WebAnno: A Flexible, Web-based and Visually Supported System for Distributed Annotations

Seid Muhie Yimam[1,3] **Iryna Gurevych**[2,3] **Richard Eckart de Castilho**[2] **Chris Biemann**[1]
(1) FG Language Technology, Dept. of Computer Science, Technische Universität Darmstadt
(2) Ubiquitous Knowledge Processing Lab (UKP-TUDA)
Dept. of Computer Science, Technische Universität Darmstadt
(3) Ubiquitous Knowledge Processing Lab (UKP-DIPF)
German Institute for Educational Research and Educational Information
http://www.ukp.tu-darmstadt.de

Abstract

We present WebAnno, a general purpose web-based annotation tool for a wide range of linguistic annotations. WebAnno offers annotation project management, freely configurable tagsets and the management of users in different roles. WebAnno uses modern web technology for visualizing and editing annotations in a web browser. It supports arbitrarily large documents, pluggable import/export filters, the curation of annotations across various users, and an interface to farming out annotations to a crowdsourcing platform. Currently WebAnno allows part-of-speech, named entity, dependency parsing and co-reference chain annotations. The architecture design allows adding additional modes of visualization and editing, when new kinds of annotations are to be supported.

1 Introduction

The creation of training data precedes any statistical approach to natural language processing (NLP). Linguistic annotation is a process whereby linguistic information is added to a document, such as part-of-speech, lemmata, named entities, or dependency relations. In the past, platforms for linguistic annotations were mostly developed ad-hoc for the given annotation task at hand, used proprietary formats for data exchange, or required local installation effort. We present WebAnno, a browser-based tool that is immediately usable by any annotator with internet access. It supports annotation on a variety of linguistic levels (called annotation layers in the remainder), is interoperable with a variety of data formats, supports annotation project management such as user management, offers an adjudication interface, and provides quality management using inter-annotator agreement. Furthermore, an interface to crowdsourcing platforms enables scaling out simple annotation tasks to a large numbers of micro-workers. The added value of WebAnno, as compared to previous annotation tools, is on the one hand its web-based interface targeted at skilled as well as unskilled annotators, which unlocks a potentially very large workforce. On the other hand, it is the support for quality control, annotator management, and adjudication/curation, which lowers the entrance barrier for new annotation projects. We created WebAnno to fulfill the following requirements:

- *Web-based:* Distributed work, no installation effort, increased availability.
- *Interface to crowdsourcing:* unlocking a very large distributed workforce.
- *Quality and user management:* Integrated different user roles support (administrator, annotator, and curator), inter-annotator agreement measurement, data curation, and progress monitoring.
- *Flexibility:* Support of multiple annotation layers, pluggable import and export formats, and extensibility to other front ends.
- *Pre-annotated and un-annotated documents:* supporting new annotations, as well as manual corrections of existing, possibly automatic annotations.
- *Permissive open source:* Usability of our tool in future projects without restrictions, under the Apache 2.0 license.

In the following section, we revisit related work on annotation tools, which only partially fulfill the aforementioned requirements. In Section 3, the architecture as well as usage aspects of our tool are lined out. The scope and functionality summary

of WebAnno is presented in Section 4. Section 5 elaborates on several use cases of WebAnno, and Section 6 concludes and gives an outlook to further directions.

2 Related Work

GATE Teamware (Bontcheva et al., 2010) is probably the tool that closely matches our requirements regarding quality management, annotator management, and support of a large set of annotation layers and formats. It is mostly web-based, but the annotation is carried out with locally downloaded software. An interface to crowdsourcing platforms is missing. The GATE Teamware system is heavily targeted towards template-based information extraction. It sets a focus on the integration of automatic annotation components rather than on the interface for manual annotation. Besides, the overall application is rather complex for average users, requires considerable training and does not offer an alternative simplified interface as it would be required for crowdsourcing.

General-purpose annotation tools like MMAX2 (Müller and Strube, 2006) or WordFreak (Morton and LaCivita, 2003) are not web-based and do not provide annotation project management. They are also not sufficiently flexible regarding different annotation layers. The same holds for specialized tools for single annotation layers, which we cannot list here for the sake of brevity.

