WERTi - Easing Second Language Acquisition

Aleksandar Dimitrov

June 11, 2008

Contents

Preface

1 Introduction

Using tools and methods made available through new achievements in computational linguistics and related subjects like cognitive science to ease the process of second language acquisition has only recently gained focus in research projects. While adaptions of traditional data sources like dictionaries have been used for quite a while we have yet to discover new and effective ways of

WERTi tries to approach this probmlem from a more general perspective. Using the the momentum of the Internet, WERTi provides a platform for implementing linguistic analysis and subsequent input enhancement methods on user specified pages on the World Wide Web. Using Java Servlets technology for serving content and the UIMA framework for processing it in a dynamic and flexible session, we aim to provide a platform for linguistic processing of online content that can go beyond input enhancement.

1.1 Acknowledgements

I'd like to thank my advisor Detmar Meurers for providing me with the unique opportunity on working on the WERTi system. WERTi was originally thought out and developed in Python by him and his team at the Ohio State University. My job was to port the system he had already written to Java using UIMA and generalize the concept as to make it easier to implement new functions.

I would also like to thank Janina Radó, who provided me with invaluable feedback about the general style guidelines of papers of this kind.

Furthermore I'd like to thank , , , and for providing testing the system and for their suggestions, remarks and feature requests.

2 Explaining the Core Functionality

2.1 UIMA Analysis Engines

The current processing line can be split up into three different parts that can interact autonomously themselves but each depend on the output of their predecessor. The components are arranged in the following manner:

HTML processing:	annotates HTML tags, finds "relevant" text within the tags.
Linguistic processing:	performs linguistic tasks (tokenization, sentence boundary detection, part-of-speech tagging. . .) on the text-annoatations from the previous processing step.
Post processing:	Performs all sorts of content enhancement on the annotations of the previous two processing steps.

2.1.1 HTML Processing

2.1.2 Linguistic Processing

2.1.3 Post Processing - Input Enhancement

3 The User Inteface

3.1 Web Interface

3.2 Programming Interface

4 Conclusion

4.1 Loose Ends