BibMix: Enrichment of Citation Metadata Based on Integration of Bibliographic Data

Masters's thesis, September 2, 2010

Bas Wenneker

BibMix: Enrichment of Citation Metadata Based on Integration of Bibliographic Data

THESIS

submitted in partial fulfilment of
the
requirements for the degree of

MASTER OF SCIENCE
in
COMPUTER SCIENCE
TRACK INFORMATION ARCHITECTURE

by

Bas Wenneker
born in Delft, The Netherlands

Web Information Systems Group
Department of Software Technology
Faculty EEMCS, Delft University of Technology
Delft, The Netherlands
http://eemcs.tudelft.nl

BibMix: Enrichment of Citation Metadata Based on Integration of Bibliographic Data

Author:	Bas Wenneker
Student id:	1213865
Email:	b.wenneker@student.tudelft.nl

Abstract

A citation in an academic text provides the reader with exact publication information to uniquely identify and locate a scholarly resource for further study. In the domain of academic research, citations are a key part of linking discrete pieces of knowledge. As opposed to the natural language mostly used in academic texts, bibliographic applications are typically based on structured publication information and therefore usually rely on the quality and completeness of automatically extracted citation metadata. However, the results from automatic Citation Metadata Extraction approaches are often unreliable. Moreover, the extracted citation metadata is only a small part of the relevant metadata that is available spread across the internet. In this thesis, we propose an approach and a supporting software component capable of enriching citation metadata with relevant bibliographic data from sources on the web.

For the evaluation of the approach, we have designed a quality measure to express the quality of enriched references as a number. Based on this measure we have assessed the quality of the enriched references of BibMix to be 0.72.

Contents

List of Figures

List of Tables

Part I

Background and Context

1 Introduction

This chapter introduces the background and situation leading to the problem addressed by this thesis. The problem is demarcated and a main research goal and research questions are formulated. Finally the approach and report structure are given.

1.1 Background

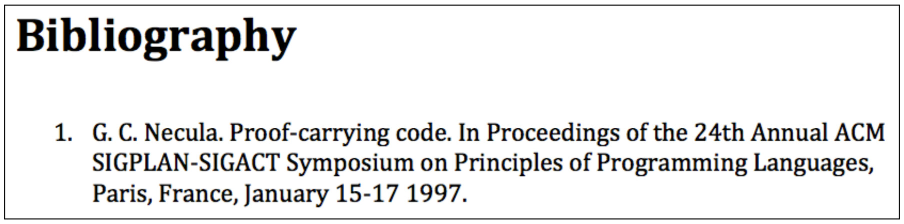
A citation in an academic text provides the reader with exact publication information to uniquely identify and locate a scholarly resource for further study. In the domain of academic research, citations are a key part of linking discrete pieces of knowledge. Citations are commonly found in the bibliography section of academic publications, on personal websites of researchers or on portals of on-line digital libraries like CiteSeer[1] and Google Scholar[2]. Figure 1 shows a fragment of a Bibliography section of a publication, showing a citation.

Citations are free-form text strings providing publication information in natural language. The information that can be extracted from citations is called *citation metadata*. A *reference* is an abstract structure referring to a publication, it acts as a container for bibliographic data, e.g. extracted citation metadata. Both citations and references refer to publications (see Figure 2), though their representation is different.

Citation metadata can be extracted both manually and automatically. However, the ever-growing number of publications and the high costs for manual extraction lead to increased interest in automatic approaches. Automatic Citation Metadata Extraction (CME) approaches often suffer from unreliable extraction results. In addition, the metadata extracted from citations usually only contains a small portion of the structured bibliographic data that exists on the web for the publication. Just like the extraction process, the enrichment of references with additional relevant bibliographic data is time-consuming when performed by human experts, and thus it is expensive.

The references resulting from state-of-the-art CME applications suffer from

[1] CiteSeer, http://citeseerx.ist.psu.edu/, last visit on August 2, 2010
[2] Google Scholar, http://scholar.google.com/, last visit on July 9, 2010


Bibliography

1. G. C. Necula. Proof-carrying code. In Proceedings of the 24th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Paris, France, January 15-17 1997.

Figure 1: An fragment of a Bibliography section, showing a citation.

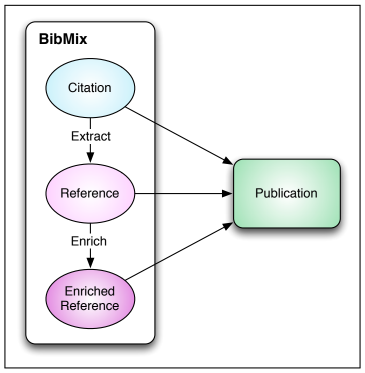


Figure 2: An overview of the BibMix citation metadata enrichment approach.

the unreliability of the extraction and the incompleteness of the extracted metadata. In this thesis, I propose an approach and a software component called BibMix, capable of enriching citation metadata based on integration of relevant bibliographic data from sources on the web. Figure 2 shows an abstract view of what the enrichment approach implemented by BibMix looks like. The metadata of a citation is extracted by a state-of-the-art CME application and represented by a reference. This reference refers to the same publication as the citation, although it might contain less information because of the unreliability of the extraction procedure. BibMix takes the reference and integrates it with bibliographic data from sources on the web, resulting in an enriched reference.

Here is an example to get a better understanding of what BibMix does. Take a look at the citation from Figure 1, it refers to a publication of G. C. Necula. Its metadata is extracted using a state-of-the-art CME application, resulting in a reference. The reference is a structured representation of the data that the CME application was able to extract. As shown in Table 1, the CME application was not able to extract all information, the address and month attributes are not extracted and thus not present in the extracted reference. Moreover, the publication date is mistaken for page numbers. When at least the title is extracted, BibMix is able to enrich the reference. The extracted metadata is used to collect bibliographic data from Bibsonomy[3] and The Collection of Computer Science Bibliographies[4]. BibMix integrates the relevant collected bibliographic data with the reference. As shown in the table, the enriched reference is richer than the reference it originates from. The enrichment approach integrated data the CME application was not able to extract (month and address) and data that was not present in the citation (publisher, page numbers and organization). In addition, BibMix also corrected the page numbers attribute and found the first name of the author.

Note that BibMix models references as BibTeX records (the the first column

[3]Bibsonomy, http://www.bibsonomy.org, last visited on August 12, 2010

[4]The Collection of Computer Science Bibliographies, http://liinwww.ira.uka.de/bibliography/index.html, last visited on August 12, 2010

BibTeX Record Fields	**Metadata present in citation**	**Metadata extracted from citation**	**Data after enrichment**
Address			Paris, France
Annotate			
Author	G. C. Necula	G. C. Necula	George C. Necula
Booktitle	In Proceedings of the 24th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages	In Proceedings of the 24th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages	In Proceedings of the 24th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages
Chapter			
Crossref			
Edition			
Editor			
Howpublished			
Institution			
Journal			
Key			
Month	January		January
Note			
Number			
Organization			ACM SIGACT and SIGPLAN
Pages		15-17	106-119
Publisher			ACM Press
School			
Series			
Title	Proof-carrying code	Proof-carrying code	Proof-carrying code
Type			
Volume			
Year	1997	1997	1997

Table 1: This table shows the metadata present in the citation (see Figure 1), the metadata that is extracted by a CME application and the metadata present the reference after enrichment.

of the table shows the fields of a BibTeX Record as described in the BibTeX specification (Patashnik, 1988)). BibTeX is a popular reference management tool and file format which is used to model references. Many popular digital libraries, reference management tools and scholarly search engines support importing and exporting bibliographic data in the BibTeX format[5]. The choice for BibTeX enables BibMix to consume bibliographic data from sources on the web, while the output of BibMix can be consumed by other applications.

[5]Tables with popular bibliographic file import en export formats: http://en.wikipedia.org/wiki/Comparison_of_reference_management_software, last visit on July 1, 2010

1.2 Terminology

In the related literature the words 'citation' and 'reference' are often used interchangeably. In this text we make a clear distinction between the two. A *citation* refers to a free-form text string providing publication information which a human reader can use to uniquely identify and locate a piece of academic knowledge. A *reference* is a more abstract structure which acts as a container for structured citation metadata. As illustrated by Figure 2 both references and citations refer a publication, the difference between them is their representation. Citations are free-form natural language texts, while references are abstract structures.

The reference model used in this text is defined in the BibTeX specification (Patashnik, 1988) (for an example, see Table 1). This model consists of a set of labelled fields like 'author, 'title' and 'publisher', which may occur only once in each reference. For more information about the BibTeX reference model see Section 2.2.

1.3 Motivation

The exploration of literature plays a crucial role in scientific research. Many scientists devote much of their valuable time exploring and digesting scientific literature. With the ever-growing volume of scientific resources, the retrieval, analysis and management of those resources are becoming a real challenge for any researcher. Therefore, it is not surprising to see how online digital libraries like CiteSeer[6], Google Scholar[7] and Scirus[8] become increasingly valuable nowadays. They raise researcher's efficiency due to a faster access to current resources and the possibility for quicker distribution of research results (Kramer et al., 2007).

Also very important in scientific research is *citation analysis*. Citation analysis is used to evaluate the publications in a field and the research conducted in a discipline, by an expert or even in an entire country (Baird and Oppenheim, 1994). Citation analysis enables us to measure the impact of academic resources, the activity in research domains which researchers or organisations are influential etc. This can help organisations in hiring processes. Many grants, tenures and employments in the academic world require candidates with outstanding research records including a number of 'high impact' papers (Canós et al., 2008). Citation analysis can help finding the right people for a position.

The commonality between digital libraries, citation analysis and several other bibliographic applications is that they all benefit from citation metadata that is complete and of high quality. Automatic CME approaches are not trivial because citation metadata is often hard to recognize for reasons such as: errors in data entry, a large variety of citation formats, the lack of (the enforcement of) a standard, defective citation gathering software, ambiguity in author names and the use of abbreviations for publication venues (Lee et al., 2007).

[6]CiteSeer, http://citeseerx.ist.psu.edu/, last visit on August 2, 2010
[7]Google Scholar, http://scholar.google.com/, last visit on July 9, 2010
[8]Scirus, http://www.scirus.com/, last visit on August 2, 2010

The current state in CME approaches and software components is discussed in Section 2.1. Despite the unreliability of the results, the increasing number of publications and the cost of manual approaches lead to increasing interest in automated CME.

1.4 Focus on the Domain of Computer Science Literature

As mentioned in the previous section, there are two main problems with automatic CME: the unreliability of the extraction results and the incompleteness of the extracted citation metadata. This master thesis limits itself to researching how the metadata of a citation can be used to collect and integrate additional relevant bibliographic data.

The BibMix software component proposed in this text will not contain a new CME component but uses an existing one. The component works independently from the used CME component.

The evaluation of the software component is done in the Computer Science domain, for that domain there are high-quality data sources available on the web from which BibMix can obtain bibliographic data.

1.5 Research Goal

The combination of the problems identified in this chapter, the demand for reliable and complete citation metadata, lead to the formulation of the main research goal of my thesis:

"Create an approach and a supporting software component to enrich BibTeX modelled references by integrating citation metadata with bibliographic data from sources on the web."

In order to achieve the main research goal, a number of research questions have been constructed:

1. What is the state-of-the-art of Citation Metadata Extraction approaches?
2. Which steps can be taken in order to integrate citation metadata with bibliographic data from sources on the web?
3. How does the approach from 2 perform?

These research questions will be answered throughout the thesis report and the results will be summarised in the conclusions.

1.6 Research Approach

The main research goal and the research questions will be addressed by researching the state-of-the-art of citation metadata extraction components, the design and implementation of a new software component for enriching extracted citation metadata and finally an evaluation of the components' performance. The approach consists of a partly overlapping but generally sequential approach consisting of five steps. Details of the activities performed are discussed in the sections indicated.

Step 1: Context Analysis - Chapter 2
Determine the state-of-the-art of CME components and select a component to be part of the citation metadata enrichment approach.

Step 2: Design of the Enrichment Approach - Chapter 3
Design a citation metadata enrichment approach taking care of both extraction and enrichment of citation metadata.

Step 3: Implementation of the Enrichment Approach - Chapter 4, 5
Implement a software component supporting the approach designed in step 2.

Step 4: Determine Evaluation Approach - Chapter 6, 7
Determine an measure for quality and develop an evaluation approach to evaluate the performance of the software component.

Step 5: Evaluation - Chapter 8
Perform an evaluation of the newly created enrichment approach and analyse the evaluation results.

1.7 Report Structure

This report is structured as follows. Part I comprises the background and context that provide an overall picture of the topic of discussion. Chapter 1 introduces the problem addressed by the thesis, demarcates it and formulates a research goal and approach. Chapter 2 introduces a number of important concepts related to this thesis.

Part II comprises the description and explanation of the enrichment approach and implementation. Chapter 3 introduces the approach for citation metadata enrichment. Chapter 4 introduces the conceptual model of the implementation of the approach, which is called BibMix. Chapter 5 provides the implementation details of BibMix and tools it makes use of.

Part III consists of the activities undertaken to perform the evaluation necessary to answer the third research question. Chapter 6 provides an overview of events that affect the quality of the enriched references and introduces a measure for the quality of enriched references. In Chapter 7 the evaluation approach is detailed, while Chapter 8 provides the evaluation results and an analysis of the results.

Part IV comprises the conclusions that can be drawn from the thesis project. Chapter 9 provides the contributions of this thesis project. Chapter 9.1 provides the answers to the research questions and main research goal. Finally, in Chapter 10 future work is presented.

The appendices complete the report and are given in the order in which they are introduced in the main text.

