A Lightweight Approach to Semantic Annotation of Research Papers

Nicola Zeni[1], Nadzeya Kiyavitskaya[1], Luisa Mich[2],
John Mylopoulos[1], James R. Cordy[3]

[1] Dept. of Information and Communication Technology,
University of Trento, Italy
{nadzeya, nzeni, john.mylopoulos}@dit.unitn.it

[2] Dept. of Computer and Management Sciences,
University of Trento, Italy
luisa.mich@unitn.it

[3] School of Computing, Queens University, Kingston, Canada
cordy@cs.queensu.ca

Abstract. This paper presents a novel application of a semantic annotation system, named Cerno, to analyze research publications in electronic format. Specifically, we address the problem of providing automatic support for authors who need to deal with large volumes of research documents. To this end, we have developed Biblio, a user-friendly tool based on Cerno. The tool directs the user's attention to the most important elements of the papers and provides assistance by generating automatically a list of references and an annotated bibliography given a collection of published research articles. The tool performance has been evaluated on a set of papers and preliminary evaluation results are promising. The backend of Biblio uses a standard relational database to store the results.

Keywords: bibliography generation, semantic annotation.

1 Introduction and Motivation

This paper presents a novel application of the lightweight semantic annotation method founded on the Cerno system [1,2] to the task of research publication analysis. At present, information mining for collections of published research papers is supported by search engines such as Google Scholar [3], the DBLP repository [4], and electronic libraries such as Citeseer [5], the ACM Portal [6], and the IEEE Digital Library [7]. These services provide access to large knowledge bases of research publications, allowing a user to search for a paper and retrieve the details of its publication. In digital libraries, it is also possible to see the abstract and citations present in the paper.

However, organization of a personal collection of electronic publications remains a manual task for authors. The authors of a new research paper are often forced to

spend time analyzing the content of accumulated related work to find information relevant to the results being published. Such effort is required for any kind of scientific publication (e.g., dissertation, technical report, book, or journal article) in order to complete a review of the state-of-the-art and provide an appropriate bibliography of related work. Obviously, authors could benefit from assistance in collecting, managing and analyzing relevant literature. Such assistance can come from tools that perform the analysis of relevant literature with "good enough" results. Such an analysis could also be used to provide other helpful services, such as – for example – the automatic construction of a bibliography for selected articles.

The difficulties posed by a research publications analysis task may not be obvious to the reader. Here are some of the issues that need to be addressed:

- *Variable document formats*. Document source files may be stored as PDF, MS Word, LaTeX, PostScript, HTML and other electronic formats;
- *Page layout*. Depending on the requirements of the publisher, the layout of a document varies; for example, pages may be organized in a one/two-column fashion; for papers published in journals, the header may be present on some pages; footnotes may be allowed or not, and so on;
- *Document structure*. At first sight, one may assume that scientific documents are semi-structured; this means that the arrangement of document elements can be predicted with some certainty: for instance, first there is a title, then a list of authors with affiliations, abstract, introduction and other sections; unfortunately, such structure is not universally accepted;
- *Semantic analysis*. The key elements that a reader (for example, a reviewer) looks for are the problem considered and the main contributions. The reader may also be interested in finding background knowledge on which the work is based, particularly references to related work;
- *Bibliography generation*. Normally, in order to generate each item of the bibliography, the writer has to manually find all bibliographical details of a selected document and fill the required template with this data.

In this work, we address all these issues and provide semantic and structural annotation of document sections. This paper presents Biblio, a tool that is based on Cerno and is intended to support the analysis of research articles. Cerno is a semantic annotation system based on software code analysis techniques and tools proven effective in the software analysis domain for processing billions of lines of legacy software source code [1]. Cerno has been already applied in several case studies involving the analysis of differently structured documents from the tourism sector [2,3]. The system has demonstrated good performance and scalability while yielding good quality results. In this work we also present preliminary experimental results of the technique on a set of published papers. Apart from semantic markup, Biblio also supports the generation of a bibliography in different formats.

