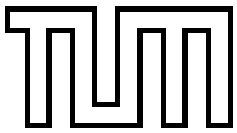
Technische Universität München
Institut für Informatik



Supporting the Exchange of Knowledge in Communities of Interest via Document Catalog Mediation

Martin Lacher

Institut für Informatik
der Technischen Universität München

Supporting the Exchange of Knowledge in Communities of Interest via Document Catalog Mediation

Martin Lacher

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzende:		Univ.-Prof. Anja Feldmann, Ph.D.
Prüfer der Dissertation:	1.	Univ.-Prof. Dr. Johann Schlichter
	2.	Univ.-Prof. Dr. Uwe M. Borghoff Universität der Bundeswehr München

Die Dissertation wurde am 18.6.2003 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 10.10.2003 angenommen.

Abstract

Loosely coupled interest groups, so-called *communities of interest*, are valuable sources of knowledge, which have become accessible to large audiences via the Internet. The exchange of documents via document catalogs is an important means of knowledge exchange among community members. The individual community members often organize personal document collections in personal catalogs, such as for example bookmarks. So far, the additional effort that community members have to invest for knowledge exchange, for example via a community catalog, strongly impedes knowledge exchange in communities.

Our first goal in this work is to design an application framework for supporting knowledge exchange via document catalogs in communities. We analyze requirements for the framework, based on psychological and sociological characteristics of the knowledge exchange process in communities. We introduce the CAIMAN[1] framework, which consists of knowledge exchange support services that fulfill the above requirements and a mediation infrastructure with which the services can be realized. CAIMAN allows a community member to use one personal document catalog for all knowledge exchanges, independent the catalog choice of the exchange partner. The exchange among the different catalogs is delegated to the CAIMAN services and mediation infrastructure.

Our second goal is to provide and evaluate a concept for a document catalog mediation infrastructure. Our mediation approach provides for semi-automatic virtual integration of document catalogs. Conceptual catalog differences are resolved by a catalog matching component and issues of querying heterogeneous document catalogs are taken care of by a catalog query infrastructure.

We introduce a novel catalog matching approach, which automatically matches the categories of two catalogs. Our matching approach uses the documents and the structure of a catalog for the matching calculations, whereas the highly subjective category names are not used. The CAIMAN matching approach is based on text classification and graph matching techniques.

Our querying approach allows to jointly query heterogeneous document catalogs that are represented with different data models, using the same query infrastructure. The approach requires the conversion of all catalog data to *RDF*[2] as lingua franca for catalog querying. As a proof of concept for our querying approach, we show how *Topic Maps* based document catalogs can be jointly queried with *RDF* based document catalogs. *RDF* and *Topic Maps* are the most popular catalog representation data models on the Web today.

We have implemented a catalog mediation prototype. Experimental results show that our approach provides for high-quality mediation, which is the basis for successful knowledge exchange.

[1] **C**ommunity knowledge exch**A**nge v**I**a **M**edi**A**tio**N** of document catalogs.

[2] Resource Description Format

Acknowledgments

First, I would like to thank my advisor, Prof. Dr. Johann Schlichter, who has given me the freedom and opportunity to work on an interesting topic and who has greatly contributed to this work with his detailed comments and guidance.

I would also like to thank Prof. Dr. Uwe Borghoff, who has agreed to be my co-advisor and whose valuable comments have inspired a major part of this work.

Moreover, I thank my colleagues at Technische Universität München as well as Stanford University, who have had a great influence on this work with their valuable comments and their own research work.

Furthermore I would like to thank my parents for their support and encouragement during difficult times. And last but not least, I would like to thank Gerlinde Gall for her great support and understanding. She helped me a lot to improve this work.

Contents

List of Figures xiii
List of Tables xv
Table of Acronyms xvii
Table of Symbols xix

I Introduction **1**

1 Introduction **3**
1.1 Document catalogs for knowledge exchange in communities 3
1.2 Problem description 4
1.3 Goals for this work 6
1.4 Related work 7
1.5 Structure and results of this work 8

II Application Framework **13**

2 Application Scenario **15**
2.1 Introduction 15
2.2 The notion of community 16
2.2.1 Historic evolution of the notion of community 16
2.2.2 Definition of community for this work 17
2.2.3 Categorization of communities 18
2.2.4 Community characteristics 19
2.2.5 Community environments 20
2.2.6 Teams and communities 21
2.2.7 Example communities and community support applications 22
2.3 Knowledge exchange in communities of interest 24
2.3.1 Characterization of knowledge 24
2.3.2 Knowledge exchange model 26
2.3.3 Supporting knowledge exchange with document catalogs 27
2.3.4 Knowledge exchange partners 32
2.3.5 Knowledge flow in communities of interest 32
2.4 Summary 34