2 Related Work

2.1 Citation Meta-Data Extraction

CME is the problem of turning citations into corresponding structured references. A number of CME techniques have been reported in recent years. The component I propose in this paper, BibMix, requires a CME component to extract metadata from citations. This metadata acts as the input to the component. The quality of the output of BibMix heavily depends on the quality of the extracted metadata.

In this section I describe the state-of-the-art of CME techniques, which can be generally divided into two categories: *template-based* and *machine learning-based* approaches.

2.1.1 Template-based Extraction

A *template-based approach* takes an input citation and matches its syntactic pattern against known templates. The templates are models of what citations may look like and template-based approaches use template databases with citation template definitions matching various citation styles.

Huang (Huang et al., 2004) describes an approach that is originally used in the domain of biology. In their approach, they use a form translation program to translate citations into a protein sequence (see Figure 3), which can be processed more easily. BLAST is a protein sequence matching program and it is able to process the protein sequence of the citation. BLAST uses a scoring table to search for most similar sequences in a protein sequence database. This database stores the templates of known citations that have been translated into protein form. After finding the most similar sequence template, a pattern extraction program is used to parse the citation metadata according to the template. Afterwards, the parsed metadata can be used to improve the performance of BLAST. First the results should be validated manually, the validated information can be added to a knowledge database used by the protein sequence matching process.

Chen et al. (Chen et al., 2008) enhanced the previously mentioned approach, resulting in a new tool called BibPro. The main enhancement of BibPro is

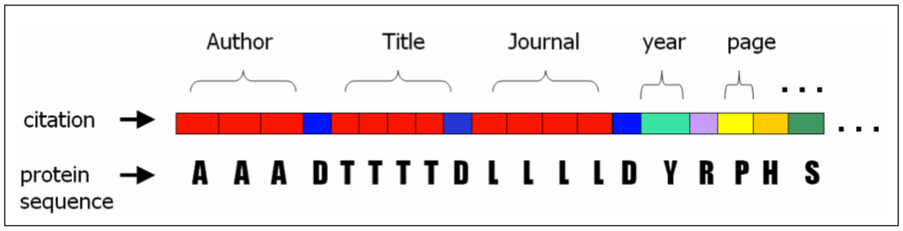


Figure 3: A transformation from a citation string to a protein sequence. (Chen et al., 2008)

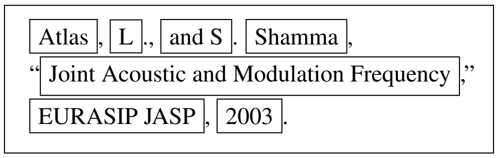

Atlas , L ., and S . Shamma ,
“ Joint Acoustic and Modulation Frequency ,”
EURASIP JASP , 2003 .

Figure 4: A tokenized citation. Each box contains one token. (Cortez et al., 2007)

that it does not need knowledge databases (e.g., an author name database) to generate a protein sequence for citations. Instead, only the order of punctuation marks in a citation is used to represent its format. Just like the approach by Huang (Huang et al., 2004), BibPro employs the BLAST to find the most similar citation formats in database. However, for templates with equal similarity scores BibPro uses the Needleman-Wunsch algorithm (Needleman and Wunsch, 1970) to choose the best-fit citation format as the extraction template.

An approach described in (Day et al., 2007) considered six major citation styles in computer science literature and constructed a hierarchical knowledge representation framework (INFOMAP). This tool is able to extract important concepts from natural language texts. As a general representation of domain knowledge, INFOMAP consists of domain concepts and their related sub-concepts, such as categories, attributes and actions. The relationships of a concept to its associated sub-concepts form a tree-like taxonomy in which INFOMAP classifies both concepts and related concepts. A powerful feature of the framework is its ability to represent and match complicated template structures. INFOMAP is able to extract author, title, journal, volume, number (issue), year, and page information from different reference styles. This approach is more powerful than typical template-based methods, since it provides an integrated hierarchical template editing environment and a more flexible template matching engine. The downside is that the templates of INFOMAP are particularly targeted at only a few commonly used citation styles, other citation styles and manually entered citations are not recognized very well.

Cortez et al. (Cortez et al., 2007) proposed an unsupervised CME method to automatically generate templates from a training data set. This approach, called FLUX-CiM, is based on a knowledge base of reference strings. Initially, labels are assigned to tokens based on the `(label, token)` pair's likelihood of appearance in the knowledge base. For tokens that do not occur in the knowledge base, a binding step is used to associate them with neighbouring tokens that have already been labelled. The tokens in FLUX-CiM are, just like in the BibPro approach, substrings of a citation delimited by punctuation rather than single words (see Figure 4). This follows from an observation by the authors that "in general, in a reference string, every field value is bounded by a delimiter, but not all delimiters bound a field."

Rule-Based Extraction A subcategory of the template-based approach is the *rule- or heuristic-based approach.* Here, rules and heuristics are used as template definitions. For example, CiteSeer (Bollacker et al., 1998) used simple heuristic rules to extract blbliographic metadata from citations, although the new version of CiteSeer, called CiteSeer^X, seems to have switched to the ParsCit machine learning-based approach (Councill and Kan, 2008) to extract citation metadata[9].

In (Besagni et al., 2003) Besagni et al. propose a rule-based approach based on Part-of-Speech (PoS) tagging. PoS tagging is the process of marking words in a text as corresponding to a part of speech, based on both syntactics and semantics. School-age children are commly taught a simplified form of this: the identification of words as nouns, verbs, adjectives, adverbs, etc. The rule-based approach acts in a bottom-up way, without an a priori model. This is because of the heterogeneity of citation structures. It gathers structural elements from basic tags to sub-fields and fields. Significant tags are first grouped in homogeneous classes according to their grammar categories and then reduced in canonical forms corresponding to record fields: "authors", "title", "conference name", "date", etc. Non-labelled tokens are integrated in one or another field by either applying PoS correction rules or using a structure model generated from well-detected records.

The rule-based approach proposed by Wei et al. (Wei et al., 2007) uses layer-upon-layer tagging. The method analyzes bibliographic attributes' appearances and punctuations to perform format and semantic taggings on two defined parsing layers. The layers differ in the granularity of the token definition. The *upperlayer reference parsing* is to parse a reference string into coarse-grained tokens while the *underlayer reference parsing* is to parse a reference string into fine-grained tokens. The layers have different rules attached that are applied to the tokens in that layer. For the semantic tagging of the metadata attributes "authors", "publications" and "locations" three specifically constructed lexicons are used.

2.1.2 Machine Learning-Based Extraction

Previous work in machine learning-based CME from research papers has been based on three major machine learning techniques. The first is Hidden Markov Models (HMMs) (Seymore et al., 1999; Connan and Omlin, 2000; Takasu, 2003; Yin et al., 2005; Hetzner, 2008). An HMM learns a generative model over input sequences and labelled sequence pairs, called the training set. HMMs are a finite state automaton with stochastic state transitions and symbol emissions (Rabiner, 1989). Each state is associated with a citation metadata field name such as "title", "author" or "date". A training set of labelled citations is first used to train an HMM. This model is then used to recover the most-likely state sequence that produces the sequence of observation symbols (i.e. the tokens/sub-strings of a citation). While enjoying wide historical success, standard HMM models

[9]CiteSeer^X Team: http://citeseerx.ist.psu.edu/about/team, last visit on April 8, 2010

have difficulty modelling multiple non-independent features of the observation sequence. Generally the results of HMM-based approaches are inferior to the results of Conditional Random Fields (CRFs)-based approaches (described later this section), however, Hetzner et al. (Hetzner, 2008) state that there may still be a place for HMM-based methods for citation parsing, due to their relative ease of use, availability of implementations, and possibility for improvement.

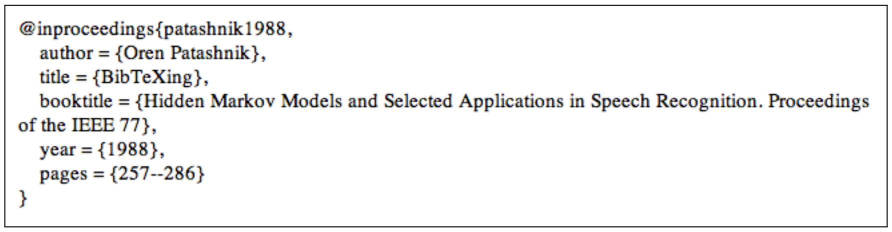
The second technique is Support Vector Machines (SVMs). This approach achieves better results in handling non-independent features through loosening the relationships between state transformation and observation sequence (Han et al., 2003). This sequence labelling problem is solved in two steps: first classifying each line independently to assign labels to its tokens, then adjusting these labels based on an additional classifier that examines larger windows of labelled tokens. Solving the information extraction problem in two steps looses the tight interaction between state transitions and observations.

Finally, Conditional Random Fields (CRFs) (Lafferty et al., 2001) enjoy the advantages of both HMM and SVM by allowing use of arbitrary and dependent features and joint inference over entire sequences. A CRF-based approach uses labelled citations from a training set to learn a model that can be applied to unseen data. This learning model scales well, handling large sets of possibly overlapping (i.e., conditionally dependent) features. A popular implementation of CRF in a CME component is ParsCit (Councill and Kan, 2008). ParsCit employs state-of-the-art CRF models to achieve its high accuracy in reference string segmentation, and heuristic rules to locate and delimit the reference strings and to locate citation contexts. Councill et al. have engineered features to rectify the classification errors made by models created from previous work. These features look at both the syntactic and semantic characteristics of tokens and their neighbours in order to label tokens correctly.

2.2 BibTeX: A Bibliographic Modelling Approach

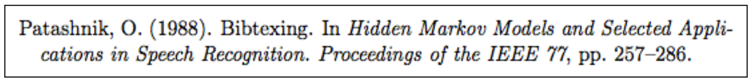
BibTeX is a reference management tool and file format which is used to model references. The reference management tool processes the references into formatted lists of citations (Patashnik, 1988). BibTeX is often used in conjunction with the LaTeX document preparation system for the TeX typesetting program (Lamport, 1986). The approach makes it easy to cite sources in a consistent manner, by separating bibliographic information from the presentation of this information. Today, the BibTeX file format is most popular amongst bibliographic modelling approaches (other popular formats are EndNote and RIS). Many popular digital libraries and scholarly search engines support importing and exporting citations in the BibTeX format[10]. Here we're only interested in the BibTeX reference model, as it is used as the internal and external reference model for the BibMix component.

[10]Tables with popular bibliographic file import en export formats: http://en.wikipedia.org/wiki/Comparison_of_reference_management_software, last visit on July 1, 2010



```
@inproceedings{patashnik1988,
   author = {Oren Patashnik},
   title = {BibTeXing},
   booktitle = {Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings
of the IEEE 77},
   year = {1988},
   pages = {257--286}
}
```

Figure 5: An example of a BibTeX entry.



Patashnik, O. (1988). Bibtexing. In *Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE 77*, pp. 257–286.

Figure 6: The citation corresponding to the BibTeX entry in Figure 5.

2.2.1 The BibTeX Reference Model

BibTeX uses a style-independent text-based file format for lists of bibliographic entries, such as articles, books, theses. The file containing the entries is like a database, it consists of records and fields. Each BibTeX record holds the bibliographic information for a single bibliographic entry. Records begin with an '@' symbol, followed by the record type (record types are described later in this section), and, in braces, a comma-separated list of entries of the form 'fieldname = {value}', where the 'fieldnames' are components of the bibliography entry such as 'author', 'title', etc. Figure 5 shows how a bibliographic record would be entered into a BibTeX file.

The record starts with the record type `@inproceedings`, which means the record is an article in a conference proceedings. After the record type comes `patashnik1988`, which represents the citation key (i.e., the keys used inside the \cite{...} in LaTeX commands to match up citations with bibliography items), not the labels that get printed in the bibliography. Finally there is a comma-separated list of field - value pairs. An example of how this reference might look as a citation in a bibliography after compilation is shown in Figure 6.

There are a total of 14 recognized record types. A list of these types, together with a description, is found in Appendix A. Furthermore, the BibTeX specification describes 26 different record fields. For a description of the record fields, see Appendix B.

Part II

Design and Architecture

3 The Enrichment Approach

This chapter introduces an approach for enrichment of citation metadata based on the integration of bibliographic data collected from sources on the web. The approach adopts the separation-of-concerns principle, separating the enrichment process in two phases: the Extraction phase and the Integration phase. These phases are further refined into several steps.

The structure of this chapter is as follows. Section 3.1 provides a global overview of the approach. In Section 3.2 we present an example of the enrichment of the metadata of a citation.

3.1 Global Overview of the Enrichment Approach

The goal of the approach is to integrate citation metadata with relevant bibliographic data from sources on the web. In order to do so, we introduce a pipeline, consisting of an Extraction phase and an Integration phase. Figure 7 provides an overview of the approach. The Extraction phase is responsible for extracting the metadata from a given citation and representing the metadata as a reference. During the Integration phase additional bibliographic data from several heterogeneous data sources is integrated with the reference. The end result of the pipeline is a BibTeX modelled reference which contains citation metadata integrated with bibliographic data from sources on the web.

3.2 The Enrichment Approach in Action

We illustrate the approach shown in Figure 7 by giving an example of a case where a citation is enriched using BibMix, the implementation of the enrichment approach. We use the citation from Figure 1, which refers to a publication of G. C. Necula in 1997.

The Extraction phase starts by passing the citation to a CME application. BibMix uses the state-of-the-art machine learning-based CME application ParsCit (for more information about ParsCit see Section 2.1.2). ParsCit extracts the metadata from the citation. As shown in Table 2 ParsCit was not

Attribute names	**Extracted metadata**
Author	G C Necula
Booktitle	In Proceedings of the 24th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages
Pages	15-17
Title	Proof-carrying code
Year	1997

Table 2: This table shows the metadata extracted from the citation shown in Figure 1.