With the *brat rapid annotation tool* (Stenetorp et al., 2012), for the first time a web-based open-source annotation tool was introduced, which supports collaborative annotation for multiple annotation layers simultaneously on a single copy of the document, and is based on a client-server architecture. However, the current version of brat has limitations such as: (i) slowness for documents of more than 100 sentences, (ii) limits regarding file formats, (iii) web-based configuration of tagsets/tags is not possible and (iv) configuring the display of multiple layers is not yet supported. While we use brat's excellent visualization front end in WebAnno, we decided to replace the server layer to support the user and quality management, and monitoring tools as well as to add the interface to crowdsourcing.

3 System Architecture of WebAnno

The overall architecture of WebAnno is depicted in Figure 1. The modularity of the architecture,

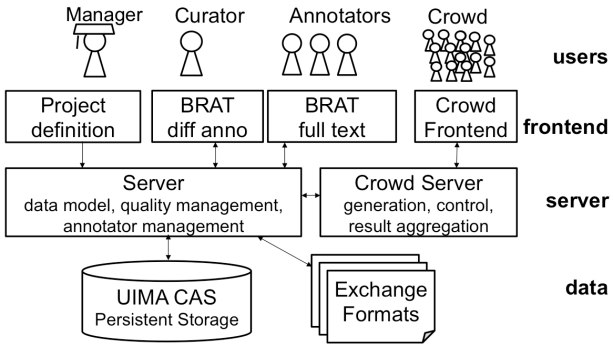

Figure 1: System architecture, organized in user, front end, back end and persistent data storage.

which is mirrored in its open-source implementation[1], makes it possible to easily extend the tool or add alternative user interfaces for annotation layers that brat is less suited for, e.g. for constituent structure. In Section 3.1, we illustrate how different user roles are provided with different graphical user interfaces, and show the expressiveness of the annotation model. Section 3.2 elaborates on the functionality of the back end, and describes how data is imported and exported, as well as our implementation of the persistent data storage.

3.1 Front End

All functionality of WebAnno is accessible via a web browser. For annotation and visualization of annotated documents, we adapted the brat rapid annotation tool. Changes had to be made to make brat interoperate with the Apache Wicket, on which WebAnno is built, and to better integrate into the WebAnno experience.

3.1.1 Project Definition

The definition and the monitoring of an annotation project is conducted by a project manager (cf. Figure 1) in a project definition form. It supports creating a project, loading un-annotated or pre-annotated documents in different formats[2], adding annotator and curator users, defining tagsets, and configuring the annotation layers. Only a project manager can administer a project. Figure 2 illustrates the project definition page with the tagset editor highlighted.

[1] Available for download at (this paper is based on v0.3.0): `webanno.googlecode.com/`

[2] Formats: plain text, CoNLL (Nivre et al., 2007), TCF (Heid et al., 2010), UIMA XMI (Ferrucci and Lally, 2004)