3 Community Knowledge Exchange Support Requirements 37
3.1 Introduction . . . 37
3.2 Requirements for supporting the knowledge exchange process . . . 38
3.2.1 Knowledge Management process blocks . . . 38
3.2.2 Application of the process blocks to the knowledge exchange process . . . 40
3.2.3 Psychological aspects of knowledge creation . . . 42
3.3 General Requirements for community support applications . . . 44
3.3.1 Experiences from groupware design and deployment . . . 44
3.3.2 Special communityware requirements . . . 46
3.4 Requirements for knowledge exchange via document catalogs . . . 46
3.4.1 Comparison of different approaches to using catalogs for knowledge exchange 47
3.4.2 Requirements for knowledge exchange using all catalogs . . . 53
3.5 Summary . . . 53

4 Application Level Design of the CAIMAN Framework 57
4.1 Introduction . . . 57
4.2 CAIMAN knowledge exchange principle . . . 58
4.3 Knowledge exchange support services . . . 59
4.3.1 Information publication . . . 60
4.3.2 Related information retrieval . . . 61
4.3.3 Category discovery . . . 62
4.3.4 Semi-automatic mediation . . . 64
4.3.5 Application requirements for the mediation infrastructure . . . 64
4.4 An example community support application with CAIMAN services . . . 65
4.5 Related work . . . 66
4.6 Summary . . . 68

III Mediation Infrastructure 71

5 CAIMAN Document Catalog Mediation Principle 73
5.1 Introduction . . . 73
5.1.1 Specification of the mediation problem . . . 73
5.1.2 Application requirements . . . 74
5.1.3 Investigated problems . . . 74
5.2 Document catalog mediation principle . . . 75
5.3 Comparison of mediation approaches . . . 77
5.3.1 Document granular mediation . . . 77
5.3.2 Category granular mediation . . . 81
5.3.3 Summary and discussion . . . 83
5.4 Overview of the CAIMAN mediation approach . . . 85
5.4.1 Preparatory mapping phase . . . 86
5.4.2 Integration phase . . . 86
5.5 Summary . . . 87

6 CAIMAN Document Catalog Matching Approach 89
6.1 Introduction . . . 89
6.1.1 Specification of the catalog matching problem . . . 89
6.1.2 Optimal catalog characteristics . . . 91
6.2 Existing matching techniques . . . 92
6.2.1 Ontology matching . . . 92
6.2.2 Database schema matching . . . 93
6.3 Text classification techniques used in CAIMAN . . . 93
6.3.1 Document corpus indexing and feature weighting . . . 96
6.3.2 Feature vector dimensionality reduction . . . 98
6.3.3 Construction of text classifiers . . . 100
6.3.4 Multi-category classification . . . 104
6.3.5 Information retrieval performance measures . . . 104
6.4 Overview of the CAIMAN catalog matching approach . . . 106
6.5 Document categorization phase . . . 108
6.5.1 Catalog indexing . . . 108
6.5.2 User catalog training . . . 109
6.5.3 Community document categorization . . . 109
6.6 Category correlation phase . . . 111
6.7 Structural matching phase . . . 112
6.7.1 Similarity estimate improvement through similarity inheritance . . . 113
6.7.2 Similarity estimate improvement through Similarity Flooding . . . 114
6.7.3 Match candidate filtering . . . 116
6.8 Summary . . . 119

7 Querying Heterogeneous Document Catalogs with CAIMAN 121
7.1 Introduction . . . 121
7.2 Theoretical background . . . 123
7.2.1 RDF and related W3C standards . . . 124
7.2.2 Topic Maps . . . 124
7.3 General approach for querying heterogeneous document catalogs . . . 125
7.3.1 RDF as lingua franca for catalog data . . . 125
7.3.2 Modelling heterogeneous catalog representations with RDF . . . 125
7.3.3 Joint querying of document catalogs using the TRIPLE query language . . . 127
7.3.4 Realization of the query component . . . 128
7.4 Related work . . . 129
7.5 Joint querying of Topic Maps and RDF catalogs . . . 129
7.5.1 Syntax layer . . . 130
7.5.2 Object layer . . . 130
7.5.3 Semantic layer . . . 132
7.6 Query example for RDF and Topic Maps . . . 132
7.6.1 Graph models of the data sources . . . 133
7.6.2 Mapping the Topic Map source graph to RDF . . . 135
7.6.3 Joint querying of the DMOZ and the Factbook catalog . . . 136
7.7 Summary . . . 137

