Inforex – a web-based tool for text corpus management and semantic annotation

Michał Marcińczuk, Jan Kocoń, Bartosz Broda

Wrocław University of Technology, Wrocław, Poland

michal.marcinczuk@pwr.wroc.pl, janek.kocon@gmail.com, bartosz.broda@pwr.wroc.pl

Abstract

The aim of this paper is to present a system for semantic text annotation called Inforex. Inforex is a web-based system designed for managing and annotating text corpora on the semantic level including annotation of Named Entities (NE), anaphora, Word Sense Disambiguation (WSD) and relations between named entities. The system also supports manual text clean-up and automatic text pre-processing including text segmentation, morphosyntactic analysis and word selection for word sense annotation.

Keywords: corpus management, corpus annotation, bootstrapping, Inforex

1. Introduction

Large text corpora are central in statistical-based Natural Language Processing (NLP) (Manning and Schütze, 2001). One can find many approaches based on supervised Machine Learning (ML) to solve NLP-related problems in the literature. For training ML algorithms a manually annotated corpus is needed. That is, a domain expert have to mark certain parts of the text with appropriate labels. The annotation process is usually hard, costly and time consuming. The problem is even more pronounced when multiple people are working simultaneously on the same corpus. However, usage of supporting Language Technology (LT) can improve the process of manual corpus annotation considerably. In this paper we describe Inforex – an example of LT that helps in this process.

Inforex is a web-based system for text corpora management and semantic annotation. The construction of the system started in early 2010, at the beginning of NEKST project. At that time we needed to gather and prepare data for the task of named entity recognition (Marcińczuk and Piasecki, 2011). In the second half of the year another project started called SyNaT. One of the tasks of the project was to build a manually annotated corpus with semantic information. New requirements emerged and we decided to extend our system with the new functionality. The system was also used in another projected started in the beginning of 2011 (Marcinczuk et al., 2011) in construction of a Polish Corpus of Suicide Notes (PCSN).

We decided to construct a system from scratch because we couldn't find system that: (a) is an open-source and freely available, (b) is platform independent, (c) store all the data (text and annotations) in a central repository integrated with the application, (d) provide transparent deployment of new versions of the system, (e) can be run on any computer without the need of downloading and installing additional software.

This paper is organised as follows: we start with description of existing systems for corpora annotation. Next, the description of Inforex (Sec. 3.) and annotation workflow (Sec. 4.) is given. Detailed description of task supported by Inforex is given in Section 5. The paper is finished with brief discussion of licensing status (Sec. 7.), system applications (Sec. 6.) and conclusions in Sec. 8.

2. Existing Annotation Environments

As a corpus annotation is not a new NLP task some systems have already been build. Before making a decision to develop Inforex form scratch we had investigated several existing systems. Most of the system have some severe limitation in terms of our requirements. Nevertheless, the analysis helped us in refining our design goals and architecture of Inforex. The list of exterminated systems includes:

- *GATE* (Cunningham et al., 2011) is widely-known and used system for corpus management and text annotation that is being developed over 15 years. It is a desktop application written in Java and can be run under almost any operating systems. It provides many of functionality we required, but we did not decide to use it because we stumbled upon many problem while developing a java-based desktop application for wordnet construction called WordnetLoom (Piasecki et al., 2011). Among the decisive factors were frequent upgrades which are very inconvenient for the users and issues with rapid bug-reproduction on developers' computers leading to high cost of bug-fixing.

- *Manufakturzysta 2.0 Luna* (Marcińczuk, 2010) is a desktop application written in C# that was used to annotate transcriptions of phone calls within the LUNA project (Mykowiecka et al., 2010). The system was designed to annotate the text on the semantic level including named entities and binary relations between the entities. The system works only with Windows operating system and it does not support parallel access to central data by different users — every instance of the application works on local data.

- *GATE Teamware* (LLC, 2010) is a web-based version of GATE also implemented in Java. Information about the system was available since 2010, but the source

code was published after the development of Inforex was started.

- *Annotatornia* (Przepiórkowski and Murzynowski, 2009) is a web-based system developed to annotate text on four levels: word-level segmentation, sentence-level segmentation, morphosyntax and WSD. Annotation of named entities, binary relations and events was not included. Implementation started in 2009 and the source code was published in July 2010.