The rest of the paper is structured as follows: Section 2 reviews the semantic annotation method of Cerno. Section 3 explains how this method can be adapted for the domain of electronic literature analysis annotation, providing some insights on the implementation details. Section 4 illustrates the document analysis process on a specific example. Section 5 provides results of a limited evaluation of Cerno-based annotation for a set of research papers, while section 6 surveys related work, and conclusions are drawn in Section 7.

2 Text Analysis with Cerno

Our document analysis is based on Cerno [1, 2], a semantic annotation system that utilizes highly efficient methods for text processing for software (code) analysis and reengineering. Cerno is based on TXL [8], a generalized parsing and structural transformation system. TXL is especially designed to allow rapid prototyping and processing of text patterns. The formal semantics and implementation of TXL are based on formal tree rewriting, but the trees are largely hidden from the user due to the by-example style of rule specification. The system accepts as input a grammar (set of text patterns) and a document, generates a parse tree for the input document, and applies transformation rules to generate output in a target format. TXL uses full backtracking with ordered alternatives and heuristic resolution, which allows efficient and flexible parsing.

The architecture of Cerno consists of a series of components that perform sequential transformations on an input document:

1. *Parse*. First, the system breaks down raw input text into its constituents, producing a parse tree of an input document. The peculiarity of parsing textual documents with Cerno, compared to design recovery of source code, is that in this case, the parse tree is composed of such structures as *document, paragraph, phrase* and *word,* rather than *program, function, expression*, and so on. In this stage, annotation fragments are delimited according to a user-defined grammar. At the same time, complex word-equivalent objects, such as phone numbers, e-mail and web addresses, and so on, are recognized using predefined structural patterns of object grammars. All grammars are described in BNF(Backus Naur Form)-like form using the TXL language notation.
2. *Markup*. The next stage uses an annotation schema to infer annotations of document fragments. This schema contains a list of concept names drawn from the domain-dependent model and a vocabulary consisting of indicators related to each concept. Cerno assumes that this domain input is constructed beforehand either automatically using some learning methods or manually in collaboration with domain experts. Indicators can be literal words, phrases or names of parsed entities. They also can be *positive*, i.e., pointing to the presence of the given concept, or *negative*, i.e., pointing to the absence of this concept. If a positive indicator is present in a text fragment, Cerno annotates the fragment with a corresponding tag, unless a negative indicator is present.
3. *Mapping*. In the last stage, annotated fragments are selected from all annotations according to a database schema template schema, are copied to an external database. The database is manually derived from the domain-dependent semantic model and represents a set of fields of a target database.

In this application we adapt Cerno to perform semantic annotation using structural, syntactic and semantic information.

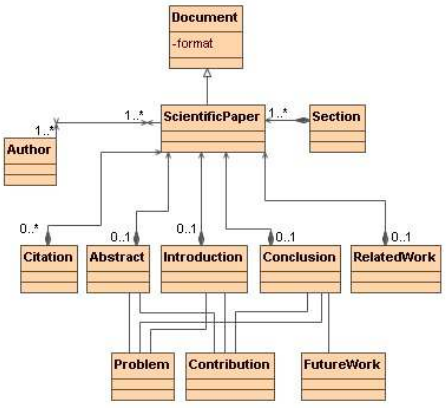
In the following section, we demonstrate our application of Cerno to the new domain of research publications analysis.

3 The Biblio Tool for Analyzing Research Papers

The design of Biblio was based on a set of initial requirements:

- *Plain text extraction.* As the input for the Cerno system must be provided in plain text format, a preprocessing stage is required in order to convert files in various formats to plain text. The main formats that must be accepted are PDF and MS Word, as they are wide-spread in scientific publishing.
- *Analysis of document structure*. The tool should be able to identify the main structural elements of the document and, in particular, to identify for the user such sections as title and author information, abstract, introduction, conclusions, and references.
- *Semantic annotation.* The tool should provide semantic recognition and extraction of important aspects of the document such as the problem addressed and the claimed contributions. Most often, phrases that briefly and explicitly describe the main achievements of a work are located in the abstract, introduction and conclusion sections. Based on this premise, we will look for this information only in these sections and ignore the remaining content of the file.
- *Other functionalities*. Apart from the functions listed above, the tool should provide the user a possibility to view all the fields, to fill or edit their contents and to execute various queries over the uploaded bibliographical data. Finally, by request, the tool must produce a citation line for a specified document in the necessary format and visualize in the interface the complete bibliography.