8 CAIMAN Knowledge Exchange Service Performance Evaluation 139
8.1 Introduction . . . 139
8.2 Experimental setup . . . 139
8.2.1 Automatic evaluation . . . 140
8.2.2 Performance measure . . . 141
8.2.3 Experiments . . . 141
8.3 Implementation . . . 144
8.4 Performance evaluation . . . 146
8.4.1 Reuters 21578 catalog . . . 147
8.4.2 Researchindex catalog . . . 151
8.5 Summary . . . 154

IV Conclusion 157

9 Conclusion 159
9.1 Summary of contributions . . . 159
9.2 Future work . . . 164
9.2.1 Application framework . . . 164
9.2.2 Mediation infrastructure . . . 164
9.3 Concluding remarks . . . 166

V Appendix 167

A CAIMAN Software Architecture Draft 169

B CAIMAN Implementation 175
B.1 Usage of the prototype . . . 175
B.2 Static Structure . . . 178
B.3 Control flow . . . 179
B.4 Functionality overview . . . 180

C Query Mapping Rules 183

D Sample Document Catalog Data 185

List of Figures

1.1 Different categorization schemes aggravate the knowledge flow via document catalogs 5
1.2 Structure of this work 10

2.1 Overview of the application scenario 15
2.2 Community portfolio derived from [Carotenuto et al. 1999] 18
2.3 Communities and teams (see also [Borghoff et al. 2001]) 21
2.4 Relation of organization, team and community 22
2.5 The relationship of data, information and knowledge 25
2.6 The differences between tacit and explicit knowledge 25
2.7 The transformation processes between tacit and explicit knowledge 26
2.8 The model of knowledge exchange in this work 27
2.9 Example document catalog with several perspectives 30
2.10 Knowledge exchange partners 33

3.1 Knowledge Management problem classes / process blocks in [Probst et al. 1997] . . 39
3.2 Steps of the knowledge exchange process with consecutive knowledge application . . 41
3.3 Information management model in [Heinen and Dietel 1991] 43
3.4 One centralized community catalog - one centralized perspective on the domain . . . 47
3.5 Using personal catalogs only for knowledge exchange 50
3.6 Using both personal and community / global catalogs to support knowledge exchange 51

4.1 Conceptual overview of the CAIMAN framework 58
4.2 User interface for the personal catalog (design study) 59
4.3 Conceptual overview of the information publication service 60
4.4 User interface of the information publication service (design study) 61
4.5 Conceptual overview of the related information retrieval service 61
4.6 Conceptual overview of the category discovery Service 63
4.7 User interface of the category discovery service (design study) 63
4.8 The CommunityItemsTool [Koch et al. 2001] with CAIMAN retrieval service 65

5.1 Mediation scenario for heterogeneous document catalog sources 76
5.2 Subjective relations among documents and mediation result with document granular vs. category granular matching 79
5.3 Mediation quality depending on the catalog used for mediation 85

6.1 Catalog matching process in CAIMAN. 90

6.2 Support Vector Machine learning: finding the best hyperplane (adapted from [Sebastiani 2002]). . . . 103
6.3 Phases of the CAIMAN catalog matching approach . . . 107
6.4 Flattening of hierarchical catalogs for classification . . . 109
6.5 Ambiguous formal meaning of category relations with same label . . . 112
6.6 Intuition behind the simple similarity inheritance algorithm . . . 113
6.7 Intuition behind the Similarity Flooding algorithm presented in [Melnik et al. 2002] . 114
6.8 Similarity Flooding example pass (adapted from [Melnik et al. 2002]) . . . 115
6.9 Filter cardinalities for the publication service . . . 117
6.10 Filter cardinalities for the related information retrieval service . . . 118
6.11 Filter cardinalities for the category discovery service . . . 118
6.12 Filter cardinalities for the stable marriage filter . . . 118

7.1 Problem scenario: Querying heterogeneous catalogs . . . 122
7.2 Conceptual overview of the layers of a future *Semantic Web* [Berners-Lee 2000] . . . 123
7.3 Example statement in the RDF data model . . . 124
7.4 The layered interoperability model [Melnik and Decker 2000]. . . . 126
7.5 Conceptual overview of joint querying of different catalogs . . . 128
7.6 The RDF schema for an RDF-based Topic Map . . . 130
7.7 Example association represented in a Topic Map graph. . . . 131
7.8 Exemplary mapping of a Topic Map node to an RDF graph . . . 132
7.9 Exemplary mapping of a Topic Map associationMember edge to an RDF graph . . . 132
7.10 RDF graph part from the Open Directory Catalog . . . 134
7.11 Topic Map graph part from the CIA World Factbook . . . 135
7.12 The generated RDF Topic Map graph. . . . 135