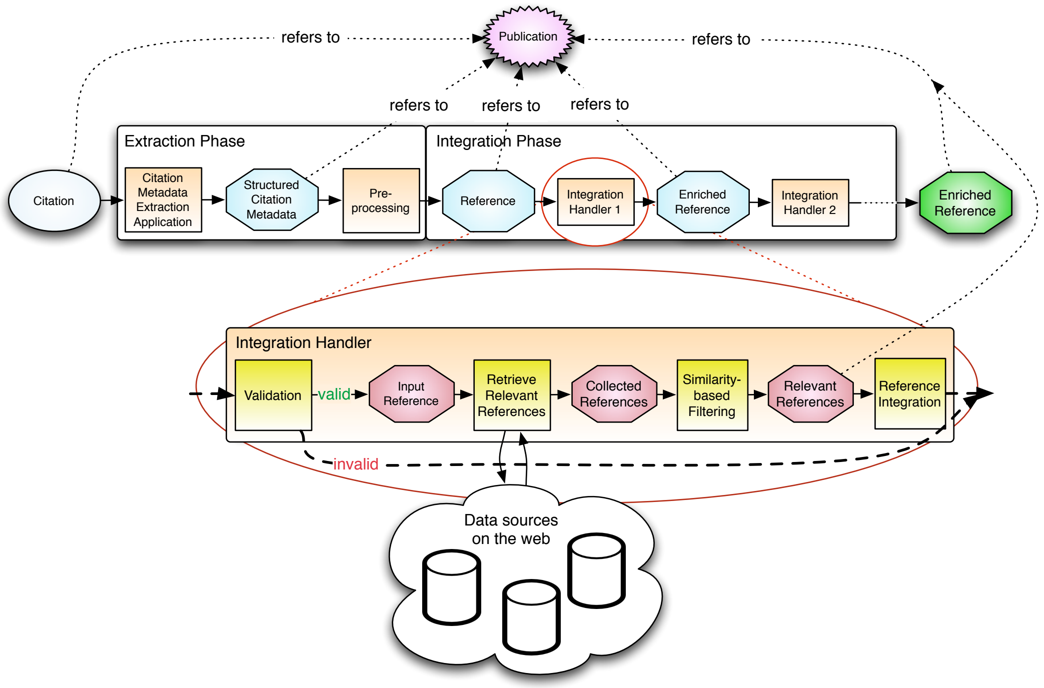


Figure 7: An overview of the enrichment approach.

BibTeX Record Fields	**Reference after Preprocessing**	**Reference after the first Integration Handler**	**Reference after the third Integration Handler**
Address			Paris, France
Annotate			
Author	G C Necula	George C. Necula	George C. Necula
Booktitle	In Proceedings of the 24th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages	In Proceedings of the 24th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages	In Proceedings of the 24th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages
Chapter			
Crossref			
Edition			
Editor			
Howpublished			
Institution			
Journal			
Key			
Month			January
Note			
Number			
Organization			ACM SIGACT and SIGPLAN
Pages	15-17	15-17	106-119
Publisher			ACM Press
School			
Series			
Title	Proof-carrying code	Proof-carrying code	Proof-carrying code
Type		An article in a conference proceedings.	An article in a conference proceedings.
Volume			
Year	1997	1997	1997

Table 3: This table shows the metadata present in the citation (see Figure 1), the metadata that is extracted by a CME application and the metadata present the reference after each integration handler of BibMix.

able to extract the values for the address and month attributes and thus they are not present in the extracted metadata. Moreover, ParsCit has mistaken the publication date for page numbers. The resulting 'structured citation metadata' is passed to the Preprocessing step.

In the Preprocessing step the structured citation metadata is transformed into a BibTeX reference model (see the result in the second column of Table 3). The transformation consists of syntactic operations converting the structure of the extracted metadata into the reference model that is used in the Integration phase. After the transformations the reference is passed on to the Integration phase.

The Integration phase may consist of several different steps handling the integration of bibliographic data. These steps, called Integration Handlers, can be further sub-divided into four sub-steps: reference validation, collection, filtering and integration steps. BibMix has three integration handlers, each integrates different bibliographic data:

1. The BibSonomy web-service is queried based on the title of the reference.
2. The BibSonomy web-service is queried based on the author names of the reference.
3. The Collection of Computer Science Bibliographies (CCSB) based on the title of the reference.

BibSonomy is a social bookmarking and publication sharing system on the web. It has a large collection of bibliographic data from various domains. BibSonomy provides a web-service, which allows third-party software to query the BibSonomy data collection. The validation step of the integration handler ensures the input reference contains at least a title, which is necessary in later steps. When the validation fails, the integration handler is skipped and the input reference is given to the next integration handler in the sequence. Otherwise, if the reference does contain the title, the reference is given along to the retrieval step.

Note that the prototype implementation of BibMix will have a predefined sequence of handlers that are executed. However, the goal of the approach is to decide which data source to target, depending on characteristics of the input reference. For example, if the 'booktitle' or 'journal' attributes contain the word 'ACM', a handler should be used to look for relevant bibliographic data in the ACM Digital Library. Furthermore, BibMix collects complete references from BibSonomy and the CCSB, but other implementations might also collect attribute values. For example, the implementation recognizes the conference from a 'booktitle' attribute. Then it looks up the address of the conference from the conference website and stores the collected value in the 'address' attribute.

During the retrieval step, a query is formulated based on the title attribute of the input reference. The query is sent to the web-service. BibSonomy responds with bibliographic data formatted in XML. The retrieval step parses the response and stores the contents in a set of references.

The BibSonomy web-service uses a fuzzy search mechanism, therefore it sometimes returns bibliographic data representing different publications but with titles that are kind of similar than the title in the query. To filter out the irrelevant parsed references, the collected references are given to the filtering step after the retrieval was finished. The filtering step compares each of the collected references to the input reference. The comparison is done by a Record Linkage tool called FRIL which looks for syntactic similarities between attributes of both references. The collected references that differ to much from the input references are removed. The remaining references are passed on to the Reference Integration step where the references are integrated with the input reference.

In the reference integration step the empty attributes of the input reference are filled with the attributes of the filtered references. An integration conflict occurs when both a filtered reference and the input reference have a value for the same attribute. For most attributes the value of the input reference is kept intact and the value from the filtered reference is ignored. For conflicts with the author and pages attributes the Reference Integration step tries to select the highest quality data. For example, during the integration of the bibliographic data from the BibSonomy web-service, an author name with the full first name was found (George C. Necula). The input reference only has the author name with an abbreviated first name (G C Necula), therefore the integration step replaces the original extracted author name (see third column of Table 3). Eventually the reference integration step results in an enriched reference: a reference which consists of the citation metadata integrated with the bibliographic data from sources on the web. The enriched reference is passed on to the next integration handler of the Integration phase.

Observations showed that for some queries based on the title, BibSonomy finds no results, while a query based on the author name does find relevant bibliographic data. For example, the query for the publication 'Leases An efficient faulttolerant mechanism for distributed file cache consistency' shows no results[11], while BibSonomy does find relevant bibliographic data when we query for the name of the author, 'Cary Gray'[12].

To be sure BibSonomy has no relevant data about the publication we are looking for, the second integration handler is introduced. This handler is skipped when the previous step did find relevant data, because it is not likely that the query based on the author names will find new relevant data. This is the case with the current example about the publication of G. C. Necula, the first integration handler integrated bibliographic data, so this step was skipped. The validation step checks if the input reference contains author names. If the reference passes the validation, the retrieval step formulates queries based on the author names from the input reference. Such queries often result in huge responses, representing all kinds of publications from all of the authors. The filtering step is very important here, because the response contains irrelevant data for the greater part. The references that remain after filtering are integrated just like in the previous integration handler. The resulting enriched reference is passed on to the next integration handler.

The third and last integration handler tries to find additional relevant bibliographic data by targeting a different data source: The Collection of Computer Science Bibliographies (CCSB). CCSB is a collection of bibliographies of scientific literature in computer science from various sources, covering most aspects of computer science. It contains over 3 million references in BibTeX format.

[11]Results for 'Leases An efficient faulttolerant mechanism for distributed file cache consistency' in BibSonomy, http://www.bibsonomy.org/search/Leases+An+efficient+faulttolerant+mechanism+for+distributed+file+cache+consistency, last visited on August 26, 2010

[12]Results for 'Cary Gray' in BibSonomy, http://www.bibsonomy.org/search/Cary+Gray, last visited on August 26, 2010

The data source CCSB has different characteristics than the BibSonomy web-service. The retrieval process is different, because CCSB does not provide us with a web-service to communicate with. In order to retrieve relevant bibliographic data from CCSB, the CCSB search engine is queried and the the web page containing the search results is parsed. The CCSB search results contain links to pages where BibTeX formatted data can be found. These pages are scraped and the BibTeX formatted data is parsed. Just like in the previous integration handler, the parsed data is transformed into a set of references which are filtered and integrated. Note that during the integration step, the mistake with the pages attribute was repaired. The CCSB is a collection of bibliographies and it contains several different BibTeX entries representing the same publication we are looking for. Using those different representations, the integration step derives that the pages attribute was wrong, because most of the representations contain the page numbers '106-119'. Besides the pages attribute, also the publisher, organization, month and address are integrated with the input reference (see Table 3).

After the CCSB integration handler, there are no succeeding handlers left, so the result of the third step is also the end result of the approach. The result is a BibTeX modelled reference which contains data citation metadata enriched with bibliographic data from sources on the web.

4 Conceptual Model

We have implemented the enrichment approach we introduced in the previous chapter. The implementation is called BibMix. In this chapter we provide a detailed description of the design of BibMix.

The chapter is structured as follows. First, Section 4.1 briefly introduces the reader to the conceptual model illustrated by a UML Class Diagram shown in Figure 8. In Section 4.2 we focus on the classes involved in the Extraction phase. Finally, in Section 4.3 we describe the classes involved in the Integration phase.

4.1 Global Overview of the Conceptual Model

This section provides an overview of the conceptual model, illustrated by an abstract UML Class Diagram in Figure 8. The model consists of several classes (rectangles), relations (lines ending in open arrows), relation cardinalities. Although most of the classes depicted in the model are generalizations, their subclasses are left out for readability reasons.

The `Pipeline` is the main class responsible for directing all the steps involved in the Extraction and Integration phases. The pipeline is also the interface other applications will interact with: they give a citation string and receive an enriched reference in return.

Before each run, the pipeline initializes the `CMEApplication`, `MetadataProcessor`, `IntegrationHandler` classes. Then the Extraction phases and Integration phase are executed consecutively. The Sequence Diagram in Figure 9 illustrates the sequence of messages exchanged by the Pipeline and objects after initialization. Sections 4.2 and 4.3.5 respectively provide detailed descriptions of the Extraction and Enrichment phases.

4.2 The Extraction Phase

The Extraction phase is responsible for the extraction of metadata from citations and makes sure the Integration phase can use the extracted metadata. In this section we describe the classes that are involved.

4.2.1 Citation Metadata Extraction

BibMix employs CME during the Extraction phase. It makes use of the ParsCit CME application[13] version 100401d. We've chosen ParsCit because it shows a state-of-the-art performance (Councill and Kan, 2008) and because it is under active development. Furthermore, it is open-source software and can be freely downloaded from the project page.

ParsCit is capable of citation metadata extraction and logical structure parsing of scientific documents, however the structure parsing feature is not used.

[13]ParsCit project page, `http://aye.comp.nus.edu.sg/parsCit/`, late visit on July 11, 2010

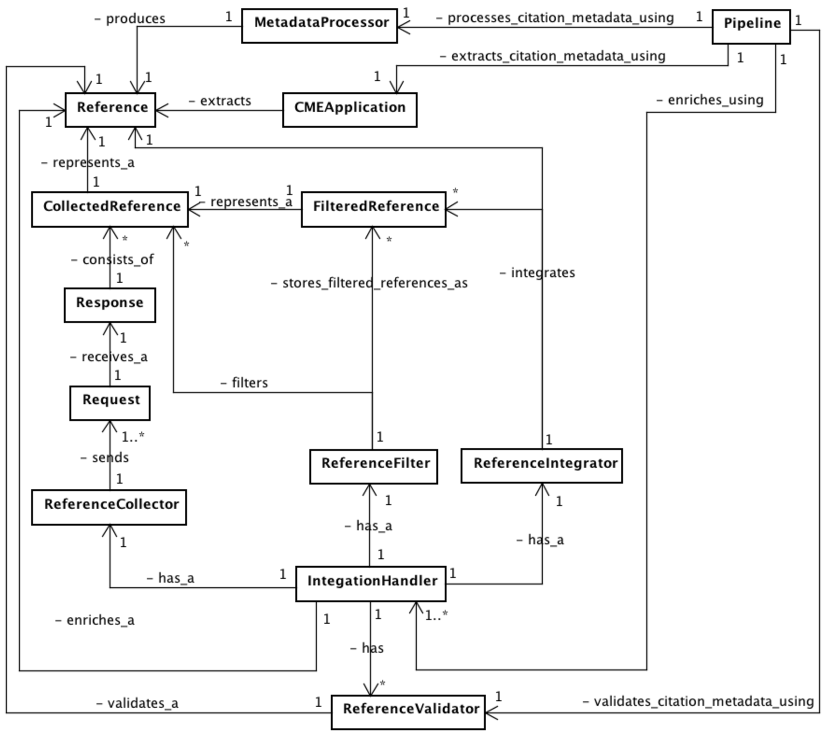


Figure 8: The conceptual model of BibMix illustrated by an abstract UML Class Diagram.