3. Inforex Characteristic

Inforex can be accessed from any standard-compliant web browser supporting JavaScript.[1] The user interface has a form of dynamic HTML pages using the AJAX technology. The server part of the system is written in PHP and the data is stored in MySQL database. The system make use of some external tools that are installed on the server or can be accessed via web services.

The documents are stored in the database in the original format — either plain text, XML or HTML. Tokenization and sentence segmentation is optional and is stored in a separate table. Tokens are stored as pairs of values representing indexes of first and last character of the tokens and sets of features representing the morpho-syntactic information. Annotations[2] created by user are stored in the same way as tokens (pair of character indexes) but in additional table. Character indexes omit all the white spaces and XML/HTML tags. In addition, HTML entities are counted as one character.

4. Annotation Workflow

The corpus annotation workflow in Inforex starts with the creation and configuration of a new corpus. This involves definition of subcorpora, flags (described in the next paragraph), selection of document perspectives, selection of existing or creation of new schemas of annotations and relations and uploading documents. When the corpus configuration is set up one can add new or existing users and grant access and permissions to the document perspectives. Users, that have appropriate permissions can perform certain actions. When user logs in to the system he or she sees only the corpora that were assigned to her/him or are public.

Flags (see Figure 1), that were mentioned in the previous paragraph, are used to track work progress. The mechanism allows to define a set of named flags that can be used to describe work state of every document within given corpus. Every flag can be set to one of several predefined states, i.e., *not ready*, *ready to process*, *being processed*, *ready to check*, *need correction*, *checked*.

[1] In practice we have only enough resources to properly test the system in Firefox. Thus, some of the complex dynamic functions might not work properly under other web browsers.

[2] Annotation is understood as a label attached to a continuous piece of text. Additional information can be attached to the annotation as a pair of strings: {argument; value}.

5. Tasks

This section presents how the system supports different kind of tasks related to the corpus construction.

5.1. Document Browsing

The XML tags in the document are used to encode the document structure. While browsing they are not displayed to the user directly but influence how the text blocks are displayed on the screen. The HTML tags (h1, em, p, li, etc.) are displayed in a default way. For custom tags user can define the formatting using CSS.

5.2. Document Content Edition

The common operation that is performed on every document is its clean up. The documents can be edited in the *Edit Content* perspective. Every modification of the content is tracked by the system and a difference with previous version is generated and stored in the database. In addition, user can add a comment in order to motivate the introduced modification.

A complete revision of the document versions is presented in the *History of Changes* perspective (see Figure 2). Every modification is displayed as a diff with previous version with date, time, user name and user comment. This mechanism is used to track back potential errors introduced during document clean up.

As the annotations are stored as a pairs of character indexes representing the annotation boundary the modification of annotated documents needs special treatment. If a document contains annotations, special markers are inserted into the document content to indicate the annotation boundaries. When the document content is changed Inforex automatically calculates the changes in annotations (and possible deletions of annotations). The user sees the list of changes that will be applied to the document and can either reject or confirm them. The users can also backtrack to document editing.

5.3. Document Segmentation

Document tokenization is stored independently from the document content in the same way as annotations — pairs of character indexes representing tokens range. The sentence segmentation is indirectly based on characters. Sentence boundaries are stored together with tokenization by marking tokens ending sentences.

For Polish two tokenizers were integrated with the system. The first one is accessible, directly through the *Tokenization* perspective utilizing a *TaKIPI-WS* web service (Broda et al., 2010b). The other tool, *maca* (Radziszewski and Śniatowski, 2011), is executed as a batch script that can reside on external server. The script inserts the tokenization directly into the database through the API provided by Inforex. Using such an approach we don't tie Inforex to one tokenization schema as any external tool can be utilised for this purpose.