8.1 Generation of the user and community catalog for automatic evaluation of the *related information retrieval* service . . . 140
8.2 Overview of the data flow in the mediation component prototype implementation . . 145
8.3 *Related retrieval* service performance on Reuters catalog with modified modApté split, Naive Bayes classifier with distribution determined category priors, macro-/micro-averaged F_1-Measure: D vs. C1 . . . 148
8.4 *Related retrieval* service user catalog scalability on Reuters catalog with modified modApté split, Naive Bayes classifier with uniform category priors, macro-/micro-averaged F_1-Measure: D vs. C1 . . . 148
8.5 *Related retrieval* service performance on Reuters catalog with modified modApté split, SVM classifier, macro-/micro-averaged F_1-Measure: D vs. C1 . . . 149
8.6 *Related retrieval* service performance on Reuters catalog with modified modApté split, context aware SVM classifier, macro-/micro-averaged F_1-Measure: D vs. C1 . 150
8.7 *Related retrieval* service user catalog scalability on Reuters catalog with modified modApté split, SVM classifier, macro-/micro-averaged F_1-Measure: D vs. C1 . . . 150
8.8 *Related retrieval* service performance on Researchindex catalog with 2-fold cross-validation, Naive Bayes classifier, macro-/micro-averaged F_1-Measure: D vs. C1 vs. C2 vs. C3 vs. C3* . . . 151
8.9 *Related retrieval* service user catalog scalability on Researchindex catalog with 2-fold cross validation, Naive Bayes classifier, macro-/micro-averaged F_1-Measure: D vs. C1 vs. C2 vs. C3 vs. C3* . . . 152

8.10 *Related retrieval* service performance on Researchindex catalog with 2-fold cross validation, SVM classifier, macro-/micro-averaged F_1-Measure: D vs. C1 vs. C2 vs. C3 vs. C3* 152
8.11 *Related retrieval* service performance on Researchindex catalog with 2-fold cross validation, context aware SVM classifier, macro-/micro-averaged F_1-Measure: D vs. C1 vs. C2 vs. C3 vs. C3* 153
8.12 *Related retrieval* service user catalog scalability on Researchindex catalog with 2-fold cross validation, SVM classifier, macro-/micro-averaged F_1-Measure: D vs. C1 vs. C2 vs. C3 vs. C3* 153

A.1 CAIMAN Agent Architecture Concept 170

B.1 CAIMAN mediation prototype main classes 178
B.2 CAIMAN mediation prototype main control flow 179
B.3 CAIMAN mediation prototype subgraph performance calculation control flow 180

C.1 TRIPLE model that allows query mapping for the Factbook catalog. 184
C.2 Part of the TRIPLE model that allows query mapping for the DMOZ catalog. 184

D.1 Open Directory Catalog Contents in RDF 185
D.2 Open Directory Catalog Structure in RDF 186
D.3 CIA World Factbook XTM source 187

List of Tables

4.1 Related Work compared to CAIMAN 68
4.2 Assignment of support services to process blocks and knowledge types 69

8.1 Overview of the catalogs used for evaluation 143

B.1 Overview of the package *de.tum.caiman* 180
B.2 Overview of the package *de.tum.caiman.classify.** 181
B.3 Overview of the package *de.tum.caiman.crawl.** 181
B.4 Overview of the package *de.tum.caiman.experiments* 182
B.5 Overview of the package *de.tum.caiman.graph* 182
B.6 Overview of the package *de.tum.caiman.match* 182

Table of Acronyms

CAIMAN	**C**ommunity knowledge exch**A**nge v**I**a **M**edi**A**tio**N** of document catalogs
COI	Community of interest
COSS	Community support system
IR	Information Retrieval
KM	Knowledge Management
KR	Knowledge Representation
RDF	Resource Description Format
SGML	Standard Generalized Markup Language
SVM	Support Vector Machine
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
XML	eXtensible Markup Language