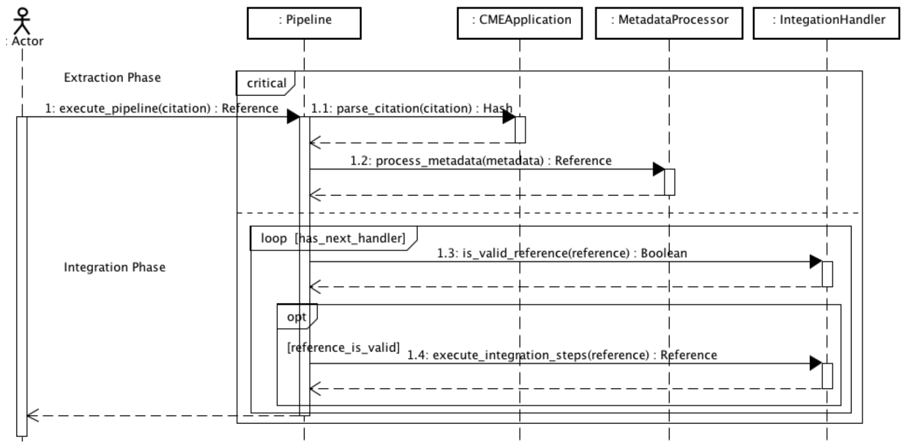

: Actor
: Pipeline
: CMEApplication
: MetadataProcessor
: IntegationHandler
Extraction Phase
critical
1: execute_pipeline(citation) : Reference
1.1: parse_citation(citation) : Hash
1.2: process_metadata(metadata) : Reference
Integration Phase
loop [has_next_handler]
1.3: is_valid_reference(reference) : Boolean
opt
[reference_is_valid]
1.4: execute_integration_steps(reference) : Reference

Figure 9: A UML Sequence Diagram of the BibMix Pipeline after initialization.

It is architected as a supervised machine learning procedure that uses Conditional Random Fields (CRFs) as its learning mechanism. ParsCit is open-source software and can be downloaded from the project page.

BibMix uses ParsCit with the default settings and default CRF model. The publicly available implementation of ParsCit comes loaded with a model trained over the full Cora dataset. This dataset is derived from one of the first studies in automated reference string parsing (Seymore et al., 1999). This dataset created a gold standard for 500 reference strings sampled from various computer science publications. These citations were segmented into thirteen different fields - 'author', 'booktitle', 'date', 'editor', 'institution', 'journal', 'location', 'note', 'pages', 'publisher', 'tech', 'title', and 'volume' - reflective of BibTeX fields that might be used to generate the references themselves.

The `ParsCit` class (a sub-class of the `CMEApplication` class) is the interface to ParsCit. It accepts a citation. This citation is given to the command-line interface of ParsCit which uses its machine learning procedure to segment the citation into the aforementioned thirteen different fields. The resulting structure is returned and passed on to the `MetadataProcessor` class, which is discussed in the next paragraph.

4.2.2 Reference Model

A `ParscitMetadataProcessor` (sub-class of `MetadataProcessor`) takes the structured metadata from ParsCit and transforms it into a special reference model: `Reference`. The `Reference` instance representing the extracted citation metadata is given to the first `IntegrationHandler` in the Integration phase.

The reference model is introduced so that the collection, filtering and integration steps in the Integration phase share a common data structure representing bibliographic information. Especially for the filtering and integration steps it is convenient that the input reference and the collected and filtered references are represented by the same model. For example, the filtering mechanisms expects both the collected references and input reference to have the same attributes, otherwise the comparison between two instances is not possible without a mapping. The same counts for the integration step: when filtered references and input reference have a different structure, it is not possible to integrate the filtered references with the input reference without a mapping.

4.3 The Integration Phase

The Integration phase consists of a sequence of `IntegrationHandler` classes. In this Section we describe the handlers and their sub-steps in detail. We begin with the explanation of the choice to use a sequence of handlers instead of collecting, filtering and integrating all bibliographic data at once. In addition, we explain how we decided on the order of the sequence of `IntegrationHandler` classes.

4.3.1 Integration Handlers

The advantage of a sequence is that it enables the approach to deal with varying characteristics of data sources and their data. For example, the sources may differ in their location, the quality of the provided data and the protocol, i.e. the way data can be collected/queried and the format of the data. Different handlers can be configured so that they can make optimal use of the data source and the data.

Another advantage is that later steps may benefit from the bibliographic data integrated in earlier steps, e.g. the querying and filtering possibilities increase as the input reference is richer. For example, in the filtering step of the first `IntegrationHandlers` in the sequence, the filtering step has to compare the collected references with the reference that represents the extracted citation metadata. If the step integrates new relevant bibliographic data, the succeeding `IntegrationHandlers` receive a richer input reference, and thus the similarity assessment of collected references and the input reference can be based on more attributes. This results in an increase of the comparison accuracy.

As mentioned in Section 3.2 the Integration phase consists of three integration handlers (each represented by a `IntegrationHandler` class). The order of the `IntegrationHandlers` is decided on the order of which the data should be collected and integrated. The outcome influences the quality of the end result of the enrichment approach: the data that is collected and filtered by the first integration handler has the highest probability to be integrated with the input reference (for an explanation of the integration see Section 4.3.5). This is because the input reference only contains citation metadata before it is integrated with relevant bibliographic data.

The order of the `IntegrationHandlers` in BibMix was conceived based on the characteristics of the two data sources shown in Table 4. We believe that the bibliographic information from BibSonomy is of higher quality than the data of the CCSB. BibSonomy has less contradicting duplicate references and the provided BibTeX is clean, i.e. the provided BibTeX can be parsed without replacement of unknown characters and has no character encoding issues. In addition, BibSonomy imports the contents of DBLP[14]. DBLP is well-known computer science bibliography listing more than 1.4 million publications. The contents of DBLP are of high quality due to the fact that all publications entries are manually entered(Ley, 2002).

We have decided to select the following order of `IntegrationHandlers`:

1. The BibSonomy web-service is queried based on the title of the reference.
2. The BibSonomy web-service is queried based on the author names of the reference.
3. The Collection of Computer Science Bibliographies (CCSB) is queried based on the title of the reference.

[14]DBLP, http://www.informatik.uni-trier.de/~ley/db/, last visited on August 24, 2010

	BibSonomy	**CCSB**
Provides	User contributed publication data + DBLP[15]	Collection of computer science bibliographies of scientific literature[16]
Domain	Various domains	Computer Science
Duplicate Entries	Not many	Lots
Quality of the provided data	Ok	Varying

Table 4: A comparison of the BibSonomy and CCSB data sources.

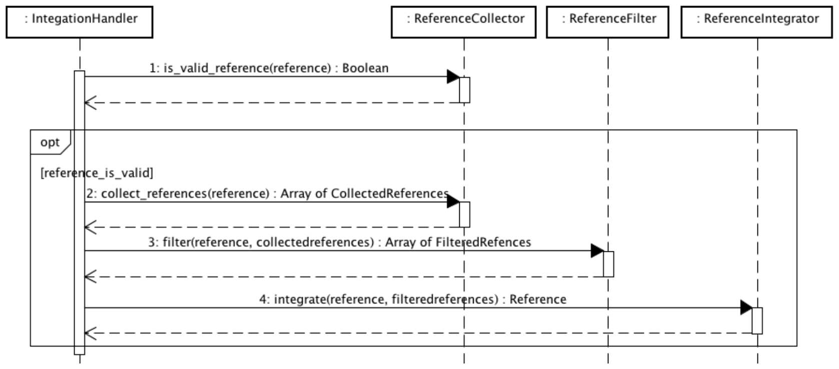

: IntegationHandler
: ReferenceCollector
: ReferenceFilter
: ReferenceIntegrator
1: is_valid_reference(reference) : Boolean
opt
[reference_is_valid]
2: collect_references(reference) : Array of CollectedReferences
3: filter(reference, collectedreferences) : Array of FilteredRefences
4: integrate(reference, filteredreferences) : Reference

Figure 10: A UML Sequence Diagram of the messages exchanged by a `IntegrationHandler` and sub-components.

Adopting the separation-of-concerns principle, the design of BibMix distinguishes four different steps in each IntegrationHandler: validation, retrieval, filtering and integration (see the Sequence Diagram in Figure 10). Each of these steps is represented by a different class, we discuss these in the paragraphs below.

4.3.2 Validation

The integration procedure of each `IntegrationHandler` starts with the validation of the input reference. The validation is handled by one or more `ReferenceValidator` instances. Each validator instance inspects a certain requirement or a set of requirements the input reference should comply to. When the reference fails the validation, the reference is passed on to the next `IntegrationHandler` in the chain. The conditional execution of the collection, filtering and integration steps is illustrated in the Sequence Diagram in Figure 10. Handlers that have no `ReferenceValidators` impose no requirements on the input reference, and thus start the retrieval process right away.

Input reference validation is done to ensure the reference contains enough information so that later steps in the `IntegrationHandler` can locate and identify the publication the input reference represents. Furthermore, the validation step can be used to prevent the execution of `IntegrationHandlers`. For example, see the case described in Section 3.2 where the input reference of the second `IntegrationHandler` only passes the validation (and starts the integration procedure) if it contains author names and is not yet enriched by the previous step.

4.3.3 Retrieval of Bibliographic Data

The `ReferenceCollector` class collects bibliographic data from sources on the web. A collector collects data from a single data source using one or more `Requests`. For example, the Integration Handler from the example in the previous chapter (see Section 3.2) which queried the BibSonomy web-service based on the title uses only a single request because there is only a single title. On the contrary, the second Integration Handler from the example which queries the BibSonomy web-service on the names of the author of a publication, needs several requests when there are several authors.

The request that is sent is based on the input reference of the Integration Handler. The response of the request is parsed and processed by the `Response` concept. It transforms the collected data into a set of `CollectedReferences`. For example, a collector formulates a query for the BibSonomy web-service and sends a request. The web-service responds with bibliographic data in XML format. The `Response` concept parses the XML and creates a set of `CollectedReferences` from it. The `CollectedReferences` also store the origin of the reference it represents, this information can be used for quality assessment of the reference during the integration step.

4.3.4 Filtering of Collected References

The references that are collected by the `ReferenceCollector` are given to the `ReferenceFilter`. The filtering component is responsible for filtering out irrelevant references. It does so by comparing each of the `CollectedReferences` to the input reference of the `IntegrationHandler`. The similarity between the two references is expressed as a numeric value. The filter filters out irrelevant references, based on a configurable similarity threshold. The remaining collected references are stored as `FilteredReferences`, along with the similarity value, which is used in the integration step.

The filtering of references in the `ReferenceFilter` class is done by a Record Linkage tool called FRIL (Fine-grained Records Integration and Linkage tool). FRIL provides a rich set of tools for comparing records, in our case, references. Furthermore, FRIL provides a graphical user interface for configuring the comparison of references (see Figure 11). At design time, developers may systematically and iteratively explore the optimal combination of parameter values to enhance comparison performance and accuracy. Section 5.3 explains more about FRIL.

The configuration of FRIL is vital for the performance and accuracy of BibMix. If the configuration is too strict, relevant information is not integrated because it is not seen as relevant. If the configuration is too tolerant, irrelevant references are integrated.

Distance metrics are one of the most important components available in FRIL. They provide FRIL with the information how to compare the attributes of references. By specifying a distance metric between two attributes, FRIL knows how the attributes should be treated (as strings, numbers or dates) and what the allowable discrepancies between values of the attributes are. The distance function maps similarity between values of attributes onto a value in the range of 0-1, where 0 means 'are not similar in terms of this distance function' and 1 means 'are exactly the same in terms of this distance function'.

In the configuration we make use of three distance metrics, i.e. Numeric distance, Pair Similarity distance and Equal Fields Boolean distance. These metrics are described briefly below. For more detailed descriptions we refer to the FRIL tutorial document[17].

The Numeric distance metric treats input values as numbers (real values). This metric allows to specify a range of values that will have a non-zero match score. Furthermore, it is possible to have a range specified by fixed value. For example, if the configuration says the range is a radius of 2, and the value is 10, the metric matches 10 to all values between 8 and 12. The score is calculated using the following formula:

$$score_{numeric}(v1, v2) = \begin{cases} 1 & if\ v2\ \in\ [lowerbound(v1)...upperbound(v1)] \\ 0 & otherwise \end{cases}$$

The Equal Fields Boolean distance metric treats input values as raw strings. This distance function returns only two values, 0 or 1. 0 is returned if compared values are different, 1 if are the values are exactly the same. The score is calculated using the following formula:

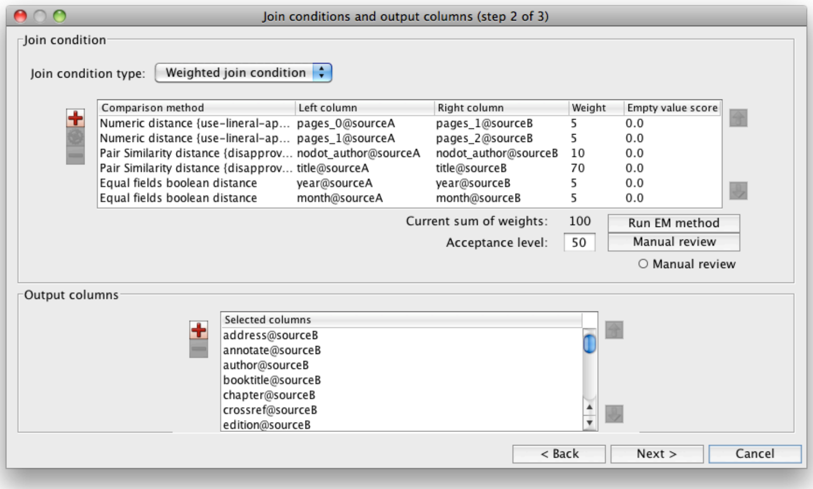
$$score_{equalfields}(v1, v2) = \begin{cases} 1 & if\ v1\ is\ equal\ to\ v2 \\ 0 & otherwise \end{cases}$$

The third and last metric is the Pair Similarity distance. This is a metric we have implemented and added to FRIL, because it is has a good overall performance and accuracy. The Pair Similarity distance between two strings is based on the number of adjacent character pairs, that are contained in both strings.