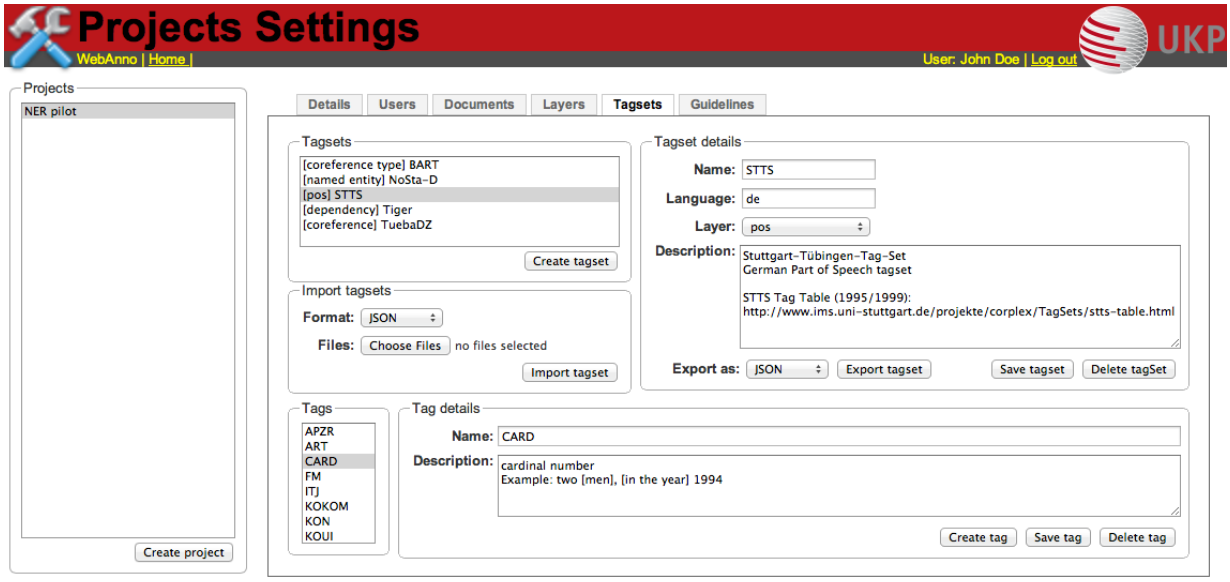

Figure 2: The tagset editor on the project definition page

3.1.2 Annotation

Annotation is carried out with an adapted version of the brat editor, which communicates with the server via Ajax (Wang et al., 2008) using the JSON (Lin et al., 2012) format. Annotators only see projects they are assigned to. The annotation page presents the annotator different options to set up the annotation environment, for customization:

- *Paging:* For heavily annotated documents or very large documents, the original brat visualization is very slow, both for displaying and annotating the document. We use a paging mechanism that limits the number of sentences displayed at a time to make the performance independent of the document size.
- *Annotation layers:* Annotators usually work on one or two annotations layers, such as part-of-speech and dependency or named entity annotation. Overloading the annotation page by displaying all annotation layers makes the annotation and visualization process slower. WebAnno provides an option to configure visible/editable annotation layers.
- *Immediate persistence:* Every annotation is sent to the back end immediately and persisted there. An explicit interaction by the user to save changes is not required.

3.1.3 Workflow

WebAnno implements a simple workflow to track the state of a project. Every annotator works on a separate version of the document, which is set to the state *in progress* the first time a document is opened by the annotator. The annotator can then mark it as *complete* at the end of annotation at which point it is locked for further annotation and can be used for curation. Such a document cannot be changed anymore by an annotator, but can be used by a curator. A curator can mark a document as *adjudicated*.

3.1.4 Curation

The curation interface allows the curator to open a document and compare annotations made by the annotators that already marked the document as *complete*. The curator reconciles the annotation with disagreements. The curator can either decide on one of the presented alternatives, or freely re-annotate. Figure 3 illustrates how the curation interface detects sentences with annotation disagreement (left side of Figure 3) which can be used to navigate to the sentences for curation.

3.1.5 Monitoring

WebAnno provides a monitoring component, to track the progress of a project. The project manager can check the progress and compute agreement with Kappa and Tau (Carletta, 1996) measures. The progress is visualized using a matrix of annotators and documents displaying which documents the annotators have marked as *complete* and which documents the curator *adjudicated*. Figure 4 shows the project progress, progress of individual annotator and completion statistics.

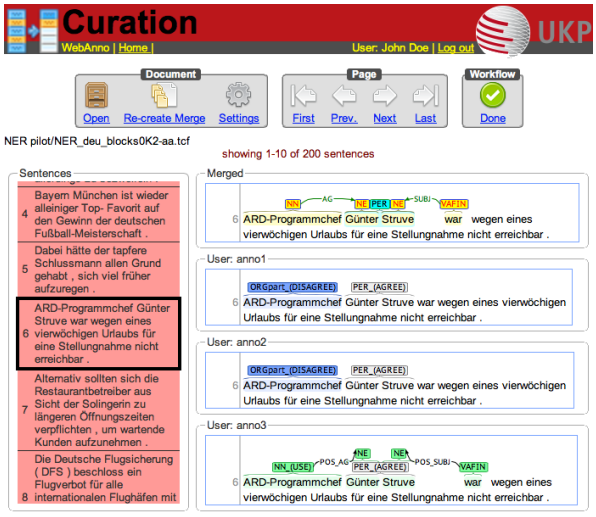

Figure 3: Curation user interface (left: sentences with disagreement; right: merging editor)