The current version of the perspective does not allow to modify the segmentation by hand. However, after recurring reports from linguists about errors in the automatic segmentation that introduce problems during the annotation we de-

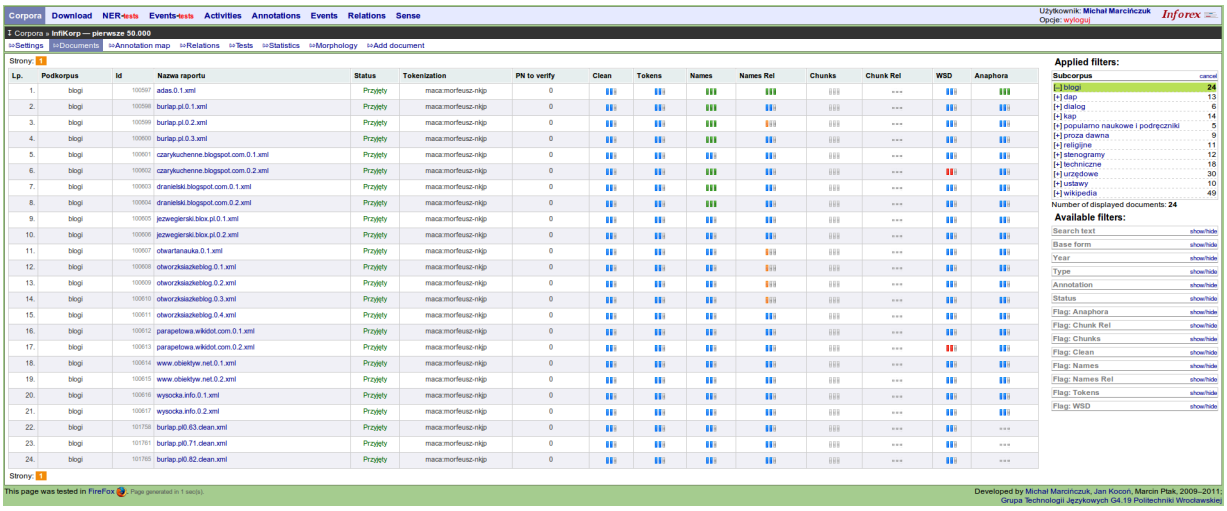

Figure 1: *List of documents* — contains basic information about the documents (left part of the table) and flags indicating the document work progress (right part of the table).

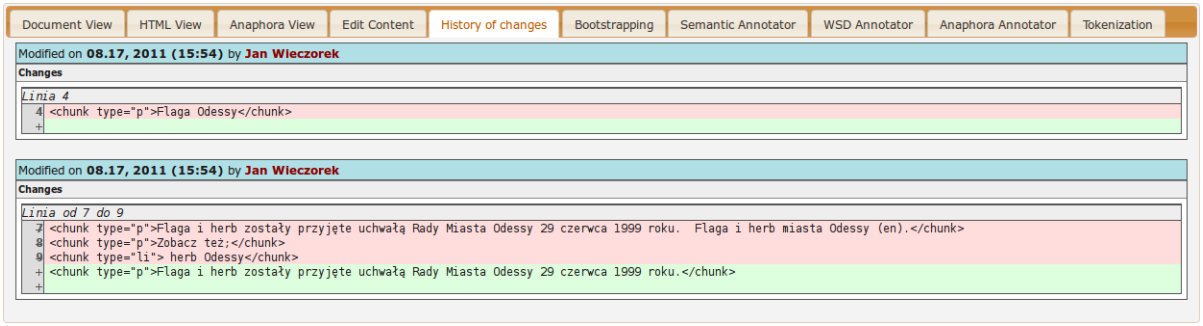

Figure 2: *History of Changes* — perspective used to track changes in the document content.

cided to extend this perspective and allow users to modify the sentence boundaries.

5.4. Named Entities Annotation

Annotation of named entities is an example of annotation-based tasks, i.e., a tasks which goal is to assign a set of predefined labels to the text. The annotation is performed within the *Semantic Annotator* perspective (see Figure 3). It was challenging to design and to develop a HTML-based interface for text annotation. The following questions had to be answered:

1. How to display annotations in a formatted HTML document in a compact way?
2. How to organize annotation of tokenized and not tokenized texts?
3. How to handle nested, overlapping and discontinuous annotations?
4. How to simplify, support and automate creation of common annotations?

We wanted to display the annotations in a compact way because we wanted to fit as much text on the screen as possible — some annotation tasks will require access to wide document context. The best way was to display the annotations as HTML formatted tags (i.e., `span` elements). However, this solution has some limitations, i.e., a problem with displaying overlapping and discontinuous annotations (nested annotations can be easily displayed with this approach).

To solve the problem with overlapping annotations we assumed, that annotations within the same group of annotations (layers) cannot overlap. Annotations from different groups can overlap but cannot be displayed in the same panel at the same time. In order to display annotations from overlapping groups the screen is split, forming *twin panels* (see Figure 4). The idea was to display the same document in two parallel panels and allow user to choose which group of annotations should be display in each panel. This way overlapping groups of annotations are displayed side-by-side. In addition, user can show/hide selected subgroups of annotations.