The FRIL configuration requires a weight for each of the distance metrics and the sum of weights has to be equal to 100. As mentioned before, each distance metric returns a value between 1 and 0. To compute the similarity between two references, FRIL multiplies the weight of each condition by the value returned by the chosen distance metric. Then, those numbers are added together to calculate a total score for given pair of references.

For BibMix, we have chosen not to compare references based on all attributes, but on just five attributes. This is because the FRIL configuration

[17]FRIL Tutorial v3.2, http://fril.sourceforge.net/FRIL-Tutorial-3.2.pdf, last visited on August 25, 2010



Join condition type: Weighted join condition

Comparison method	Left column	Right column	Weight	Empty value score
Numeric distance {use-lineral-ap...	pages_0@sourceA	pages_1@sourceB	5	0.0
Numeric distance {use-lineral-ap...	pages_1@sourceA	pages_2@sourceB	5	0.0
Pair Similarity distance {disapprov...	nodot_author@sourceA	nodot_author@sourceB	10	0.0
Pair Similarity distance {disapprov...	title@sourceA	title@sourceB	70	0.0
Equal fields boolean distance	year@sourceA	year@sourceB	5	0.0
Equal fields boolean distance	month@sourceA	month@sourceB	5	0.0

Current sum of weights: 100

Acceptance level: 50

Selected columns
address@sourceB
annotate@sourceB
author@sourceB
booktitle@sourceB
chapter@sourceB
crossref@sourceB
edition@sourceB

Figure 11: The comparison configuration screen in FRIL.

requires a weight for each of the distance metrics and the sum of weights has to be equal to 100. If we would compare all attributes, then the weight has to be divided over twenty four instead of five attributes. This means that relevant references with just a few attributes will have a low similarity number. For example, we compare two references that both have a title, author and month attribute. Furthermore, the reference that is from an external source also has the booktitle attribute. Generally speaking, these attributes would be sufficient for a human being to assess the similarity between the references. However, the weights are divided over all the attributes, so the combined scores for the matching attributes will not be enough to get through the filtering step. Therefore, we compare the attributes of references that are present in most references.

The configuration used by BibMix has six comparisons methods, based on five attributes, e.g. pages, author, title, month and year (the order is not relevant). The pages attribute is splitted before comparison. The resulting numbers are compared using the Numeric distance metric with a radius of 2, as it seemed that some similar references had different page numbers (probably caused by input errors). The month and year attributes are compared using the Equal Fields Boolean distance metric. Each of the aforementioned metrics have a weight of 5 (see Figure 11). Finally, the title and author attributes are compared using the Pair Similarity distance metric. The author attribute has weight 10 and the title attribute is weighted 70. The author weight for the author attribute is low compared to the weight of the title. The comparison of author names is difficult because the order in which the authors are listed may differ. Furthermore, author names may be abbreviated or not. The high weight for the title attribute is justified by the fact that it is present in practically every

	Reference 1	Reference 2	Similarity	Weight	Score
Pages-start	106	106	1	5	5
Pages-end	119	120	1	5	5
Month	January	February	0	5	0
Year	1997	1997	1	5	5
Author	G C Necula	George C Necula	0.87	10	8.7
Title	Proof-carrying code	Proof carrying code	0.93	70	65.1
				Total score	88.8

Table 5: An example illustrating the calculation of the similarity number of two references.

reference and it is the attribute that is unique most of the times.

We illustrate the calculation of the similarity number of two references with an example. The references 'reference 1' and 'reference 2' both represent the citation from 1. The reference 'reference 1' is the input reference, while 'reference 2' represents a reference that is collected from a source on the web. Table 5 shows the the attributes of the references that are compared, the score computed by the corresponding distance metric, the weights and the calculated score. The 'Pages-start' attribute is the first page number, the values for this attribute are equal in both references, so the score is calculated by multiplying the similarity score by the weight: $1 \times 5 = 5$. The same counts for the 'Pages-end' attribute, because of the range we configured 119 and 120 are matched. The month attributes are unequal and according to the Equal Fields Boolean distance metric the similarity score is 0, the calculated score of this comparison is $0 \times 5 = 0$. Note that the year attributes are equal so the calculated score for this attribute is $1 \times 5 = 5$. The Pair Similarity distance metric judges that the author and title attributes of both references are kind of equal, the scores for the attributes are 0.87 and 0.93 respectively. The total of all calculated scores forms the basis for the similarity number (shown in the bottom row of Table 5). BibMix divides the total score by 100. The resulting similarity number is then compared to a threshold. If the similarity number is higher than the threshold, the reference is integrated.

A final check in the filtering step makes sure that collected references representing a conference article are not mixed with the input reference when the input reference represents a journal. The other way around also checked, when the input reference represents a conference article, all collected references that represent a journal are filtered out.

4.3.5 Integration of Filtered References

The integration of bibliographic data from data sources on the web with the input reference of the `EnrichmentHandler` is performed by a `ReferenceIntegrator`. As shown in Figure 10 the integrator receives a set of `FilteredReferences` from the handler. The integrator decides how the bibliographic data is integrated. In addition, it also decides how integration conflicts are solved.

When integrating, the `ReferenceIntegrator` loops over the `FilteredReferences` instances in descending order of similarity (based on the similarity numbers stored by the `FilteredReferences`). The empty attributes of the input reference are filled with the attributes of the filtered reference. When the input reference already has a value for a certain attribute, the value of the attribute of a filtered reference is ignored. The order ensures that the most similar references are integrated first. This method we call 'naive' integration, because only new values of previously empty attributes are added to the input reference.

After the 'naive' integration, the `ReferenceIntegrator` investigates two special cases where the values of attributes of the input record are replaced with the values from `FilteredReferences`. The first case checks if the filtered references have higher quality author names than the input record. ParsCit normalizes all author names: the first names are abbreviated with only the first character, e.g. 'George C. Necula' becomes 'G C Necula'. The integrator first separates the first and last names from each other in both input and filtered references. Then it starts aligning names where the last names are similar. Finally it checks if the abbreviated first name(s) from the input reference coincide with the first character of the first name(s) in the filtered reference. For example, we have an input reference with the author name 'G C Necula' and a filtered reference with the author name 'George C. Necula'. The last names are separated from the first names. The last names are both 'Necula', so we go on with the comparison of the first names. Both the abbreviated first names are equal to the first characters of the first names in the filtered reference. The value for the author attribute in the input reference will be replaced by the author name in the filtered reference.

The other special case is with the pages attribute. ParsCit often mistakes the publication date written as '15-17 January' for the page numbers of a publication. This actions that follow are only executed when the highest page number in the input reference is 31 or below, because a publication date of '45-89 January' is not likely. When this requirement is met and there are two or more filtered references that both have different page numbers than the input reference, the value for the pages attribute in the input reference will be replaced by the pages in the filtered references. Note that when there are several different page numbers present in the filtered references, the pages attribute in the input reference remains untouched because of the uncertainty the right page number is chosen. Here's an example illustrating the case: we have an input reference with page numbers '15-17' and two filtered references both having pages '106-119'. The highest page number is 17 and it is lower than 31, so the page numbers of the filtered references are checked. Both mention page numbers '106-119', now there is a high probability ParsCit has made a mistake, so the value for the pages attribute in the input reference will be replaced by the pages stored in the filtered references. If there was a third filtered reference with page numbers '45-89', the input reference remained intact.

5 BibMix Implementation

BibMix implements the approach introduced in Section 3. This implementation provides a means for evaluating the effectiveness of the approach designed during this thesis project. In this chapter we describe the implementation in terms of decisions made during implementation and tools that are used.

This chapter is structured as follows. First, Section 5.1 presents the implementation details of BibMix. The tools that are used by BibMix are detailed in Section 5.3.

5.1 Implementation Details

BibMix is implemented in the Ruby programming language[18]. To be more precise, it is developed as a plug-in for Ruby on Rails (RoR). Ruby provides tools to establish flexible and reusable software. RoR provides means for easy application deployment, excellent testing facilities. Furthermore, RoR has a plugin-in structure that was perfectly suited for BibMix. Plug-ins are used to package self-contained libraries for reuse within other RoR applications.

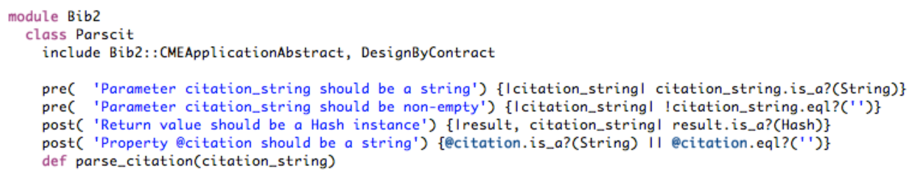
5.2 Design by Contract

Ruby is a very flexible language and practically any type of data structure can be implemented in the language. However, Ruby is a dynamically typed language and does not have support for abstract classes and interfaces. This is inconvenient when aiming for formal, precise and verifiable interface specifications for software components. To make sure this formal interface is verifiable and consistent, we implemented most classes based on a paradigm called Design by Contact (DbC) (Meyer, 1986). DbC prescribes that software components should agree on a contract, which describes the preconditions, postconditions and invariants of classes that are part of the components.

For BibMix we have chosen to enforce the DbC paradigm in two ways, e.g. unit-testing and pre- and postcondition checking of method invocations.

Martin Traverso and Brian McCallister have developed a module imple-

[18]The Ruby Language, http://www.ruby-lang.org/en/, last visited on August 4, 2010



```
module Bib2
  class Parscit
    include Bib2::CMEApplicationAbstract, DesignByContract

    pre(  'Parameter citation_string should be a string') {|citation_string| citation_string.is_a?(String)}
    pre(  'Parameter citation_string should be non-empty') {|citation_string| !citation_string.eql?('')}
    post( 'Return value should be a Hash instance') {|result, citation_string| result.is_a?(Hash)}
    post( 'Property @citation should be a string') {@citation.is_a?(String) || @citation.eql?('')}
    def parse_citation(citation_string)
```

Figure 12: DbC applied in the Ruby programming language.

menting the DbC approach in Ruby[19]. Figure 12 shows an small part of a class definition where the Ruby DbC approach is applied to an instance method. Using the `pre` and `post` statements the pre- and post conditions are checked before and after method invocations. As an added benefit, the pre- and post conditions serve to document the inner workings of the methods and thereby enhancing the maintainability of the code.

Unit-tests provide a strict, written contract that a tested piece of code must satisfy. Ruby on Rails provides excellent unit-testing facilities. The components of BibMix are unit-tested.

5.3 Tools Used

BibMix makes heavy use of the ParsCit CME application and the Record Linkage tool called FRIL. These tool are already discussed in Section 4.

5.3.1 Parsing BibTeX Records

BibMix incorporates two tools for parsing BibTeX records scraped from the CCSB search result pages. The first tool, called `aurels-rbib`[20] is a tool for parsing BibTeX written in Ruby. The second tool, `Bibutils`[21], this is utilized for cleaning up badly formatted BibTeX.

When the BibTeX records collected from the CCSB need to be transformed into a set of `CollectedReference` instances, the records are first parsed using `aurels-rbib`. In most cases this goes well, however, the data from the CCSB sometimes contains illegal characters that are not handled correctly by the parser. Then BibMix passes the invalid BibTeX to `bib2xml`, which converts BibTeX to a MODS XML intermediate format. MODS[22] (Metadata Object Description Schema) is a schema for a bibliographic element set that may be used for a variety of purposes, and particularly for library applications. Consecutively, BibMix executes `xml2bib` on the output of `bib2xml`. This converts the MODS XML representation back to the BibTeX format. The resulting BibTeX record contains the same information as the original record and the illegal characters are converted to characters that are handled well by `aurels-bib`.

[19]DbC for Ruby, http://www.koders.com/ruby/fidF1C36153377AD691CE9581FF479D0BE1C6F69E19.aspx?s=game, last visited on August 10, 2010

[20]Aurels-rbib Project Page, http://github.com/aurels/rbib, last visited on August 25, 2010

[21]Bibutils Project Page, http://www.scripps.edu/~cdputnam/software/bibutils/, last visited on August 25, 2010

[22]MODS Project Page, http://www.loc.gov/standards/mods/, last visited on August 25, 2010

Part III
Evaluation

6 Evaluation Introduction

This chapter provides an overview of events that affect the quality of the enriched references and introduces a measure for the quality of enriched references.

6.1 Events Affecting the Reference Quality

The quality of the enriched output reference of BibMix is dependent on two main events. The first event is that BibMix did not select the right data source to collect data from. The other event is that BibMix did not retrieve the right bibliographic data from the selected data sources.

In our prototype implementation the same data sources are selected every time a reference is enriched, so during the evaluation it will not occur that the wrong data sources are selected. However, if this error would occur, the result will be that the reference contains less attribute values, given that the data is not provided by an other data source.