In order to use Cerno, we first had to elicit domain-dependent knowledge. For this purpose, we designed a conceptual model representing scientific document content, as shown on the left in Fig. 1. On the basis of this model, the annotation schema and database schema template was derived, as seen on the right in Fig. 1. The underlying database schema includes the tables *Document, Document_author, Document_citation, Document_contribution, Document_problem, Document_Future_Work, Author, Citation* and others, as shown in Fig. 2.




```
<Document>
   <Title></Title>
   <Authors></Authors>
   <Email></Email>
   <Abstract>
       <Problem></Problem>
       <Contribution></Contribution>
   </Abstract>
   <Introduction>
       <Problem></Problem>
       <Contribution></Contribution>
   </Introduction>
   <Conclusion>
       <Problem></Problem>
       <Contribution></Contribution>
       <FutureWork></FutureWork>
   </Conclusion>
   <Citation>
       <CitationTitle></CitationTitle>
       <CitationYear></CitationYear>
   </Citation>
</Document>
```

Figure 1. Conceptual model of the domain and annotation schema

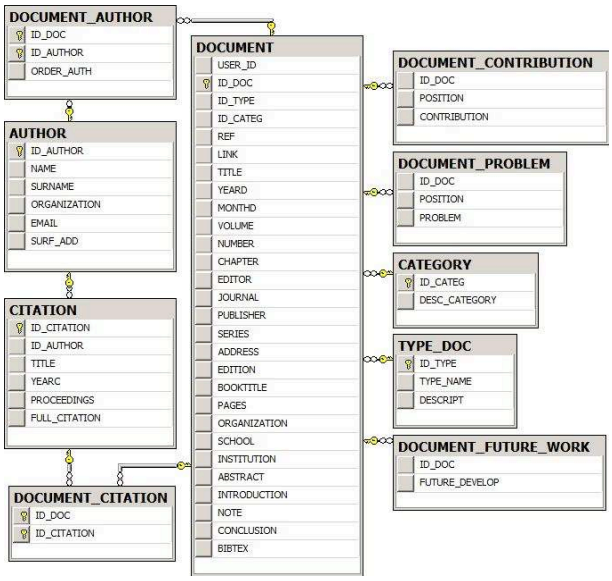


Figure 2. Database schema

If input documents are PDF files we use a PDF text extractor tool to convert them documents into plain text files. For MS Word documents, we first convert input files into RTF format and then extract plain text from the converted files.

To address the problem of varied page layouts, we adapted Cerno, with a preprocessing module which flattens paragraphs and sentences in such way as to obtain one sentence per line.

Recognition of document structure is another challenge that we accommodated. In order to identify structural elements, we used a set of patterns combining position information and keywords. For instance, a pattern for identifying authors information takes into account both position within the document and syntactic representation of a text line, i.e., a line containing authors information normally is the sequence of names separated by commas and followed by email addresses. Note, that the pattern for each structural element is a generalization of several commonly used formats.

As for semantic elements, in Cerno they are extracted on the basis of the category wordlists yielded manually using entities of the conceptual model and context driven indicators. For example, the contribution information is certainly present in sentences having one of the following structures: “this work proposed…”, “the present paper analyzes…” and other similar phrases.

Finally, the issue of generating citation lines is also realized by reusing the information recognized in the documents of the user’s collection. This information is then used by the query engine when retrieving data from the database.

Following Cerno’s method, document processing is then organized into three phases:

1. *Parse*: Lightweight parse of the document structure;
2. *Markup*:
    a. Structural markup of the main sections;
    b. Semantic markup of relevant facts in specific sections;
3. *Mapping*: Annotations are copied into the database.