To solve the problem with discontinuous annotations we decided to use relations mechanism (described in Section 5.7.). Every continuous part of annotation is represented by a single annotation. Then, all the annotations are connected with a special type of relation in a continuous chain. That is, first part with the second one, second one with the third one, and so on.

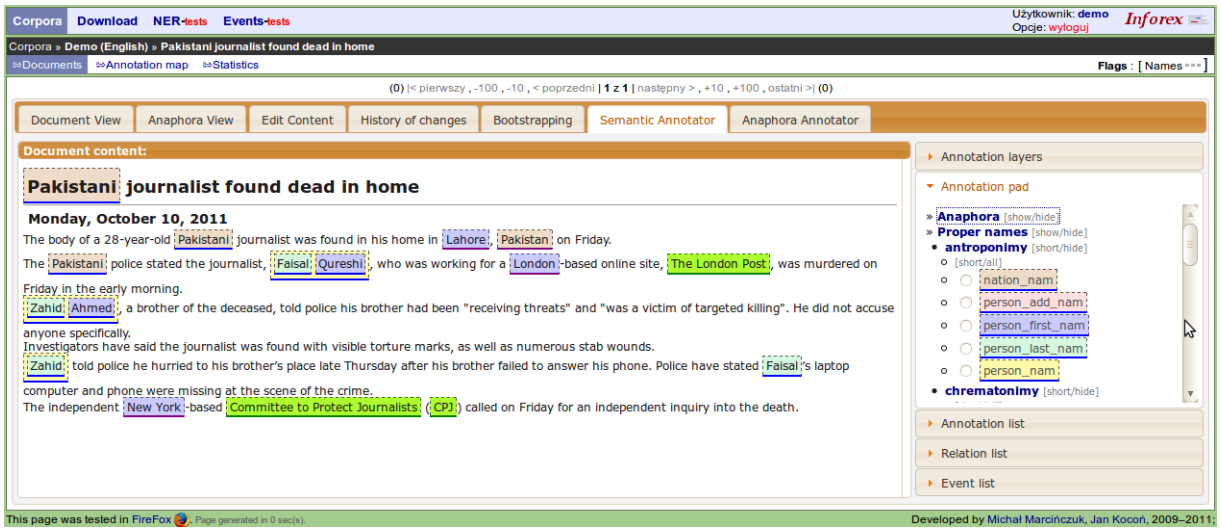

Figure 3: *Semantic Annotator* — perspective used to create, modify and delete semantic annotations.

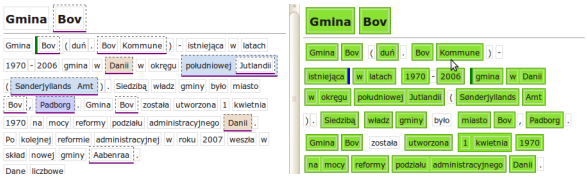

Figure 4: Twin-panel with *named entities* on the left and *agreement chunks* on the right.

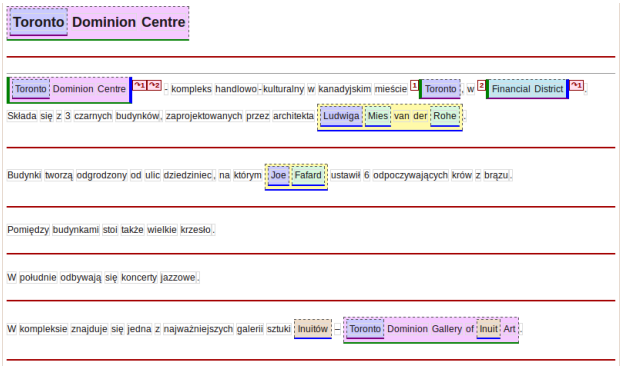

Figure 5: Sentence segmentation highlight mode.