The event that BibMix does not retrieve correct bibliographic data from a selected data source, given that the data source does contain the correct data, may be caused by a number of reasons.

1. **Errors in the extraction of citation metadata** When the citation metadata that is extracted during the Extraction phase contains an error for attributes that are used for the formulation of a query for a data source, the data source might return incorrect bibliographic data. For example, the after the metadat was extracted from the citation 'B. L. Peuto and L. J. Shustek. An instruction Timing Model of CPU Performance, Proceedings of the 4th Annual Symposium on Computer Architecture, Maryland New York, March 1977, 165-178.', the 'title' attribute value was 'An instruction Timing Model of CPU', while the 'booktitle' attribute value was 'Performance, Proceedings of the 4th Annual Symposium on Computer Architecture'.

2. **Errors in the citation string** When the evaluation dataset contains a citation which mentions the author 'B ChazeUe', while the real name of the author is actually 'B Chazelle', the data is integrated most likely. This error is probably the result of Optical Character Recognition (OCR). OCR is the automatic translation of scanned images of handwritten, typewritten or printed text into machine-encoded text. Furthermore, the citation 'Ashley, K. D. and Rissland E. L., Accord: Generating Blue Book Citations in HYPO. Proceedings of 1st International Conference on Artificial Intelligence Law ICAIL-871, pages 67-14. New York: ACM Press.' suffers from a similar error. The page numbers are actually '67-74'.

3. **Data source provides incorrect data** In some cases, the data source provides bibliographic data that is not correct. For example, the CCSB returns the address 'pub-IEEE:adr' for the citation 'Xiaogang Qiu and Michel Dubois. "Options for Dynamic Address Translation for COMAs",

In Proceedings of the 25th Annual International Symposium on Computer Architecture ISCA, pages 214-225. 1998.'. Another example of where the CCSB provides incorrect data is for the citation 'Ware C. and Osborne, S. "Exploration and virtual camera control in virtual three dimensional environments," Proceedings of the 1990 Symposium on Interactive 3D Graphics (Snowbird, Utah). In ACM Computer Graphics, 24(2) March 1990, pages 175-183.'. This citation represents a journal, however, the provided data contains a reference for which the journal name was stored by the 'booktitle' attribute.

6.2 Reference Quality Assessment

To express the quality of a reference as a number, we have to understand what a reference is. A reference is a structured representation of a publication. The quality number should say something about how good a reference is at representing a publication. The measure we introduce in this section is created for the evaluation of enriched references of the implementation of the approach we propose. Furthermore, the quality measure is designed to be used in future projects to compare enrichment results to the results of this thesis project.

The attribute values should be conform the BibTeX reference model specification which define both the syntactics as the semantics of each attribute(Patashnik, 1988) (see Appendix B).

A common place where publications are represented by references are online digital libraries. Therefore, we looked at the characteristics of references (their attributes) in the datasets of several digital libraries in order to define a quality measure for references. We use the number of references that have a value for a given attribute in the DBLP[23] dataset to compute a weight for the attribute. The quality of a reference is the sum of the weights for the correct attributes of the reference. We chose the DBLP dataset because it is a large bibliography, containing references of publications on miscellaneous topics in the Computer Science domain (the domain we are targeting with our evaluation dataset). Furthermore, DBLP is manually authored. We did not combine the attribute statistics of DBLP with other bibliographies in order to calculate the weights of the attributes because this might have biased the attribute weights due to duplicate references.

We have assigned a weight to 22 attributes of the reference model. These weights are based on attribute occurrences in 2267820 BibTeX modelled references from the DBLP dataset. The 'crossref' attribute has a value in 297.783 references in the DBLP dataset, however, it is not assigned a weight for the reason that it is used for cross-referencing other references in the same dataset. Cross-referencing means that any fields that are missing from the a reference are inherited from the reference being cross-referenced. Because we collect data from a number of sources, this attribute loses its function. Moreover, it does not say anything about the publication the reference refer to. The same applies

[23]DBLP, http://www.informatik.uni-trier.de/~ley/db/, last visited on August 30, 2010

Attributes	**Occurrences in DBLP**	**Weights**
Address	49	0.01
Annotate	0	0.01
Author	2255899	0.17 or 0.13
Booktitle	1306019	0.10
Chapter	2	0.01
Crossref	297783	0
Edition	0	0.01
Editor	504801	0.03
Howpublished	0	0.01
Institution	0	0.01
Journal	557509	0.04
Key	0	0
Month	2524	0.01
Note	24669	0.01
Number	509852	0.03
Organization	0	0.01
Pages	1328758	0.10
Publisher	732625	0.06
School	100	0.01
Series	310060	0.02
Title	2267820	0.17
Type	0	0.01
Volume	859730	0.06
Year	1431361	0.11

Table 6: The number of occurrences of attributes in the DBLP dataset and the weight of the attributes.

to the 'key' attribute, which is used for 'alphabetizing, cross referencing, and creating a label when the "author" information is missing' (from the BibTeX specification (Patashnik, 1988)).

The sum of the weights for the attributes should be 1. We have assigned a weight of 0.01 to the attributes that had a value in less than 100.000 (except for 'crossref' and 'key') references, otherwise their weight would have been negligible or 0. The remaining weight of 0.89 was divided over the other attributes.

The 'author' attribute is an exceptional case. This attributes weight depends on the form of the first name(s) of the author(s) of a publication. When the first name of an author is abbreviated,e.g. G. C. Necula, the 'author' attributes' weight is 0.13 (roughly 3/4 of the full score). When the first name is present in its full form,e.g. George C. Necula, the weight is 0.17.

The result is shown in Table 6. So for example, if we have a reference containing correct values for the 'author' (with a full firstname), 'title', 'institution' and 'crossref' attributes, the quality of the reference would be computed by $0.17 + 0.17 + 0.01 + 0 = 0.35$.

7 Evaluation Approach

For the evaluation of BibMix, we will perform two different evaluations. The first will evaluate the performance of the Extraction phase, the second will evaluate the performance of the Integration Phase.

This chapter is structured as follows. Section 7.1 presents the dataset used in this evaluation. In Section 7.2 we introduce the reader to the evaluation of the performance of the Extraction phase. Finally, in Section 7.3 we describe the evaluation for the quality of the enriched output references of BibMix.

7.1 Evaluation Dataset

As mentioned before evaluation will be targeted at the domain of Computer Sciences publications. We use a dataset that was composed for the evaluation of the CME component FLUX-CiM (Cortez et al., 2007). They made available their experimental data set for the sake of future comparison[24]. The data set is a heterogeneous collection composed by assorted references from several conferences and journals in the Computer Science area. The dataset consists of 300 citation strings randomly selected from the ACM Digital Library[25].

7.2 Evaluating the Performance of the Extraction Phase

For CME applications it is common to express the performance of the CME in terms of precision and recall. For the evaluation of the Extraction phase of BibMix do so as well. The formulas for precision and recall are given below:

$$precision = \frac{|correctly\ extracted\ attributes|}{|extracted\ attributes|}$$

The precision is the percentage of extracted references that have no attributes that were incorrectly extracted. An example of an incorrect extraction would be that the date '15-17 January' is mistaken for page numbers.

$$recall = \frac{|correctly\ extracted\ attributes|}{|corrent\ attributes|}$$

The recall indicates the probability that BibMix did extract a value which actually is present in the citation. For example, a citation contains the publication address, but the 'address' attribute was not part of extracted metadata.

The precision and recall are highly and inversely correlated. For example, if we have configured BibMix to be really strict about the collected references to integrate, the precision will be high, however, the recall will be low because BibMix leaves out several relevant attribute values. Therefore we introduce a

[24]FLUX-CiM Computer Science dataset, http://www.dcc.ufam.edu.br/~eccv/flux-cim, last visit on July 9, 2010

[25]ACM Digitial Library, http://portal.acm.org/dl.cfm last visit on July 9, 2010

third metric, the F1 measure. The F1 measure is the harmonic mean of precision and recall and gives a better overall impression of the performance than recall and precision, as it incorporates both.

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

To compute the precision, recall and F1 metrics, we have manually constructed a dataset that are the correct extraction results. During the evaluation we will compare the extraction results with the results from the manually constructed dataset.

7.3 Evaluation of the Quality of the Enriched References

In Section 6.2 we introduced a formula to express the quality of a reference as a number. With the evaluation we will compute the quality numbers for the enriched references. To do so, we will compare the output references of BibMix to those provided by a human expert. An attribute value of an enriched reference is marked as correct when it is equal to the attribute value provided by the human expert. The evaluation result will be the average of the quality numbers for the enriched references of the evaluation dataset.

8 Evaluation

In this chapter we present the results of the evaluation that were described in the previous chapter.

8.1 Evaluation of the Extraction Phase Performance

We have performed the evaluation described in Section 7.2. The results are shown in Table 7. Note that only the performance of the attributes that were present in the citations is shown, except for the month attribute. ParsCit was not trained to extract months from citations.

	Precision	**Recall**	**F1**
Address	85.7%	65.2%	74.1
Author	87.0%	100.0%	93.0
Booktitle	91.1%	100.0%	95.3
Journal	100.0%	85.7%	92.3
Pages	83.3%	95.6%	89.0
Publisher	100.0%	50.0%	66.7
Title	95.0%	100.0%	97.4
Volume	100.0%	75.0%	85.7
Year	100.0%	98.9%	99.4
Averages	93.6%	85.6%	88.1

Table 7: The evaluation results of the Extraction phase in terms of the precision, recall and F1.

8.2 Evaluation of the Quality of the Enriched References

We have performed the evaluation described in Section 7.3. The results are shown in Figure 13. The average score computed was 0.72. Approximately 70% of the references had the full score for the 'author' attribute.

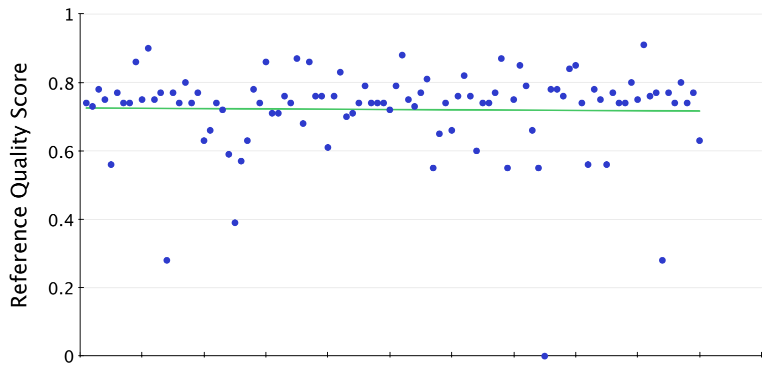


Figure 13: A scatter plot of the reference quality scores.

This figure clearly shows that a few enriched references have a low score. The points that were lowest are all caused by errors in the extraction results (errors of type 1 from Section 6.1. Due to the errors in the extraction, no relevant data could be found. Note that there is one enriched reference which is completely wrong (its quality score is 0). This was the citation 'Wan, W., Perkowski, M.: A New Approach to the Decomposition of Incompletely Specified Multi-Output Functions Based on Graph Coloring and Local Transformations and Its Application to FPGA Mapping, European Design Automation Conference, pages 23'. All data that was extracted from this citation was faulty errors, and there could not be found any relevant bibliographic data from sources on the web.

Part IV

Conclusions and Future Work

9 Conclusions

In this chapter we will review our research question and final goal and investigate if the goal has been reached. The research goal we defined was:

"Create an approach and a supporting software component to enrich BibTeX modelled references by integrating citation metadata with bibliographic data from sources on the web."

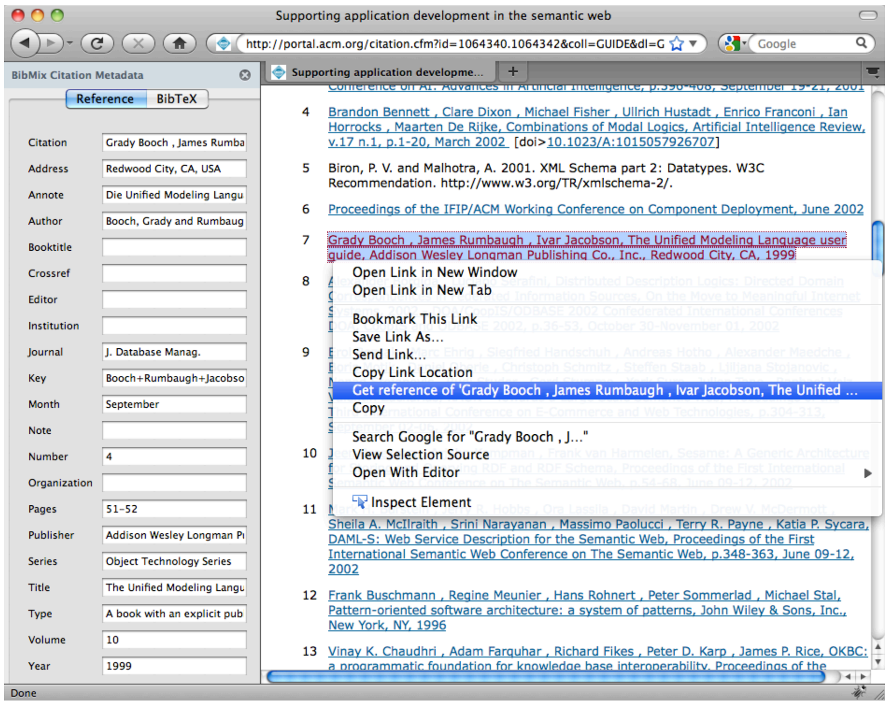
Citations contain valuable metadata that can act as a starting point for the integration process: citations often contain enough bibliographic information that enable the location and identification of relevant bibliographic data on the web. However, this metadata is first have to be extracted from the citation.