The result of the first stage is a parse tree of the input text. A fragment of the grammar imposed on input documents is provided in Fig. 3. Essentially, the input is considered as the sequence of lines, among which we can find the headers of abstracts, introductions, and other sections. The tool recognizes such special lines, annotates them as the headers of appropriate sections and identifies the structural partitioning of the document based on these annotations.

```
% input is defined as a sequence of zero or more lines
define program
  [repeat line]
end define
% we want to distinguish between special lines, which are headers,
   and any other lines
define line
  [special_line]
 |[not special_line] [repeat token_not_newline] [opt newline]
end define
% different types of the headers are specified
define special_line
 [abstract_line] |[introduction_line] |[conclusion_line]
 |[references_line]
end define
```

Figure 3. A grammar fragment

When the structural markup is complete, Cerno's engine semantically annotates phrases in the chosen sections, abstract, introduction and conclusions. Fig. 4 shows some indicators used for this application in order to identify future work and contribution information. For the reasons of TXL compiler performance, we could not use a lemmatizer to handle different morphologic word forms, and instead we listed all word forms manually.

```
FutureWork : future work, future works, future development, future
research, future investigation, future investigations, in future, in the
future;
Contribution : this paper presents, this paper introduces, this paper
proposes, this paper shows, this paper demonstrates, this paper studies,
this paper explores, this paper investigates, this paper provides, this
paper describes, this paper discusses, <...>
```

Figure 4. Indicative phrases for semantic annotation

The result of this stage is the text file normalized in its format and annotated with structural and semantic markups (Fig. 5) which is then passed on to the Biblio application for further analysis.

```
<Doc><Head><Title>ToMAS: A System for Adapting Mappings while Schemas
Evolve</Title><Authors>Yannis Velegrakis Renrее J. Miller Lucian Popa
John Mylopoulos </Authors></Head>
<Citation> [1] P. Bernstein, A. Levy, and R. Pottinger. A Vision for
Management of Complex Models. SIGMOD Record, 29(4):55-63, December 2000.
</Citation>
       <...>
</Doc>
```

Figure 5. An annotated text output

The user interacts with the Biblio GUI that allows to fully control the process of annotation, data acquisition and data manipulation. The prototype has been implemented in Borland Delphi.

The graphic user interface (GUI) of Biblio is composed of the following sections:

- *Document*: Shows the information identified by Cerno for the current document with extra fields related to identification of the document that the user can fill through the interface, e.g., page numbers, editors, conference venue, etc.
- *Citations*: Shows the citations from the current document, extracted by Cerno;
- *Author*: Allows management of the author information for the current document.
- *Advanced search*: An interface to retrieve records from the database by database query.
- *Import*: Allows selection of new papers, processing of them by the Cerno engine and upload of the extracted information to the user's database.
- *Export*: Allows export of the bibliographical data of the selected papers in different formats, such as MS Word and BibTex.

The *Document*, *Citations*, and *Author* sections are devoted to manage data details for each document loaded into the database. For each new document processed, a record with a unique identifier is created, in case if the database does not contain a document with the same title and authors.

The Import section is the core of the application. It is composed by four components:

- *Pre-process component*: this module is responsible for the extraction of text from document. It uses different sub-program accordingly with the input document, e.g., for PDF document the PDFtoText tool is used for text extraction.
- *A normalize component*: this component is based on TXL parser and it allows to normalize the input text by uniforming the structure of sentences, paragraph.
- *The TXL component*: it annotates the document as described in section 3.2.
- *The database front-end:* it is responsible for import of the annotated text into the database, and verification of the consistency of data.

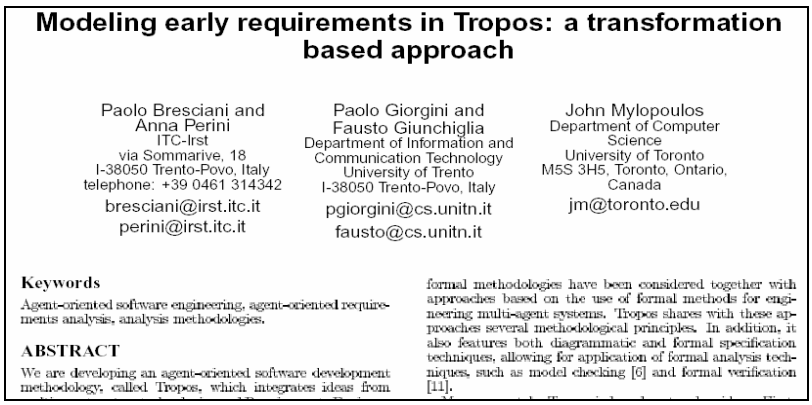
When structural and semantic annotation is completed, the relevant information is copied into a relational database. Business rules on database side, enforced on it control the integrity of the data.