During annotation, if the word segmentation is provided the system automatically expand text selection to capture whole tokens. The process of annotation is supported in three additional ways:

- *quick mode* — in the quick mode user selects one type of annotation and then after every selection of text this annotation type is automatically added (in normal mode user have to choose annotation type after every text selection),
- *common annotations* — allows to display selected types of annotations instead of full list of annotations. It is useful for groups with lots of rare annotation types which would require lot of scrolling.
- *sentence segmentation highlight* — allows to display every sentence starting from a new line and separated by a horizontal line to clearly indicate the sentence boundaries (see Figure 5). This mode is useful in sentence-context annotation tasks. For example, syntactic chunks cannot cross sentence boundaries and semantic relations between proper names are contained within one sentence.

5.5. Annotation Bootstrapping

Bootstrapping perspective (see Figure 6) allows to run external module to recognize named entities and to verify the results of automatic recognition. The automatically recognized annotations are presented to the user for the verification. For every proposition the user can choose one of four options: *accept* if the annotation is correct, *discard* if the annotation is incorrect (the annotation border is incorrect), *change annotation type* if the annotation border is correct but the annotation type is wrong and the last option *later* leaves the proposition unchanged for later verification. The missing annotations (not recognized in bootstrapping) must be added manually in the Semantic Annotator perspective. The discarded annotations are stored in the database to prevent the system from repeating wrong decisions. Storing mistakes of the system also enables calculation of the performance of the bootsrapping module.

5.6. Word Sense Annotation

The perspective for word sense annotation (*WSD Annotator*) was based on system presented in (Broda et al., 2010c). The perspective consists of three parts: (1) a list of words to be annotated, (2) a document view with marked words for annotation, (3) a list of senses for selected word. The perspective allows user to browse the instances of selected word in a predefined order or to jump directly to a first not annotated word.

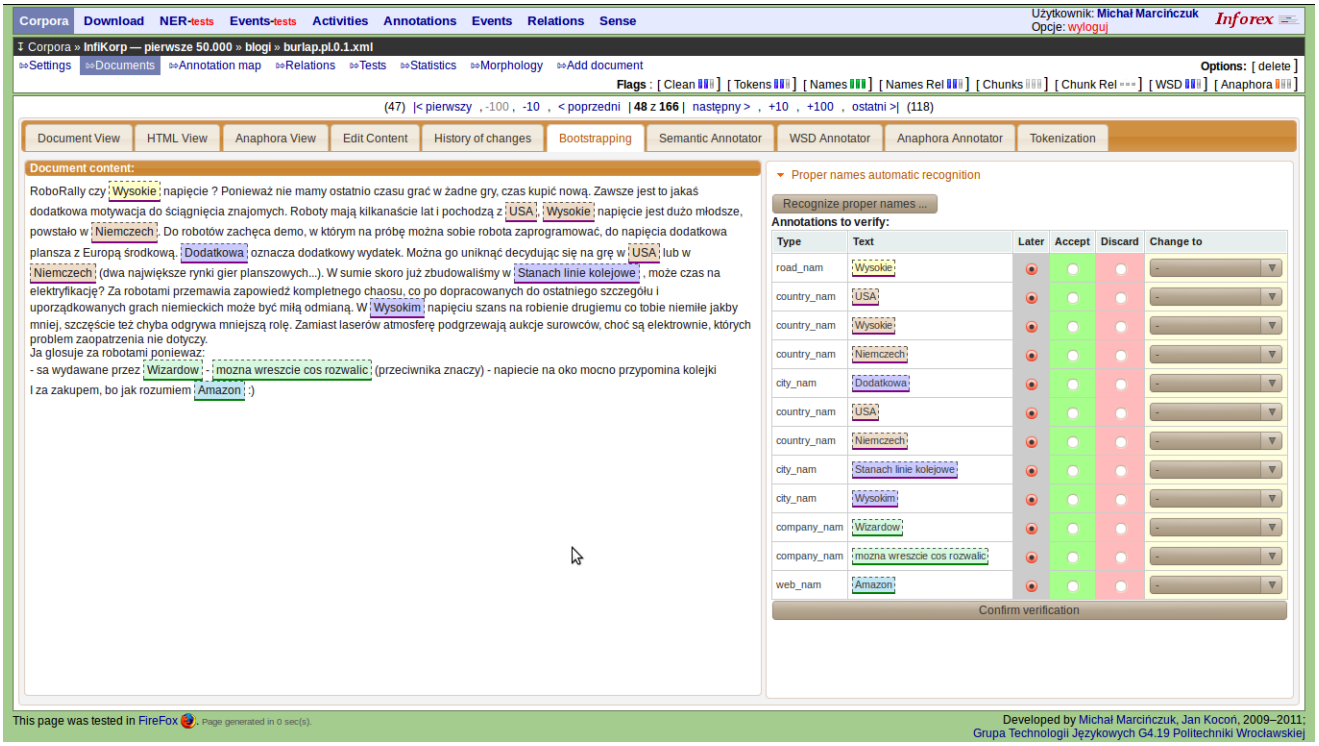

Figure 6: *Bootstrapping* — perspective for manual verification of bootstrapped annotation.