In this thesis document we propose an approach for the enrichment of citation metadata based on integration of bibliographic data. The approach incorporates a pipeline consisting of two phases: the Extraction and the Integration phase. The Extraction phase is responsible for both the extraction of metadata of the input citation and the transformation of the metadata into a BibTeX reference model. After the Extraction phase has transformed the extracted citation metadata into the reference model, the Integration phase starts a sequence of integration handlers. Each of the integration handlers handles the integration of bibliographic data from different data sources. An integration handler first validates if the input reference contains enough bibliographic information to locate and identify of relevant bibliographic data from data sources on the web. If the reference passes the validation, bibliographic data is collected from sources on the web and transformed into a set of references. The next step uses advanced filtering technique to make sure only relevant references are integrated during the integration step that follows. The integration procedure integrates the filtered references with the input reference. The end result of the sequence of integration handlers is an enriched reference, which consists of the citation metadata from the input citation integrated with relevant bibliographic data from sources on the web.

We have evaluated the performance of the Extraction phase. The average precision, recall and F1 of the Extraction phase are 93.6%, 85.6% and 88.1 respectively. Additionally, we have assessed the average quality of the enriched references of BibMix to be 0.72. This quality was expressed using a quality measure we have designed to be used for the evaluation. Furthermore, the quality measure was designed to be used in future projects to compare enrichment results to the results of this thesis project.

A public version of BibMix is available on `http://wisserver.st.ewi.tudelft.nl:3007/reference/create`. There the user can enter citation strings into a textbox and the enriched result is shown in a table.

In addition, we have developed an extension for the Mozilla Firefox webbrowser (the extension is shown in Figure 14). The extension is called the BibMix Reference Sidebar. This extension enables users to select a citation string on a web-page. This citation string is then sent to the BibMix webservice, which


Supporting application development in the semantic web
http://portal.acm.org/citation.cfm?id=1064340.1064342&coll=GUIDE&dl=G
Google
BibMix Citation Metadata
Supporting application developme...
Reference
BibTeX
Open Link in New Window
Open Link in New Tab
Bookmark This Link
Save Link As...
Send Link...
Copy Link Location
Get reference of 'Grady Booch , James Rumbaugh , Ivar Jacobson, The Unified ...
Copy
Search Google for "Grady Booch , J..."
View Selection Source
Open With Editor
Inspect Element
Done

Citation	Grady Booch , James Rumba
Address	Redwood City, CA, USA
Annote	Die Unified Modeling Langu
Author	Booch, Grady and Rumbaug
Booktitle	
Crossref	
Editor	
Institution	
Journal	J. Database Manag.
Key	Booch+Rumbaugh+Jacobso
Month	September
Note	
Number	4
Organization	
Pages	51-52
Publisher	Addison Wesley Longman Pi
Series	Object Technology Series
Title	The Unified Modeling Langu
Type	A book with an explicit pub
Volume	10
Year	1999

4 Brandon Bennett , Clare Dixon , Michael Fisher , Ullrich Hustadt , Enrico Franconi , Ian Horrocks , Maarten De Rijke, Combinations of Modal Logics, Artificial Intelligence Review, v.17 n.1, p.1-20, March 2002 [doi>10.1023/A:1015057926707]

5 Biron, P. V. and Malhotra, A. 2001. XML Schema part 2: Datatypes. W3C Recommendation. http://www.w3.org/TR/xmlschema-2/.

6 Proceedings of the IFIP/ACM Working Conference on Component Deployment, June 2002

7 Grady Booch , James Rumbaugh , Ivar Jacobson, The Unified Modeling Language user guide, Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, 1999

11 Sheila A. McIlraith , Srini Narayanan , Massimo Paolucci , Terry R. Payne , Katia P. Sycara, DAML-S: Web Service Description for the Semantic Web, Proceedings of the First International Semantic Web Conference on The Semantic Web, p.348-363, June 09-12, 2002

12 Frank Buschmann , Regine Meunier , Hans Rohnert , Peter Sommerlad , Michael Stal, Pattern-oriented software architecture: a system of patterns, John Wiley & Sons, Inc., New York, NY, 1996

13 Vinay K. Chaudhri , Adam Farquhar , Richard Fikes , Peter D. Karp , James P. Rice, OKBC: a programmatic foundation for knowledge base interoperability, Proceedings of the

Figure 14: A screenshot of the Firefox extension 'BibMix Reference Sidebar' in action.

starts enriching the citation metadata. After the enrichment, the result is sent back to the browser and the bibliographic data is shown in the browser's sidebar. Additionally, a generated BibTeX entry of the enriched reference is made available under the 'BibTeX' tab-page of the sidebar.

9.1 Research Questions

This thesis project addressed the research goal by answering the following research questions.

Question 1: What is the state-of-the-art of Citation Metadata Extraction approaches?

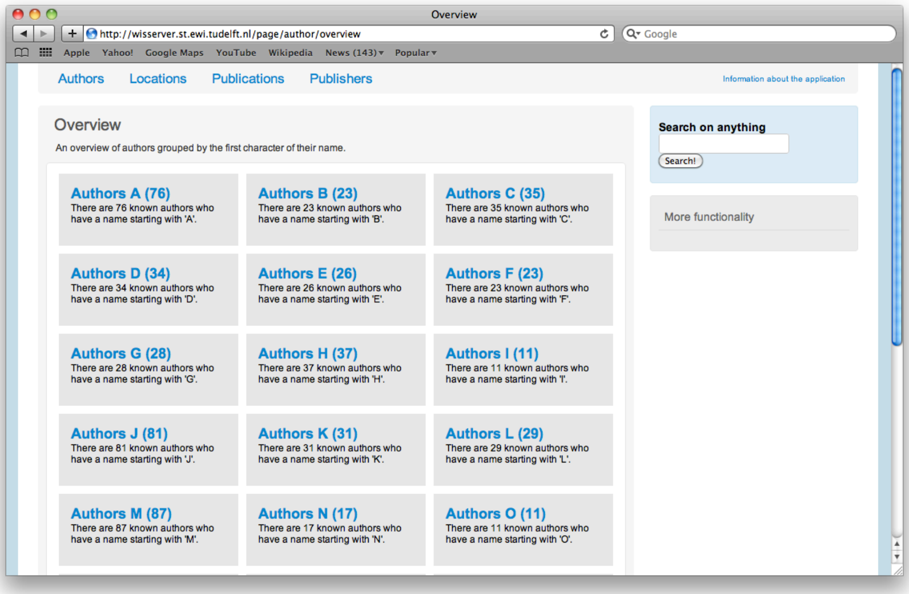
In Chapter 2 we introduced several state-of-the-art of CME approaches. Research shows that machine learning-based CME approaches outperform template-based (and rule-based) CME approaches in terms of performance and accuracy.

Question 2: Which steps can be taken in order to integrate citation metadata with bibliographic data from sources on the web?

In Part II we present an approach, a design and an implementation for citation metadata enrichment. The approach incorporates an Extraction and Integration phase to integrate relevant bibliographic data with citation metadata. The Extraction phase is responsible for the extraction of citation metadata. The Integration phase consists of a sequence of integration handlers. The handlers can be subdivided in four sub-steps for validation, collection, filtering and integration. The validation step ensures the input reference contains enough bibliographic information to locate and identify new relevant data in data sources on the web. The collection step collects relevant bibliographic data and transforms it into a set of references. A filtering step, based on the syntactic comparison of a selected number of attributes from each of the collected references and the input reference, is responsible for removing irrelevant references from the set. The reference integration step integrates the filtered references with the input reference. Each of the integration handlers in the sequence passes along the enriched reference until no handlers reside. The enriched reference of the last integration handler is the output of the approach.

Question 3: How does the approach from 2 perform?

In Chapter 7 we describe the evaluation procedure we have performed to evaluate BibMix, the implementation of the citation metadata enrichment approach we have proposed in this thesis document. The result and the analysis of the evaluation are presented in Chapter 8. We have compared the results of the Extraction phase to those provided by a human expert. The average precision, recall and F1 of the Extraction phase are 93.6%, 85.6% and 88.1 respectively. Additionally, we have assessed the average quality of the enriched references of BibMix to be 0.72. This quality was expressed using a quality measure we have designed to be used for the evaluation (see Section 6.2). Furthermore, the quality measure was designed to be used in future projects to compare enrichment results to the results of this thesis project.


Overview
http://wisserver.st.ewi.tudelft.nl/page/author/overview
Google
Apple Yahoo! Google Maps YouTube Wikipedia News (143) Popular
Authors Locations Publications Publishers
Information about the application
Search on anything
Search!
More functionality

Overview

An overview of authors grouped by the first character of their name.

Authors A (76) There are 76 known authors who have a name starting with 'A'.

Authors B (23) There are 23 known authors who have a name starting with 'B'.

Authors C (35) There are 35 known authors who have a name starting with 'C'.

Authors D (34) There are 34 known authors who have a name starting with 'D'.

Authors E (26) There are 26 known authors who have a name starting with 'E'.

Authors F (23) There are 23 known authors who have a name starting with 'F'.

Authors G (28) There are 28 known authors who have a name starting with 'G'.

Authors H (37) There are 37 known authors who have a name starting with 'H'.

Authors I (11) There are 11 known authors who have a name starting with 'I'.

Authors J (81) There are 81 known authors who have a name starting with 'J'.

Authors K (31) There are 31 known authors who have a name starting with 'K'.

Authors L (29) There are 29 known authors who have a name starting with 'L'.

Authors M (87) There are 87 known authors who have a name starting with 'M'.

Authors N (17) There are 17 known authors who have a name starting with 'N'.

Authors O (11) There are 11 known authors who have a name starting with 'O'.

Figure 15: The Linked Data Application showcasing the added value of enrichment of bibliographic data.

9.2 Contributions

In this section we describe two other contributions besides the approach and the implementation we have developed during the thesis project.

9.2.1 Showcase Application

During the thesis project we have worked together with Bas Schoenmakers on an application[26] showcasing the added value of enrichment of bibliographic data. It was our task to enrich a set of citations from members of the Web Information Systems (WIS) Research group at the Delft University of Technology. After the enrichment Schoenmakers was able convert all the relevant data to RDF. He was able to interlink the resulting entities to entities in the Linked Data cloud and stored all the data in an RDF triple-store[27].

After the enrichment we were able to interlink the bibliographic data with three different datasets (DBLP[28], DBPedia[29] and GeoNames[30]). With the new

[26]STPublications Linked Data Application, `http://wisserver.st.ewi.tudelft.nl/`, last visited on August 30, 2010

[27]The store can be found at: `http://wisserver.st.ewi.tudelft.nl:8893/sparql`, last visited on August 30, 2010

[28]DBLP, `http://dblp.l3s.de/d2r/`, last visited on August 26, 2010

[29]DBPedia, `http://dbpedia.org/`, last visited on August 26, 2010

[30]GeoNames[31] `http://geonames.org/`, last visited on August 26, 2010

links we could add several features that were not possible before the enrichment. A few examples of the new features are:

- Aggregate references by continent
- A world map for every author to see in which countries her or she published something
- Show the top co-author relationships in a graph
- Show the topics an author wrote about

My contributions for this showcase application were the enrichment of citation metadata of a set of citations of members of the WIS Group. Furthermore, we have developed the showcase application base system and navigation structure, implemented search functionality, added content negotiation for semantic web-browsers. Finally, we have also provided the data that was needed in order to show the co-author relationships and to show the topics an author wrote about.

For overview of the application, the features we were able to create after enrichment and the interlinking process I refer to the thesis document of Bas Schoenmakers (Schoenmakers, 2010).

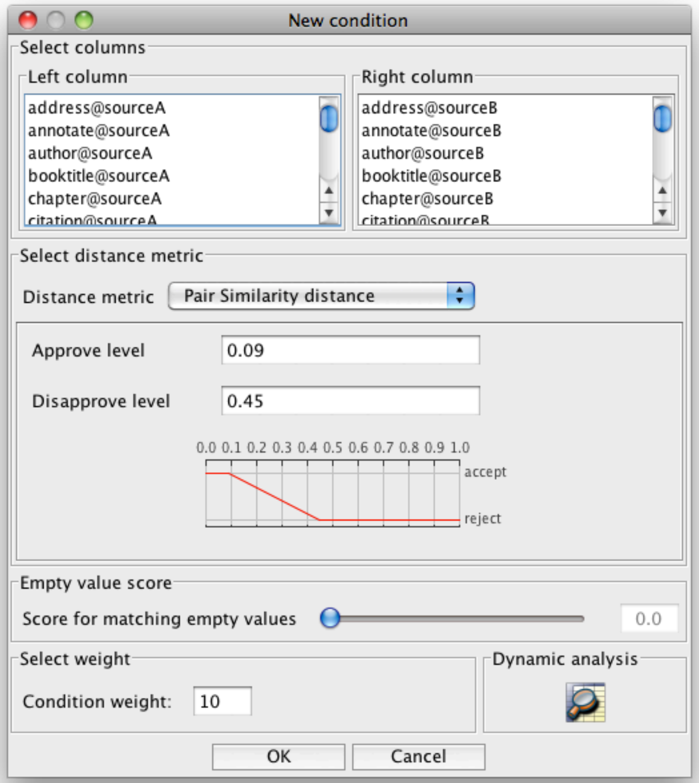
9.2.2 FRIL: Fine-grained Records Integration and Linkage tool

The filtering of references in BibMix is done by a Record Linkage tool called FRIL (see Section 4.3.4 for more information about the filtering procedure). FRIL provides a rich set of tools for comparing records, in our case, references. Furthermore, FRIL provides a graphical user interface for configuring the comparison of references. At design time, developers may systematically and iteratively explore the optimal combination of parameter values to enhance comparison performance and accuracy. BibMix uses the command-line interface of FRIL, however, the interface was broken. FRIL is open-source software, so we had access to the source code and we were able to fix the errors that kept BibMix from using the command line interface.