Finally, the *Export* section allows to control the export of data. The tool allows to manage different citation line generation.

4 A Running Example

Now we illustrate by example how Biblio processes documents.

When a user needs to write a paper, he collects a bunch of papers, and loads them into the system, see for an input file example in Fig. 6.



Modeling early requirements in Tropos: a transformation based approach

Paolo Bresciani and Anna Perini
ITC-Irst
via Sommarive, 18
I-38050 Trento-Povo, Italy
telephone: +39 0461 314342
bresciani@irst.itc.it
perini@irst.itc.it

Paolo Giorgini and Fausto Giunchiglia
Department of Information and Communication Technology
University of Trento
I-38050 Trento-Povo, Italy
pgiorgini@cs.unitn.it
fausto@cs.unitn.it

John Mylopoulos
Department of Computer Science
University of Toronto
M5S 3H5, Toronto, Ontario, Canada
jm@toronto.edu

Keywords

Agent-oriented software engineering, agent-oriented requirements analysis, analysis methodologies.

ABSTRACT

We are developing an agent-oriented software development methodology, called Tropos, which integrates ideas from

formal methodologies have been considered together with approaches based on the use of formal methods for engineering multi-agent systems. Tropos shares with these approaches several methodological principles. In addition, it also features both diagrammatic and formal specification techniques, allowing for application of formal analysis techniques, such as model checking [6] and formal verification [11].

Figure 6. An input document

In case of the PDF format, the input document is passed to the PDFtoText tool (http://www.foolabs.com/xpdf/) to convert it to plain text format. In case of MS Word format, MS Office provides internal facilities to convert files into plain text. After that, the text document is passed to the Cerno's engine. The output of the processing is then handled by the interface to feed the database of bibliographical data. The records of this database are used for filling the fields of the user interface, as shown in Fig. 7.

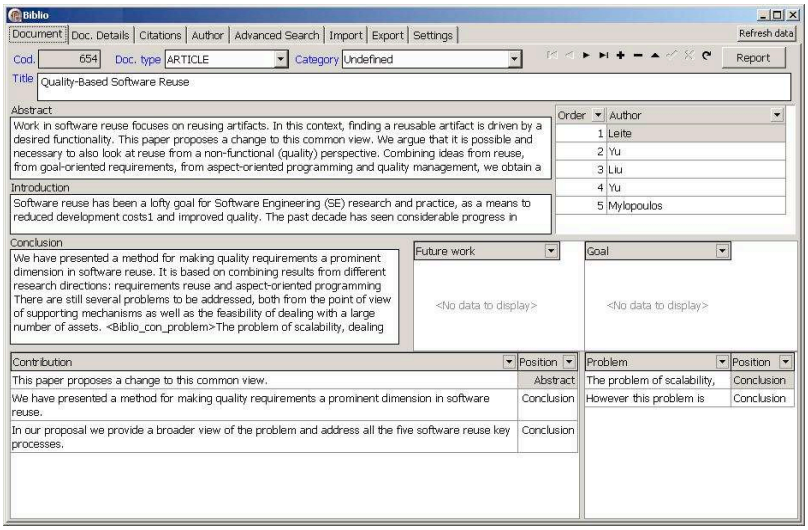


Figure 7. GUI of Biblio

In addition, for generating complete citations, the tool allows the user to manually add missing bibliographic information that cannot be found in the document body, such as the year of publication, journal, etc. The result can be exported to a file in MS Word or BibTEX format. For example, Fig. 8 shows a fragment of the generated bibliography for our example paper.