Table of Symbols

α	Weight of category vector for context aware categorization
α_i	Weight of inherited ancestor similarity value for tree inheritance algorithm
β_F	Relative importance of recall and precision in the F_{β_F} Categorization performance measure
β_R	Weight of the positive training examples for the Rocchio classifier
c_{SVM}	Cost parameter for C-SVM classifier
c_i	Category i from the set of categories: $c_i \in \mathcal{C}$
$\mathcal{C}$	Set of categories in a catalog
C	Random variable for category chosen in categorization
Γ	Document catalog
γ_R	Weight of the negative training examples for the Rocchio classifier
d_{SVM}	SVM Kernel degree
d_j	Document from the domain of documents: $d_j \in \mathcal{D}$
$\overrightarrow{d_j}$	Document feature vector
D	Random variable for document chosen from the set of documents
$\mathcal{D}$	Set of documents
$\mathcal{D}_p$	Pre-classified set of documents (*corpus*)
$\mathcal{D}_{tr}$	Training- and validation set of documents
$\mathcal{D}'_{tr}$	Training set of documents for parameter tuning
$\mathcal{D}_{te}$	Test set of documents
$\mathcal{D}'_{te}$	Test set of documents for parameter tuning
E	Set of edges in a catalog
$F^{\mu}_{\beta_F}$	Micro-Averaged F-measure for categorization performance
$F^{\nu}_{\beta_F}$	Macro-Averaged F-measure for categorization performance
$\mathcal{F}^K$	Category match filter function for knowledge exchange service K
$\mathcal{G}$	Catalog graph
k_x	Number of cross-validation experiments performed
$\kappa_t r$	Maximum number of training documents per category used for classifier training
$\kappa_t e$	Maximum number of test documents per category used for establishing a category match
M	Catalog matching
m_i	Category match pair
μ	Minimum category similarity for filtered category pair
ν	Parameter for SVM classification, approximates the number of support vectors
P	Set of catalog perspectives
π	Precision
ϕ_{d_i}	Classification density function for document d_i
Φ	Perfect Boolean classifier function

$\hat{\phi}$	Estimated classification density function
$\hat{\Phi}$	Estimated Boolean classifier function
r	Catalog root category
ρ	Recall
ϱ_i	Rocchio Centroid vector of category c_i
$\sigma(c_i, c_j)$	Similarity measure for classes c_i and c_j
σ_{min}	Minimum category similarity allowed in structural matching
t_k	k-th term in a document space
$\mathcal{T}$	Term set for all documents
$\mathcal{T}'$	Reduced term set for document set
τ	Threshold for simple structural matching
w_{kj}	Weight of term t_k in document d_j

Part I

Introduction

Chapter 1

Introduction

"Real knowledge is to know the extent of one's ignorance."
Confucius (551 BC - 479 BC)

1.1 Document catalogs for knowledge exchange in communities

The importance of knowledge as a factor of production is increasing and knowledge may eventually become the most important factor of production in post-industrial societies. The evolution of global communication networks has facilitated the exchange of knowledge in many ways. Loosely coupled interest groups use the global means of communication to exchange their knowledge in a certain domain on a global scale. These self-spawned, voluntarily assembled *communities of interest* have quickly attracted research and economic interest, because communities have characteristics that foster knowledge exchange. Early research in sociology [Hillery 1955] and psychology [Lave 1991] described communities as a geographically constrained social phenomenon. The large potential of electronically mediated communities (also called *virtual communities*) for knowledge exchange support has been found and described in the field of social informatics and CSCW (Computer Supported Cooperative Work) [Ishida 1998a; Schlichter et al. 1998; Koch 2002]. Later, significantly influenced by research in the field of Knowledge Management [Wenger and Snyder 2000; Schmidt 2000], communities have also been described as an organizational tool that can be used to support the exchange of knowledge in organizations. Communities of interest profited greatly from the evolution of the Internet, as it allowed people, who are interested in similar topics, to globally exchange knowledge in an almost effortless fashion. The exchange of knowledge in communities of interest is focused on a domain of interest of the community members. This interest focus has a number of advantages and is also the reason why we specialized on communities of interest in this work.

Communities often use *shared information spaces* [Borghoff and Schlichter 2000] as a tool to support the community-internal knowledge exchange processes. Shared information spaces hold information that is of interest for the community as a whole, offer ways to structure the contained information and provide mechanisms for information distribution to globally dispersed community members. Information spaces that physically contain information in the form of documents are often referred to as *digital libraries*, whereas information spaces that only refer to documents are called *catalogs* [Nöhmeier 1998]. *Document catalogs* are ubiquitously used for information space organization, for example in personal catalogs such as bookmarks, document management systems or global catalogs such as the Yahoo or Google directory. Document catalogs naturally lend themselves for knowledge exchange in communities, although some special community characteristics turn out problematic in

the exchange process. So far, document catalogs have mostly been successfully used for knowledge exchange in teams. However, the principles that apply there are not directly applicable to communities. Knowledge exchange in communities has different characteristics than knowledge exchange in teams. Knowledge exchange via document catalogs means extra effort for the community members, if not supported by means of automatic exchange. The exchange may come to a halt, if the effort that has to be invested is disproportionate to the benefit of a knowledge exchange. Thus, what is required is more automated support for knowledge exchange that respects the special community characteristics.

Heterogeneous information resources like different community document catalogs cannot easily be integrated for uniform access. However, uniform access is what is required for automated exchange of knowledge via document catalogs. Different information sources can exhibit various heterogeneities. The information can be stored in different formats and moreover, it can be organized and categorized differently. For example, two document catalogs can refer to the same set of documents and still have a completely different category structure. Ways of overcoming these heterogeneities in document catalogs have to be found for automated knowledge exchange.