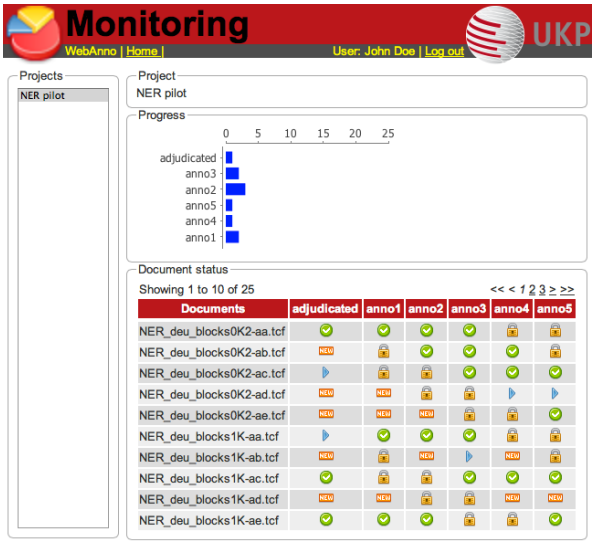

Figure 4: Project monitoring

3.1.6 Crowdsourcing

Crowdsourcing is a way to quickly scale annotation projects. Distributing a task that otherwise will be performed by a controlled user group has become much easier. Hence, if quality can be ensured, it is an alternative to high quality annotation using a large number of arbitrary redundant annotations (Wang et al., 2013). For WebAnno, we have designed an approach where a source document is split into small parts that get presented to micro-workers in the CrowdFlower platform[3]. The crowdsourcing component is a separate module that handles the communication via CrowdFlower's API, the definition of test items and job parameters, and the aggregation of results. The crowdsourced annotation appears as a virtual annotator in the tool.

Since it is not trivial to express complex annotation tasks in comparably simple templates suitable for crowdsourcing (Biemann, 2013), we proceed by working out crowdsourcing templates and strategies per annotation layer. We currently only support named entity annotation with predefined templates. However, the open and modular architecture allows to add more crowdsourced annotation layers.

3.2 Back End

WebAnno is a Java-based web application that may run on any modern servlet container. In memory and on the file system, annotations are stored as UIMA CAS objects (Ferrucci and Lally, 2004). All other data is persisted in an SQL database.

3.2.1 Data Conversion

WebAnno supports different data models that reflect the different communication of data between the front end, back end, and the persistent data storage. The brat data model serves exchanging data between the front end and the back end.

The documents are stored in their original formats. For annotations, we use the type system from the DKPro Core collection of UIMA components (Eckart de Castilho and Gurevych, 2009)[4]. This is converted to the brat model for visualization. Importing documents and exporting annotations is implemented using UIMA reader and writer components from DKPro Core as plug-ins. Thus, support for new formats can easily be added. To provide quick reaction times in the user interface, WebAnno internally stores annotations in a binary format, using the *SerializedCasReader* and *SerializedCasWriter* components.

3.2.2 Persistent Data Storage

Project definitions including project name and descriptions, tagsets and tags, and user details are kept in a database, whereas the documents and annotations are stored in the file system. WebAnno supports limited versioning of annotations, to protect against the unforeseen loss of data. Figure 5 shows the database entity relation diagram.