1. "Using Semantic Networks for Database Management", *In Proceedings of the First International Conference on Very Large Databases,* Framingham, MA, pp.144-172, 1975
2. Mylopoulos J., Leite J., Yu Y., Liu L., Yu E.,"Quality-Based Software Reuse",,
3. "Information Systems as Social Structures",,
4. Mylopoulos J., RodríguezGianolli P.,"A Semantic Approach to XML-Based Data Integration",,
5. Mylopoulos J., Bresciani P., Perini A., Giunchiglia P.,"A Knowledge Level Software Engineering Methodology for Agent Oriented Programming",,

Figure 8. The generated citation list

5 Evaluation

In order to carry out a preliminary evaluation of our tool, we collected a set of 60 Computer Science publications. The articles were published in different years, on different events, in various formats and layouts.

In order to evaluate the quality of structural markup, we calculated the recall and precision rates of the markup produced by Cerno for the *Title* and *Author* fields against manual human opinions. This information is necessary for obtaining a citation line, and at the same time, can be found in the body of any published paper.

Recall is a measure of how well the tool performs in finding relevant items, while precision indicates how well the tool performs in not returning irrelevant items. In this evaluation, we also took into account partial answers, giving them a half score, as shown in formulas (1) and (2). The annotations are *partially correct* if the entity type is correct and the contents are overlapping but not identical.

$$Recall = (TP + \tfrac{1}{2}\,Partially\ Correct) / (TP + \tfrac{1}{2}\,Partially\ Correct + FN) \quad (1)$$

where TP – true positive answers, FN – false negative answers

$$Precision = (TP + \tfrac{1}{2}\,Partially\ Correct) / (TP + \tfrac{1}{2}\,Partially\ Correct + FP) \quad (2)$$

where TP – true positive answers, FP – false positive answers

The results of the evaluation, according to these quality measures, are shown in Table 1. All values are provided in percents.

Table 1. Evaluation results for most crucial structural annotations.

	Title	Author
Recall	0.93	0.90
Precision	0.90	0.94

The estimated data demonstrate high quality rates in identification of structural sections of the electronic publications. However, some errors were observed. These errors were caused mainly by the erroneous output of the PDF converter for some documents.

To evaluate semantic annotations, we considered three concepts: *Contribution, Problem,* and *Future Work*. Manual creation of the reference annotation requires expensive domain expertise. In Table 2 we provide the evaluation for such elements.

Table 2. Evaluation results for semantic annotation.

	Contribution	Problem	Future Work
Precision	0.91	0.92	1.00
Recall	0.71	0.64	0.45

Observing the obtained estimates, we see that identification of Contribution, Problem and Future Work information was very accurate, 91, 92 and 100% respectively. Still, some false positive answers were detected. The Recall values are indicative of the need of improve patterns to identify the concepts. To avoid them, the Biblio domain dependent components can be improved in future.

Although, this evaluation is preliminary and can be extended to the larger articles collections, the overall results suggest for appropriateness of using lightweight text processing techniques for the analysis of research publications in electronic format. With the small effort required to adapt the Cerno framework to the different domain,

we achieved high performance rates for both structural markup and recognition of text fragments related to the semantic concepts.

6 Related work

The need to extract bibliographical data was underlined by Cruz *et al*. in [9], where the authors proposed a methodology to create citation indices. A number of commercial and academic products for organization and storage of bibliographic data are already available, the detailed survey of existing tools can be found in [10]. Existing systems support the process of automatic extraction of citations from existing documents and the creation of personal or shared databases.

Noodle [11], WestCheck [12] and LawPro [13] are platforms providing support for extraction of citations from exiting document. This information is stored in a local or an on-line database, and then, citation lines in different styles can be generated.

Other academic projects, such as Citeseer [14,15], CiteBase [16], and AnaPort [17] are search engines that extract structural information and citations from on-line scientific documents. They support such document formats as PDF, MS Word, and PostScript. A citation line in different format is provided as output.

Another group of tools support automatic formatting of bibliographical data on the basis of user-defined annotations, such as on-line system for generation of citations CitationMachine [18], Polaris on-line database which is developed and used at present at the University of Trento [19].