1.2 Problem description

An example based on a case study, which has been presented in [Bonifacio et al. 2000], illustrates the problem field that this work tackles. The case study describes a group of consultants, which could potentially generate great synergy from exchanging their project experiences. The consultants are with their clients most of the time and manage the documents that they create or read as part of their work using their personal document catalogs. The firm discovers the great potential synergy that could arise, if documents from the personal catalogs were also available to all consultants through a central document catalog. Consequently, a central document catalog is designed and the consultants are asked to store any documents or references they have used in the new central document catalog in addition to their personal catalogs.

At first, the newly introduced common catalog is a huge success and a lot of experiences are exchanged via the catalog. However, with a growing number of categories and documents in the catalog, it becomes more and more difficult for the consultants to find the right place to store or find relevant documents. But not only the sheer amount of information becomes a problem.

As new categories are added by the consultants, the categories are named in a way that is deemed appropriate by the creator of the respective category. However, the names considered meaningful by one consultant may be completely counterintuitive to another consultant, thus making it impossible for the latter to find a desired category or document. The search in the common catalog becomes tedious and time-consuming and does not contribute to the success of a consultant on a project.

Moreover, even making a document available to peer consultants via the central repository becomes a nuisance. First, if a consultant finds a document that is relevant to her work, she categorizes it into her personal catalog. In order to make the document available to her peer consultants via the central repository, she has to categorize it again into the central catalog. In order to do that, the right category in a potentially large and not easily understandable catalog has to be found first, which may require considerable searching effort. Figure 1.1 illustrates the core problem that this case study reveals.

The catalog on the left hand side of Figure 1.1 is the consultant's own personal catalog, which consists of bookmarks. The categories of the central catalog on the right hand side of Figure 1.1,

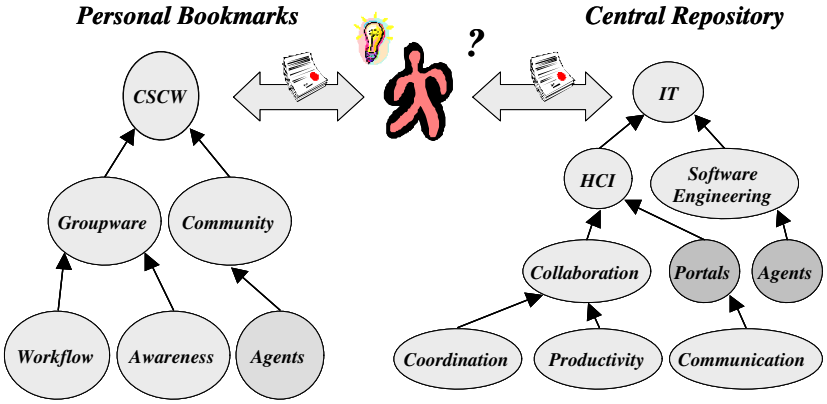


FIGURE 1.1: Different categorization schemes aggravate the knowledge flow via document catalogs

which is a document repository in this example, may be counterintuitive to her and prevent a flow of documents between the two catalogs.

The case study we have described above illustrates many of the problems we are going to tackle in this work. Although the group of consultants is more a *community of practice* than a *community of interest* (see Section 2.2), the problem of lacking knowledge exchange remains the same. The key question that puts the above mentioned problems into a common context is:

> *Consider a community of interest that uses document catalogs as a medium for knowledge exchange among the members. How can the exchange of knowledge via document catalogs within the community, as well as with other communities or other external entities be effectively supported ?*

This general problem entails a number of smaller more specific problems. In the following we are going to describe some of those specific problems, along with some existing solutions.

Knowledge exchange via catalogs in communities can be supported in many ways. A large number of commercial so-called *Knowledge Management Systems* [Maier and Klosa 2000] have attempted to effectively support the exchange of knowledge via document catalogs. However, existing systems have often neglected the social characteristics of communities as well as psychological characteristics of human information processing and consequently have not been successful [Whiting 1999]. Communities of interest are a social phenomenon that is self-organizing and largely resists external pressure. If community support functionalities are not adapted to communities, they are likely not to be accepted by community members.

Moreover, providing the general possibility to exchange documents via a catalog is not specific enough to address the needs of the community members. Existing research prototypes that deal with knowledge exchange via catalogs in one way or the other, often neglect the fact that the knowledge exchange process is a complex process, which needs to be addressed by support functionalities in many aspects. More important than the fact that documents can be exchanged, is often the way in which this exchange is enabled and how different aspects of knowledge exchange are supported. An approach that neglects these issues is also likely to fail.