In addition, the string similarity measures that come with FRIL were not sufficient for the comparison of author names and titles of references. Observations showed that the PairDistance algorithm[32] performs very well, even when comparing full and abbreviated author names. Therefore, we have implemented the PairDistance for FRIL, including a graphical user interface to explore the optimal configuration (see Figure 16).

We have contributed the string similarity implementation and the bugfixes back to the developers of FRIL. This is, in our oppinion, a valueable addition to the FRIL codebase. First, because FRIL does not have many String Similarity metrics available for comparison. And second, because the commandline interface was not working and therefore FRIL could hardly be incorporated by other

[32]PairDistance algorithm: `http://www.catalysoft.com/articles/StrikeAMatch.html`, last visit on July 5, 2010



Left column: address@sourceA, annotate@sourceA, author@sourceA, booktitle@sourceA, chapter@sourceA, citation@sourceA

Right column: address@sourceB, annotate@sourceB, author@sourceB, booktitle@sourceB, chapter@sourceB, citation@sourceB

Distance metric: Pair Similarity distance

Approve level: 0.09

Disapprove level: 0.45

Score for matching empty values: 0.0

Condition weight: 10

Figure 16: The graphical user interface where Pair Similarity algorithm in FRIL can be configured.

tools like BibMix. Second, becaThe lead developer, Pawel Jurczyk, contacted me that the additions will be part of the next release of FRIL.

10 Future Work

Following the approach presented in this thesis, a number of projects could be taken up. In this section we describe a case that could be useful for further research.

Dynamic Arrangement of the Integration Phase and Incorporation of Linked Data BibMix is constructed for the Computer Science domain: it targets two data sources that consist of computer science publications for the greater part. The Integration phase is implemented as a static sequence of integration handlers in this prototype. It would be interesting to see if the quality of the enriched references becomes higher when the Integration phase would be arranged dynamically by targeting different data sources based on the semantics of references.

There are endless possibilities for incorporating Semantic Web data in the enrichment process. The Semantic Web could be used not just for the collection of relevant bibliographic data, but it might also play a role during the filtering process. For example, FRIL provides an excellent means for the comparison of references in the filtering step. However, the comparison of references is based on the syntactic characteristics of the compared attributes. It would be interesting to see if the references could also be compared based on semantics.

Bibliography

Baird, L. M. and C. Oppenheim (1994). Do citations matter? *J. Inf. Sci. 20*(1), 2–15.

Besagni, D., A. Belaïd, and N. Benet (2003). A segmentation method for bibliographic references by contextual tagging of fields. In *ICDAR '03: Proceedings of the Seventh International Conference on Document Analysis and Recognition*, Washington, DC, USA, pp. 384. IEEE Computer Society.

Bollacker, K. D., S. Lawrence, and C. L. Giles (1998). Citeseer: an autonous web agent for automatic retrieval and identification of interesting publications. In *AGENTS '98: Proceedings of the second international conference on Autonomous agents*, New York, NY, USA, pp. 116–123. ACM.

Canós, J. H., M. Llavador, E. Mena, and M. R. Borges (2008). A service-oriented infrastructure for early citation management. In *ECDL '08: Proceedings of the 12th European conference on Research and Advanced Technology for Digital Libraries*, Berlin, Heidelberg, pp. 160–171. Springer-Verlag.

Chen, C.-C., K.-H. Yang, H.-Y. Kao, and J.-M. Ho (2008). Bibpro: A citation parser based on sequence alignment techniques. In *AINAW '08: Proceedings of the 22nd International Conference on Advanced Information Networking and Applications - Workshops*, Washington, DC, USA, pp. 1175–1180. IEEE Computer Society.

Connan, J. and C. W. Omlin (2000). Bibliography extraction with hidden markov models.

Cortez, E., A. S. da Silva, M. A. Gonçalves, F. Mesquita, and E. S. de Moura (2007). Flux-cim: flexible unsupervised extraction of citation metadata. In *JCDL '07: Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries*, New York, NY, USA, pp. 215–224. ACM.

Councill, C. L. G. I. and M.-Y. Kan (2008, May). Parscit: an open-source crf reference string parsing package. In *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2008/.

Day, M.-Y., R. T.-H. Tsai, C.-L. Sung, C.-C. Hsieh, C.-W. Lee, S.-H. Wu, K.-P. Wu, C.-S. Ong, and W.-L. Hsu (2007). Reference metadata extraction using a hierarchical knowledge representation framework. *Decis. Support Syst. 43*(1), 152–167.

Gravano, L., P. G. Ipeirotis, H. V. Jagadish, N. Koudas, S. Muthukrishnan, L. Pietarinen, and D. Srivastava (2007). Using q-grams in a dbms for approximate string processing.

Han, H., C. L. Giles, E. Manavoglu, H. Zha, Z. Zhang, and E. A. Fox (2003). Automatic document metadata extraction using support vector machines. In *JCDL '03: Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries*, Washington, DC, USA, pp. 37–48. IEEE Computer Society.

Hetzner, E. (2008). A simple method for citation metadata extraction using hidden markov models. In *JCDL '08: Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries*, New York, NY, USA, pp. 280–284. ACM.

Huang, I.-A., J.-M. Ho, H.-Y. Kao, and W.-C. Lin (2004). Extracting citation metadata from online publication lists using BLAST. In *Advances in Knowledge Discovery and Data Mining*, Volume 3056 of *Lecture Notes in Computer Science*, pp. 539–548. Springer Berlin / Heidelberg.

Kramer, M., H. Kaprykowsky, D. Keysers, and T. Breuel (2007). Bibliographic meta-data extraction using probabilistic finite state transducers. In *ICDAR '07: Proceedings of the Ninth International Conference on Document Analysis and Recognition*, Washington, DC, USA, pp. 609–613. IEEE Computer Society.

Lafferty, J. D., A. McCallum, and F. C. N. Pereira (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, San Francisco, CA, USA, pp. 282–289. Morgan Kaufmann Publishers Inc.

Lamport, L. (1986). *LATEX. A document preparation system. User's Guide and Reference Manual.*

Lee, D., J. Kang, P. Mitra, C. L. Giles, and B.-W. On (2007). Are your citations clean? *Commun. ACM 50*(12), 33–38.

Ley, M. (2002). The dblp computer science bibliography: Evolution, research issues, perspectives. In *SPIRE 2002: Proceedings of the 9th International Symposium on String Processing and Information Retrieval*, London, UK, pp. 1–10. Springer-Verlag.

Meyer, B. (1986). *Design by Contract.* Interactive Software Engineering Inc.

Needleman, S. B. and C. D. Wunsch (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology 48*(3), 443 – 453.

Patashnik, O. (1988). Bibtexing. In *Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE 77*, pp. 257–286.

Rabiner, L. (1989, feb). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE 77*(2), 257 –286.

Schoenmakers, B. (2010). Designing linked data applications. Thesis document.

Seymore, K., A. Mccallum, and R. Rosenfeld (1999). Learning hidden markov model structure for information extraction. In *In AAAI 99 Workshop on Machine Learning for Information Extraction*, pp. 37–42.

Takasu, A. (2003). Bibliographic attribute extraction from erroneous references based on a statistical model. *Digital Libraries, Joint Conference on 0*, 49.

Wei, W., I. King, and J. H.-M. Lee (2007). Bibliographic attributes extraction with layer-upon-layer tagging. In *ICDAR '07: Proceedings of the Ninth International Conference on Document Analysis and Recognition*, Washington, DC, USA, pp. 804–808. IEEE Computer Society.

Yin, P., M. Zhang, Z. Deng, and D. Yang (2005). Metadata extraction from bibliographies using bigram HMM. In *Digital Libraries: International Collaboration and Cross-Fertilization*, Volume 3334 of *Lecture Notes in Computer Science*, pp. 310–319. Springer Berlin / Heidelberg.

Part V
Appendix

A BibTeX Record Types

Each BibTeX record starts with an '@' character following by the type of the record. A number of record types have been specified in the BibTeX specification (Patashnik, 1988). Below is a list of record types also mentioned in the specification.

article An article from a journal or magazine. Required fields: `author`, `title`, `journal`, `year`. Optional fields: `volume`, `number`, `pages`, `month`, `note`.

book A book with an explicit publisher. Required fields: `author` or `editor`, `title`, `publisher`, `year`. Optional fields: `volume` or `number`, `series`, `address`, `edition`, `month`, `note`.

booklet A work that is printed and bound, but without a named publisher or sponsoring institution. Required field: `title`. Optional fields: `author`, `howpublished`, `address`, `month`, `year`, `note`.

conference The same as `INPROCEEDINGS`, included for *Scribe* compatibility.

inbook A part of a book, which may be a chapter (or section or whatever) and/or a range of pages. Required fields: `author` or `editor`, `title`, `chapter` and/or `pages`, `publisher`, `year`. Optional fields: `volume` or `number`, `series`, `type`, `address`, `edition`, `month`, `note`.

incollection A part of a book having its own title. Required fields: `author`, `title`, `booktitle`, `publisher`, `year`. Optional fields: `editor`, `volume` or `number`, `series`, `type`, `chapter`, `pages`, `address`, `edition`, `month`, `note`.

inproceedings An article in a conference proceedings. Required fields: `author`, `title`, `booktitle`, `year`. Optional fields: `editor`, `volume` or `number`, `series`, `pages`, `address`, `month`, `organization`, `publisher`, `note`.

manual Technical documentation. Required field: `title`. Optional fields: `author`, `organization`, `address`, `edition`, `month`, `year`, `note`.

mastersthesis A Master's thesis. Required fields: `author`, `title`, `school`, `year`. Optional fields: `type`, `address`, `month`, `note`.

misc Use this type when nothing else fits. Required fields: none. Optional fields: `author`, `title`, `howpublished`, `month`, `year`, `note`.

phdthesis A PhD thesis. Required fields: `author`, `title`, `school`, `year`. Optional fields: `type`, `address`, `month`, `note`.

proceedings The proceedings of a conference. Required fields: `title`, `year`. Optional fields: `editor`, `volume` or `number`, `series`, `address`, `month`, `organization`, `publisher`, `note`.

techreport A report published by a school or other institution, usually numbered within a series. Required fields: `author`, `title`, `institution`, `year`. Optional fields: `type`, `number`, `address`, `month`, `note`.

unpublished A document having an author and title, but not formally published. Required fields: `author`, `title`, `note`. Optional fields: `month`, `year`.

B BibTeX Record Fields

Each record consists of a set of fields. There is a set of fields described in the BibTeX specification, which can be interpreted by BibTeX or third-party tools. Those which are unknown are ignored by BibTeX and thus can be used to store additional information without interfering with the final outcome of a document after the compilation of a BibTeX database. Below are the fields which are mentioned in the specification.

address Usually the address of the `publisher` or other type of institution. For major publishing houses, van Leunen recommends omitting the information entirely. For small publishers, on the other hand, you can help the reader by giving the complete address.

annote An annotation. It is not used by the standard bibliography styles, but may be used by others that produce an annotated bibliography.

author The name(s) of the author(s), in the format described in the LaTeX book.

booktitle Title of a book, part of which is being cited. See the LaTeX book for how to type titles. For book entries, use the `title` field instead.

chapter A chapter (or section or whatever) number.

crossref The database key of the entry being cross referenced.

edition The edition of a book—for example, "Second". This should be an ordinal, and should have the first letter capitalized, as shown here; the standard styles convert to lower case when necessary.

editor Name(s) of editor(s), typed as indicated in the LaTeX book. If there is also an `author` field, then the `editor` field gives the editor of the book or collection in which the reference appears.

howpublished How something strange has been published. The first word should be capitalized.

institution The sponsoring institution of a technical report.

journal A journal name. Abbreviations are provided for many journals; see the *Local Guide*.

key Used for alphabetizing, cross referencing, and creating a label when the "author" information is missing. This field should not be confused with the key that appears in the `\cite` command and at the beginning of the database entry.

month The month in which the work was published or, for an unpublished work, in which it was written. You should use the standard three-letter abbreviation, as described in Appendix B.1.3 of the LaTeX book.

note Any additional information that can help the reader. The first word should be capitalized.

number The number of a journal, magazine, technical report, or of a work in a series. An issue of a journal or magazine is usually identified by its volume and number; the organization that issues a technical report usually gives it a number; and sometimes books are given numbers in a named series.

organization The organization that sponsors a conference or that publishes a manual.

pages One or more page numbers or range of numbers, such as `42-111` or `7,41,73-97` or `43+` (the '+' in this last example indicates pages following that don't form a simple range). To make it easier to maintain *Scribe*-compatible databases, the standard styles convert a single dash (as in `7-33`) to the double dash used in TeX to denote number ranges (as in `7-33`).

publisher The publisher's name.

school The name of the school where a thesis was written.

series The name of a series or set of books. When citing an entire book, the the `title` field gives its title and an optional `series` field gives the name of a series or multi-volume set in which the book is published.

title The work's title, typed as explained in the LaTeX book.

type The type of a technical report—for example, "Research Note".

volume The volume of a journal or multivolume book.

year The year of publication or, for an unpublished work, the year it was written. Generally it should consist of four numerals, such as `1984`, although the standard styles can handle any `year` whose last four nonpunctuation characters are numerals, such as '(about 1984)'.