Our work relates well to the ShaRef project [20,21] that suggest the ShaRef tool allowing for both Web-based and offline creation, management, and sharing of reference information. This information can be exported to MS Word or Web browsers applications. The ShaRef approach is based on an XML data model. Nevertheless, no automation of the data extraction process is provided.

The proposed tool, Biblio, differs from earlier efforts particularly in extraction of semantic knowledge from documents. Biblio uses a combination of structural and lightweight semantic annotation to provide the user with information about the focus, main achievements and future work directions of the scientific documents. The tool is easily expandable to identify different information by changing the conceptual model and domain dependent inputs.

7 Conclusions

This paper addresses the problem of providing automatic support for authors who need to deal with large volumes of research literature for different purposes, such as creating a bibliography, constructing a personal bibliography, or managing a personal collection of publications.

To help the user in these tasks, we proposed Biblio, a novel user-friendly application for analysis of research publications in electronic format. The process is based on the lightweight semantic annotation Cerno. The tool allows the user to

identify, visualize and store important system elements of the selected publications, and to automatically generate a bibliography from a set of chosen articles.

The tool is able to address the problems described in the introduction using a lightweight semantic annotation process. Different document formats are handled through a preprocessing phase. The document structure is captured using a combination of techniques based on mutual disposition of elements and syntax used to express such elements. Eventually Biblio can extract semantic information from documents by using the lightweight pattern-based Cerno's analysis. The backend of Biblio is a standard relational database that can be used for querying and evaluating the system. A preliminary evaluation of structural and semantic annotations demonstrates the potential for high quality results using our method.

References

1. Kiyavitskaya, N., Zeni, N., Mich, L., Cordy, J.R., Mylopoulos, J.: Text Mining with Semi Automatic Semantic Annotation, Proc. of PAKM'06, LNCS, Vol. 4333, 143–154.
2. Kiyavitskaya, N., Zeni, N., Cordy, J.R., Mich, L., Mylopoulos, J.: Applying Software Analysis Technology to Lightweight Semantic Markup of Document Text. In Proc. of Int. Conf. on Advances in Pattern Recognition (ICAPR 2005), Bath, UK, 2005, 590–600
3. Google Scholar, http://scholar.google.com
4. DBLP, http://dblp.uni-trier.de/
5. Citeseer, http://citeseer.ist.psu.edu/
6. ACM Portal, http://portal.acm.org/dl.cfm
7. IEEE Digital Library, http://ieeexplore.ieee.org/Xplore/conhome.jsp
8. Cordy, J.R.: The TXL Source Transformation Language. Science of Computer Programming, 61(3), 2007, 190–210
9. Cruz B. J. M., Lluch O. J., Krichel T., Pons B. P., Velasco S. E., Velasco S. L. "INCISO: Automatic Elaboration of a Citation Index in Social Science Spanish Journals" AMETIST, no. 0, September 2006
10. Roth, D.L. "The emergence of competitors to the Science Citation Index and the Web of Science", Current Science, Vol. 89 (2005), 1531–1536.
11. Noodle, http://www.noodletools.com/tools.html
12. WestCheck, http://www.westlaw.com
13. LawPro, http://www.llrx.com/
14. Bollacker, K.D.; Lawrence, S.; Giles, Discovering relevant scientific literature on the Web, C.L. Intelligent Systems and Their Applications, IEEE, Vol. 15 (2000) 42–47
15. Lawrence, Steve, Kurt Bollacker, and C. Lee. Giles "Indexing and retrieval of scientific literature". In Proc. of International Conference on Information and Knowledge Management, CIKM99, (1999) 139–146.
16. CiteBase, http://www.citebase.org/
17. AnaPort, http://www.ana-project.org/ref/
18. CitationMachine, http://citationmachine.net/index.php?source=11#here
19. Polaris on-line database, http://polaris.unitn.it
20. Wilde, E., Anand, S., Bücheler, T., Jörg, M., Nabholz, N., Zimmermann, P., "Collaboration Support for Bibliographic Data", International Journal of Web Based Communities, 3(1), 2007.
21. ShaRef project, http://dret.net/projects/sharef/