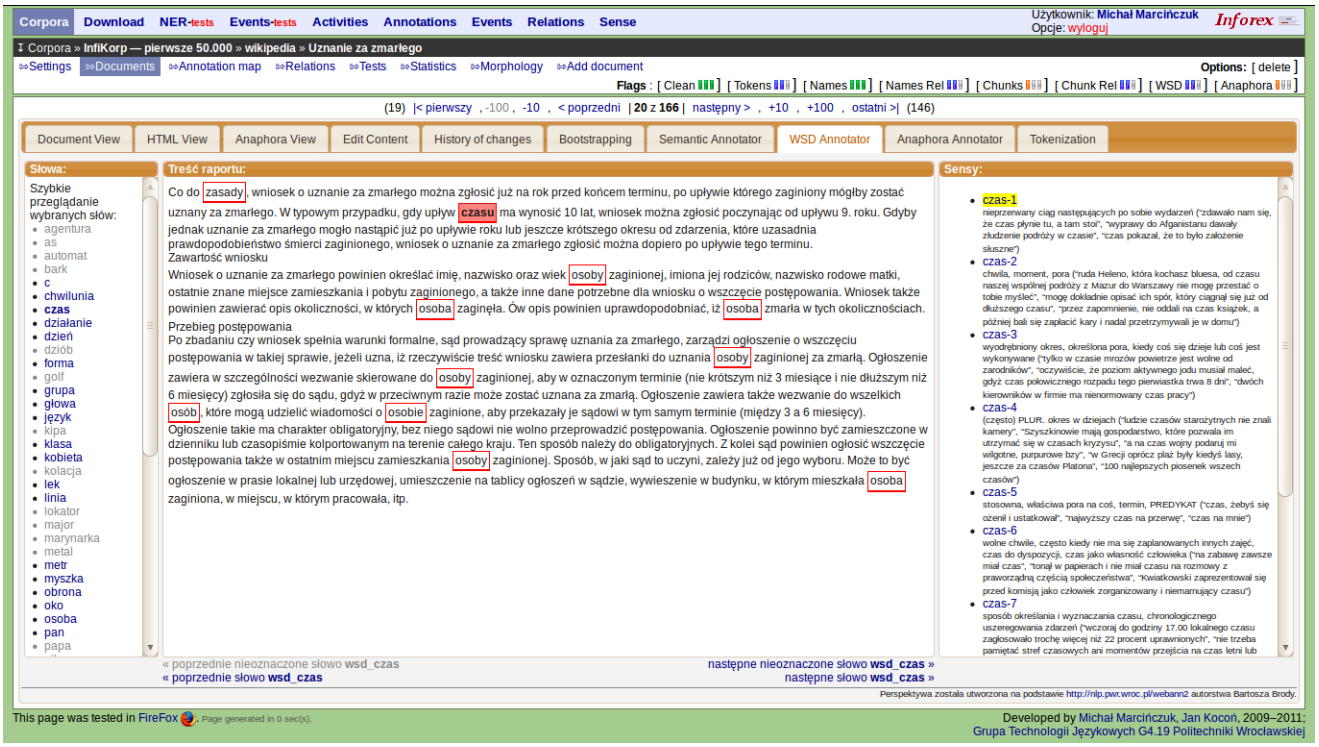

Figure 7: *WSD Annotator* — perspective for word sense annotation.

5.7. Relation Annotation

Annotation of relations is performed in the *Semantic Annotator* perspective. Relations can be created between any types of annotations according to a predefined schema. The schema defines groups of relations, annotation layers to which the relations are assigned and constrains on annotation types that can be connected with given relation. The constraints can be set on the level of annotation layers, annotation groups and single annotation types.