[3] www.crowdflower.com

[4] code.google.com/p/dkpro-core-asl/

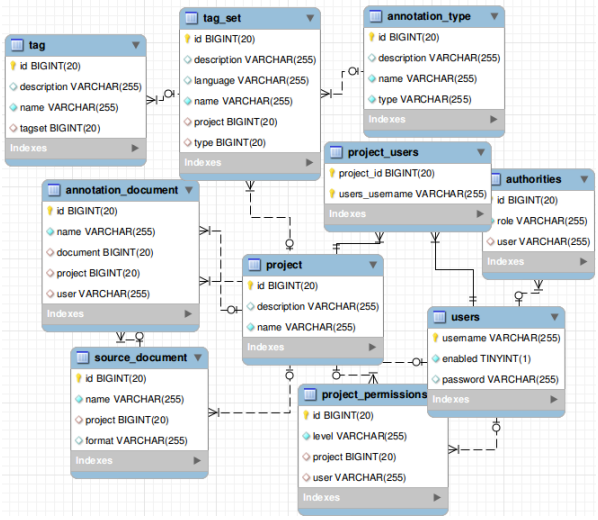

Figure 5: WebAnno database scheme

4 Scope and Functionality Summary

WebAnno supports the production of linguistically annotated corpora for different natural language processing applications. WebAnno implements ease of usage and simplicity for untrained users, and provides:

- Annotation via a fast, and easy-to-use web-based user interface.
- Project and user management.
- Progress and quality monitoring.
- Interactive curation by adjudicating disagreeing annotations from multiple users.
- Crowdsourcing of annotation tasks.
- Configurable annotation types and tag sets.

5 Use Cases

WebAnno currently allows to configure different span and arc annotations. It comes pre-configured with the following annotation layers from the DKPro Core type system:

Span annotations

- *Part-of-Speech (POS) tags*: an annotation task on tokens. Currently, POS can be added to a token, if not already present, and can be modified. POS annotation is a prerequisite of dependency annotation (Figure 6).

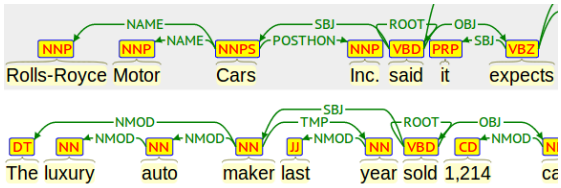

Figure 6: Parts-of-speech & dependency relations

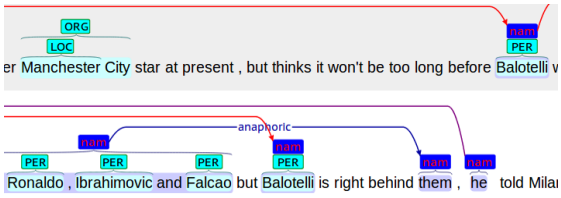

Figure 7: Co-reference & named entites

- *Named entities*: a multiple span annotation task. Spans can cover multiple adjacent tokens, nest and overlap (Figure 7), but cannot cross sentence boundaries.

Arc Annotations

- *Dependency relations*: This is an arc annotation which connects two POS tag annotations with a directed relation (Figure 6).
- *Co-reference chains*: The co-reference chain is realized as a set of typed mention spans linked by typed co-reference relation arcs. The co-reference relation annotation can cross multiple sentences and is represented in co-reference chains (Figure 7).

The brat front end supports tokens and sub-tokens as a span annotation. However, tokens are currently the minimal annotation units in WebAnno, due to a requirement of supporting the TCF file format (Heid et al., 2010). Part-of-speech annotation is limited to singles token, while named entity and co-reference chain annotations may span multiple tokens. Dependency relations are implemented in such a way that the arc is drawn from the governor to the dependent (or the other way around, configurable), while co-reference chains are unidirectional and a chain is formed by referents that are transitively connected by arcs.

Based on common practice in manual annotation, every user works on their own copy of the same document so that no concurrent editing occurs. We also found that displaying all annotation layers at the same time is inconvenient for annotators. This is why WebAnno supports showing

and hiding of individual annotation layers. The WebAnno curation component displays all annotation documents from all users for a given source document, enabling the curator to visualize all of the annotations with differences at a time. Unlike most of the annotation tools which rely on configuration files, WebAnno enables to freely configure all parameters directly in the browser.