Knowledge exchange via document catalogs is limited to certain kinds of knowledge. If the exchange support functionalities neglect specific kinds of knowledge or support the exchange in an

insufficient way, the exchange as a whole is at risk. Low quality knowledge exchange will not be accepted by community members and exchange participation will decline.

Heterogeneity of document catalogs is an obstacle to knowledge exchange, as we have seen in Figure 1.1. Every support functionality that aims to facilitate the exchange of knowledge via document catalogs, has to overcome heterogeneities of document catalogs in some way. Heterogeneities in document catalogs can be grouped into:

- Differences in the data formats that are used for catalog representation.
- Conceptual differences, i.e. differences of the catalog structure and categories.

The process of the resolution of heterogeneities is called *mediation*. A component that solves the problem of different data formats in the mediation process is called a *wrapper*. A wrapper converts queries and query results between different data formats. The conversion has to consider syntax as well as higher level semantics and is ideally loss-free.

A component that solves the problem of conceptual differences is called a *mediator*. A mediator has to have enough knowledge to integrate the *mediated* data sources by overcoming conceptual differences. A mediator can, for example, provide for a virtual integration of several heterogeneous catalogs by transforming a query over one catalog into queries over other catalogs. One way to resolve conceptual differences is to establish matches between categories, i.e. corresponding categories in different catalogs.

Most existing work on the resolution of heterogeneities among information sources is focused on heterogeneous databases. Those approaches, however, can only partly be transferred to document catalogs.

1.3 Goals for this work

The overall goal of this work is to provide a comprehensive, integrated and implementable concept for supporting the exchange of knowledge via document catalogs in communities of interest. Our main focus is to provide a proof of concept for the technical mediation infrastructure, on which higher level functionalities are based. The proof of concept includes a detailed description of the technical concept as well as a prototypical implementation for parts for which an experimental performance evaluation is important.

The goals for this work on an application design level, which are pursued in Chapters 3 and 4, are:

- Provide a systematic account of characteristics of communities, types of knowledge that can be exchanged via document catalogs and aspects of the knowledge exchange process in a detailed application scenario.
- Identify requirements for knowledge exchange support functionalities based on the characteristics of the application scenario.
- Provide general guidelines for community support applications that integrate the knowledge exchange support functionalities. As this integration into work processes and applications is highly situated, the integration guidelines may be at an abstract level.
- Provide a concept for application functionalities that can support knowledge exchange between community members based on the mediated exchange of documents and other catalog data.

The recommended functionalities should adhere to the requirements and characteristics of the application scenario.

- Identify requirements for the catalog mediation infrastructure with which the knowledge exchange support functionalities can be realized.

Having identified the requirements for the catalog mediation infrastructure, we can focus on the technical realization of mediation, which is the biggest contribution of this work. Our goals concerning this infrastructure, which are pursued in Chapter 5 to 8, are:

- We aim to identify the most suitable mediation approach, based on the different approaches' advantages and disadvantages with respect to our scenario. Based on the mediation approach, we aim to define a mediation process that integrates the mediation functionality into the knowledge exchange support functionalities. Moreover, we aim to identify further components that are required to realize the chosen mediation approach, such as wrappers, catalog integration and query processing components.

- To realize our mediation approach, we need to design a concept for automated virtual document catalog integration, i.e. the resolution of conceptual heterogeneities of document catalogs. The concept should adhere to the application requirements resulting from the knowledge exchange support functionalities.

- In order to solve the problem of heterogeneous data models of catalogs, a general concept for querying heterogeneous document catalogs is required. Our goal is to provide a concept for a general approach including a proof of concept for example catalog representation data models. The proof of concept should include a detailed solution to the problem of data model conversion.

- The mediation quality must be sufficiently high to provide a benefit for community members in the knowledge exchange process. We aim to provide experimental evidence that our mediation approach can provide for high-quality mediation by implementing a prototype of our mediation concept.

1.4 Related work

On the one hand, several related fields of research provide solutions, on which we rely in this work. On the other hand, solutions provided in this work may be transferred to those related fields for application. We are going to give a brief overview of those related fields and their relations to the goals of this work. More detailed accounts of related work can be found in the respective chapters of this work.

Communities have been examined as a social phenomenon and a useful tool for knowledge exchange in the CSCW (Computer Supported Cooperative Work) and Knowledge Management field. In CSCW, research on communities evolved from *Groupware* research, from which many principles can be transferred to community support research.

Information exchange via document catalogs has been studied in many previous works from various fields such as digital libraries, information management or information retrieval. Most existing

approaches are either not targeted for information exchange in communities, do not support the knowledge exchange process specifically, or do not support the exchange of different kinds of knowledge.