5.8. Anaphora Annotation

Anaphora is a kind of relation that connect two elements. In general, anaphora could be annotated using general mechanism for relations. However, the number of operations required to create an anaphora relation is too large. In order to simplify and speed up annotation of anaphora a dedicated perspective was designed and implemented, namely *Anaphora Annotator* (see Figure 8). The perspective consists of three parts: (1) left part is a document view with tokenization, (2) middle part is a document view with selected named entities and (3) right part with a list of anaphora types. The process of creating a new relation requires three operations: (1) selection of a source word or named entity, (2) selection of a target named entity and (3) selection of anaphora type.

5.9. Annotation of Events

Annotation of events can be done in the *Semantic Annotator* perspective. Events are defined as set of pairs {attribute; value}. *attribute* is a name of slot defined in the schema, and *value* is an annotation of defined type or category. One can add several types of events to one document. For every created event user can add several slots, and for every slot one annotation can be selected.

5.10. Data Export

The document content, tokenization, sentence segmentation, annotations (syntactic chunks, proper names, WSD) and relations between annotations (syntactic relations between chunks, semantic relations between named entities and anaphora) can be exported to a XML-like corpus format called CCL. The CCL format is based on XCES (Ide et al., 2000) with a few simple extensions that enables simple encoding of all the required annotation levels.

6. Applications

Inforex is being used to construct and annotate corpora within three ongoing projects:

- NEKST[3] — two corpora of Polish stock exchange reports (1215 documents) and economic news from Polish Wikinews (797 documents) annotated with named entities (Marcińczuk and Piasecki, 2011);
- SyNaT[4] — a Wrocław University of Technology Corpus (KPWr; *Korpus Politechniki Wrocławskiej*) containing samples of documents from various domains (blogs, science, stenographic recordings, dialogue, contemporary prose, etc.) annotated with named entities, semantic chunks, word senses, syntactic relations between chunks, semantic relations between named entities and anaphora relations (Broda et al., 2010a). At the moment of writing the corpus consists of more than 1300 documents;
- PCSN[5] — a Polish Corpus of Suicide Notes annotated with named entities, semantic and pragmatic information (Marcińczuk et al., 2011). At the moment of writing the corpus consists of 626 genuine suicide notes and 51 simulated suicide notes.

7. Access and License

Inforex is hosted at Wrocław University of Technology and is available at the following address `http://nlp.pwr.wroc.pl/inforex`. To test the major features of the application one can login using demo account (user and password are `demo`).
We plan to release the source code of Inforex on a free license as soon as the system will be tested enough and will be relatively stable. The source code and further information will be posted on the Inforex web page.

8. Conclusion

Inforex is a web-based system for semantic annotation of text corpora. Major functions of the system are already implemented and used in couple projects by several users. However, the system is still under development and new features are being added when required. The list of features to be implemented contains for example a perspective to fix the automatic sentence-level segmentation.

Acknowledgements

Work co-financed by Innovative Economy Programme project POIG.01.01.02-14-013/09 (`http://www.ipipan.waw.pl/nekst/`) and NCBiR NrU.: SP/I/1/77065/10 (`http://www.synat.pl`).

9. References

Bartosz Broda, Michał Marcińczuk, Marek Maziarz, Adam Radziszewski, and Adam Wardyński. 2010a. WUTC: Towards a Free Corpus of Polish. In *Proceedings of the Eighth Conference on International Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. May 23–25, 2012.

Bartosz Broda, Michał Marcińczuk, and Maciej Piasecki. 2010b. Building a Node of the Accessible Language Technology Infrastructure. In Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odjik, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta. May 19–21, 2010.

Bartosz Broda, Maziarz Maziarz, and Maciej Piasecki. 2010c. Evaluating LexCSD — a Weakly-Supervised Method on Improved Semantically Annotated Corpus in a Large Scale Experiment. In S. T. Wierzchoń M. A. Kłopotek, A. Przepiórkowski and K. Trojanowski, editors, *Proceedings of Intelligent Information Systems*.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan, Niraj Aswani, Ian Roberts, Genevieve Gorrell, Adam Funk, Angus Roberts, Danica Damljanovic, Thomas Heitz, Mark A. Greenwood, Horacio Saggion, Johann Petrak, Yaoyong Li, and Wim Peters. 2011. *Text Processing with GATE (Version 6)*.

[3]Project home page: `http://www.ipipan.waw.pl/nekst/`.