6 Conclusion and Outlook

WebAnno is a new web-based linguistic annotation tool. The brat annotation and GUI front end have been enhanced to support rapidly processing large annotation documents, configuring the annotation tag and tagsets in the browser, specifying visible annotation layers, separating annotation documents per user, just to name the most important distinctions. Besides, WebAnno supports project definition, import/export of tag and tagsets. Flexible support for importing and exporting different data formats is handled through UIMA components from the DKPro Core project. The monitoring component of WebAnno helps the administrator to control the progress of annotators. The crowdsourcing component of WebAnno provides a unique functionality to distribute the annotation to a large workforce and automatically integrate the results back into the tool via the crowdsourcing server. The WebAnno annotation tool supports curation of different annotation documents, displaying annotation documents created by users in a given project with annotation disagreements. In future work, WebAnno will be enhanced to support several other front ends to handle even more annotation layers, and to provide more crowdsourcing templates. Another planned extension is a more seamless integration of language processing tools for pre-annotation.

Acknowledgments

We would like to thank Benjamin Milde and Andreas Straninger, who assisted in implementing WebAnno, as well as Marc Reznicek, Nils Reiter and the whole CLARIN-D FAG 7 for testing and providing valuable feedback. The work presented in this paper was funded by a German BMBF grant to the CLARIN-D project, the Hessian LOEWE research excellence program as part of the research center "Digital Humanities" and by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program under grant No. I/82806.

References

Chris Biemann. 2013. Creating a system for lexical substitutions from scratch using crowdsourcing. *Lang. Resour. Eval.*, 47(1):97–122, March.

Kalina Bontcheva, Hamish Cunningham, Ian Roberts, and Valentin Tablan. 2010. Web-based collaborative corpus annotation: Requirements and a framework implementation. In *New Challenges for NLP Frameworks workshop at LREC-2010*, Malta.

Jean Carletta. 1996. Assessing agreement on classification tasks: the kappa statistic. In *Computational Linguistics, Volume 22 Issue 2*, pages 249–254.

Richard Eckart de Castilho and Iryna Gurevych. 2009. DKPro-UGD: A Flexible Data-Cleansing Approach to Processing User-Generated Discourse. In *Online-proceedings of the First French-speaking meeting around the framework Apache UIMA*, LINA CNRS UMR 6241 - University of Nantes, France.

David Ferrucci and Adam Lally. 2004. UIMA: An Architectural Approach to Unstructured Information Processing in the Corporate Research Environment. In *Journal of Natural Language Engineering 2004*, pages 327–348.

Ulrich Heid, Helmut Schmid, Kerstin Eckart, and Erhard Hinrichs. 2010. A Corpus Representation Format for Linguistic Web Services: the D-SPIN Text Corpus Format and its Relationship with ISO Standards. In *Proceedings of LREC 2010*, Malta.

Boci Lin, Yan Chen, Xu Chen, and Yingying Yu. 2012. Comparison between JSON and XML in Applications Based on AJAX. In *Computer Science & Service System (CSSS), 2012*, Nanjing, China.

Thomas Morton and Jeremy LaCivita. 2003. WordFreak: an open tool for linguistic annotation. In *Proceedings of NAACL-2003*, NAACL-Demonstrations '03, pages 17–18, Edmonton, Canada.

Christoph Müller and Michael Strube. 2006. Multi-level annotation of linguistic data with MMAX2. In S. Braun, K. Kohn, and J. Mukherjee, editors, *Corpus Technology and Language Pedagogy: New Resources, New Tools, New Methods*, pages 197–214. Peter Lang, Frankfurt a.M., Germany.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 Shared Task on Dependency Parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a Web-based Tool for NLP-Assisted Text Annotation. In *Proceedings of the Demonstrations at EACL-2012*, Avignon, France.

Qingling Wang, Qin Liu, Na Li, and Yan Liu. 2008. An Automatic Approach to Reengineering Common Website with AJAX. In *4th International Conference on Next Generation Web Services Practices*, pages 185–190, Seoul, South Korea.

Aobo Wang, Cong Duy Vu Hoang, and Min-Yen Kan. 2013. Perspectives on Crowdsourcing Annotations for Natural Language Processing. In *Language Resources And Evaluation*, pages 9–31. Springer Netherlands.