421–451, 1994.

[21] F. Schiel, "BAS Phonolex," http://www.bas.uni-muenchen.de/forschung/Bas/BasPHONOLEXeng.html.

[22] R. Baayen, R. Piepenbrock, and L. Gulikers, "Celex lexical database," CD-ROM, 1995.

[23] P. Szigetvári, "Hungarian corpuses," http://budling.nytud.hu/ szigetva/etcetera/Hungarian/corpuses.html.

[24] "Italian festival," http://www2.pd.istc.cnr.it/FESTIVAL/home/download-FESTIVAL.htm.

[25] "Polish-Japanese Institute of Information Technology," http://www.pjwstk.edu.pl.

[26] M. Wolff, M. Eichner, and R. Hoffmann, "Measuring the quality of pronunciation dictionaries," in *Proc. PMLA ISCA Workshop*, 2002, pp. 117–122.

[27] "SAMPA – computer readable phonetic alphabet," http://www.phon.ucl.ac.uk/home/sampa/.

[28] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo: Morgan Kaufmann, 1993.

[29] B. Bartnicka, B. Hansen, W. Klemm, V. Lehmann, and H. Satkiewicz, *Grammatik des Polnischen*, ser. Slavolinguistica. Munich: Verlag Otto Sagner, 2004, vol. 5.

[30] P. Siptár and M. Törkenczy, *The Phonology of Hungarian*, ser. The Phonology of the World's Languages. Oxford University Press, 2007.

[31] V. Boehlke, "A generic chaining algorithm for nlp webservices," in *LREC 2010; Web Services and Processing Pipelines in HLT Workshop*, 2010, pp. 30–36.

[32] L. Richardson and S. Ruby, *RESTful Web Services*. Cambridge, MA: O'Reilly, 2007.

[33] "Component metadata," http://www.clarin.eu/cmdi.

[34] D. Broeder, M. Kemps-Snijders, D. Van Uytvanck, M. Windhouwer, P. Withers, P. Wittenburg, and C. Zinn, "A data category registry– and component-based metadata framework," in *Proc. LREC*, Valletta, Malta, 2010.

[35] M. Hinrichs, T. Zastrow, and E. Hinrichs, "Weblicht: Web-based LRT services in a distributed eScience infrastructure," in *Proc. LREC*, Valletta, Malta, 2010, pp. 489–493.

[36] D. Hull, K. Wolstencroft, R. Stevens, C. Goble, M. Pocock, P. Li, and T. Oinn, "Taverna: a tool for building and running workflows of services." *Nucleic Acids Research*, vol. 34, no. Web Server issue, pp. 729–732, 2006.

[37] "http://eu.clarin-d.de/index.php/en/," CLARIN-D web page.