[4]Project home page: `http://www.synat.pl`.

[5]Project home page: `http://pcsn.uni.wroc.pl/`

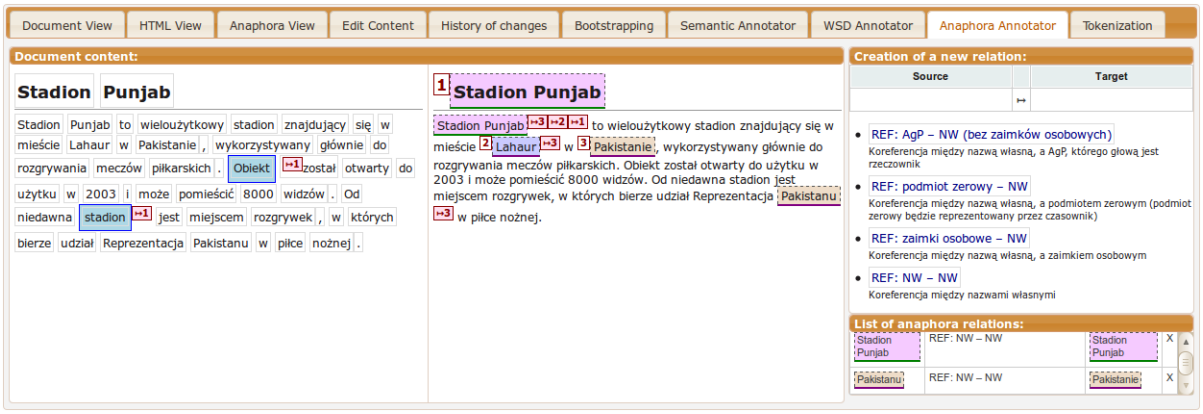

Figure 8: *Anaphora Annotator* — perspective used to create and delete anaphora between named entities and words.

Nancy Ide, Patrice Bonhomme, and Laurent Romary. 2000. Xces: An xml-based encoding standard for linguistic corpora. In *Proceedings of the Second International Language Resources and Evaluation Conference. Paris: European Language Resources Association*.

Fairview Research LLC. 2010. GATE Teamware 1.3 User Guide. Technical report.

C. D. Manning and H. Schütze. 2001. *Foundations of Statistical Natural Language Processing*. The MIT Press.

Michał Marcińczuk and Maciej Piasecki. 2011. Statistical Proper Name Recognition in Polish Economic Texts. *Control and Cybernetics*, 40(2).

Michal Marcinczuk, Monika Zasko-Zielinska, and Maciej Piasecki. 2011. Structure Annotation in the Polish Corpus of Suicide Notes. In *TSD*, pages 419–426.

Michał Marcińczuk, Monika Zaśko-Zielińska, and Maciej Piasecki. 2011. Structure Annotation in the Polish Corpus of Suicide Notes. In Vaclav Habernal, Ivan; Matousek, editor, *Text, Speech and Dialogue, 14th International Conference, TSD 2011*, volume 6836 of *Lecture Notes in Computer Science*. Springer.

Michał Marcińczuk. 2010. Manufakturzysta 2.0 Luna. Dokumentacja techniczna. Technical report.

Agnieszka Mykowiecka, Katarzyna Głowińska, and Joanna Rabiega-Wiśniewska. 2010. Domain-related Annotation of Polish Spoken Dialogue Corpus LUNA.PL. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may. European Language Resources Association (ELRA).

Maciej Piasecki, Michał Marcińczuk, Radosaw Ramocki, and Marek Maziarz. 2011. WordnetLoom: a Wordnet Development System Integrating Form-based and Graph-based Perspectives. *Int. J. of Data Mining, Modelling and Management*. To Appear.

Adam Przepiórkowski and Grzegorz Murzynowski. 2009. Manual annotation of the National Corpus of Polish with Anotatornia. In Stanisław Goźdź-Roszkowski, editor, *The proceedings of Practical Applications in Language and Computers PALC 2009*, Frankfurt am Main. Peter Lang. Forthcoming.

Adam Radziszewski and Tomasz Śniatowski. 2011. Maca — a configurable tool to integrate Polish morphological data. In *Proceedings of the Second International Workshop on Free/Open-SourceRule-Based Machine Translation*, Barcelona, Spain. Universitat Oberta de Catalunya. January 20, 2011.