The exchange of documents via catalogs has been explored for a long time in the field of Information Retrieval [Baeza-Yates and Ribeiro-Neto 1999; van Rijsbergen 1979]. Information Retrieval is concerned with the approximate retrieval of unstructured data based on informal queries. Information Retrieval techniques can be used for classification of documents into a catalog, for example, and for document catalog mediation, as we are going to explain in Chapter 5. Moreover, we are also going to use Information Retrieval techniques in the novel catalog matching approach presented in Chapter 6.

In the field of digital libraries, a number of works have tackled the problem of integration of heterogeneous document catalogs, especially the query infrastructure [Borghoff et al. 1996; Borghoff and Schlichter 1996; Melnik et al. 2000]. We can use the existing approaches for querying heterogeneous information sources as guidelines for the query infrastructure in this work.

Interoperation of databases is another important field that offers a plethora of solutions concerning the integration of heterogeneous information sources. Problems that have been investigated in databases include general approaches and architectures for data integration and querying of heterogeneous data sources (see for example [Wiederhold 1992; Garcia-Molina et al. 1995; J.Bayardo et al. 1997; Raghavan and Garcia-Molina 2001b]). The basis for interoperation and integration of databases is schema matching. A number of automated schema matching approaches for databases have been published (for a survey, see [Rahm and Bernstein 2001]). In principle, the schema matching problem is closely related to the catalog matching problem, although solutions cannot be directly transferred. We are going to present an account of existing approaches to schema matching in Section 6.2.

Mapping of conceptual structures onto each other has also been explored in Artificial Intelligence. So-called *ontologies* represent formal conceptual structures, which can be used for various reasoning and information management tasks. On the one hand, numerous works on using ontologies for the integration of heterogeneous information sources have been published (see [Wache et al. 2001] for a survey). On the other hand, techniques for mapping ontologies onto each other have been presented (see [Madhavan et al. 2002]). A more detailed account of related work in this field is also included in Section 6.2.

1.5 Structure and results of this work

Our research approach in this work, is a **constructivist** approach for the biggest part. First, we specify the problem that we aim to solve. Then, we construct a solution to the problem, by motivating the elements of the solution with existing research results. Where necessary, such as for example for the proposed mediation approach, we evaluate the quality of our proposed solution **empirically by experiments**. We perform experimental evaluations, if the validity of a solution is not directly apparent from its construction.

Contributions of this work are basically on two levels. The first level is the application framework level, the second level is the technical infrastructure for catalog mediation. The application framework level is concerned with:

- the detailed account of the characteristics of the application scenario that are important for successful knowledge exchange.

- the systematic analysis of requirements for software functionalities that support knowledge exchange in communities.
- the application design of services that support knowledge exchange in communities and fulfill the above requirements.
- the identification of requirements for a mediation infrastructure that is required for the knowledge exchange services.

The technical infrastructure level is concerned with:

- the construction of a general catalog mediation approach that fulfills the requirements introduced by the knowledge exchange services.
- the construction of a catalog matching approach that can serve as the basis for catalog mediation.
- the construction of a query infrastructure concept that allows for basic interoperation of several heterogeneous document catalogs.
- the evaluation of the mediation approach to provide evidence for its effectiveness concerning knowledge exchange.

Figure 1.2 shows a structure overview of this work.

This work contains the following results.

The application scenario, in which the knowledge exchange takes place, is described in Chapter 2. We first define communities of interest and their characteristics as the basis for the requirements analysis. Thereafter, we introduce our model of the knowledge exchange process and define the document catalog model for this work. Finally, we describe among which knowledge exchange participants knowledge is exchanged in our scenario.

Requirements for an application that can support knowledge exchange in our application scenario, are described in Chapter 3. We are going to apply a process model from the Knowledge Management field to divide the general knowledge exchange process into more specific steps (see Section 3.2), for each of which we identify support requirements. Additionally, we present requirements for community support applications in existing work and apply them to our scenario. Finally, we compare different ways of using document catalogs for knowledge exchange. We show that using all available catalogs for knowledge exchange works best in our scenario and identify further application requirements that result from this approach.

The application design of the CAIMAN framework (abbreviation for "**C***ommunity knowledge exch***A***nge v***I***a* **M***edi***A***tio***N** *of document catalogs*) is motivated in Chapter 4. First, we are going to introduce the CAIMAN *knowledge exchange services*, which directly support a community member in the exchange of knowledge via document catalogs. We are going to motivate application requirements for the mediation infrastructure from the knowledge exchange services. The functionalities of the knowledge exchange services are designed to fulfill the requirements presented in Chapter 3. As the CAIMAN services need to be integrated into a community support application, we are going to present an example for this integration.

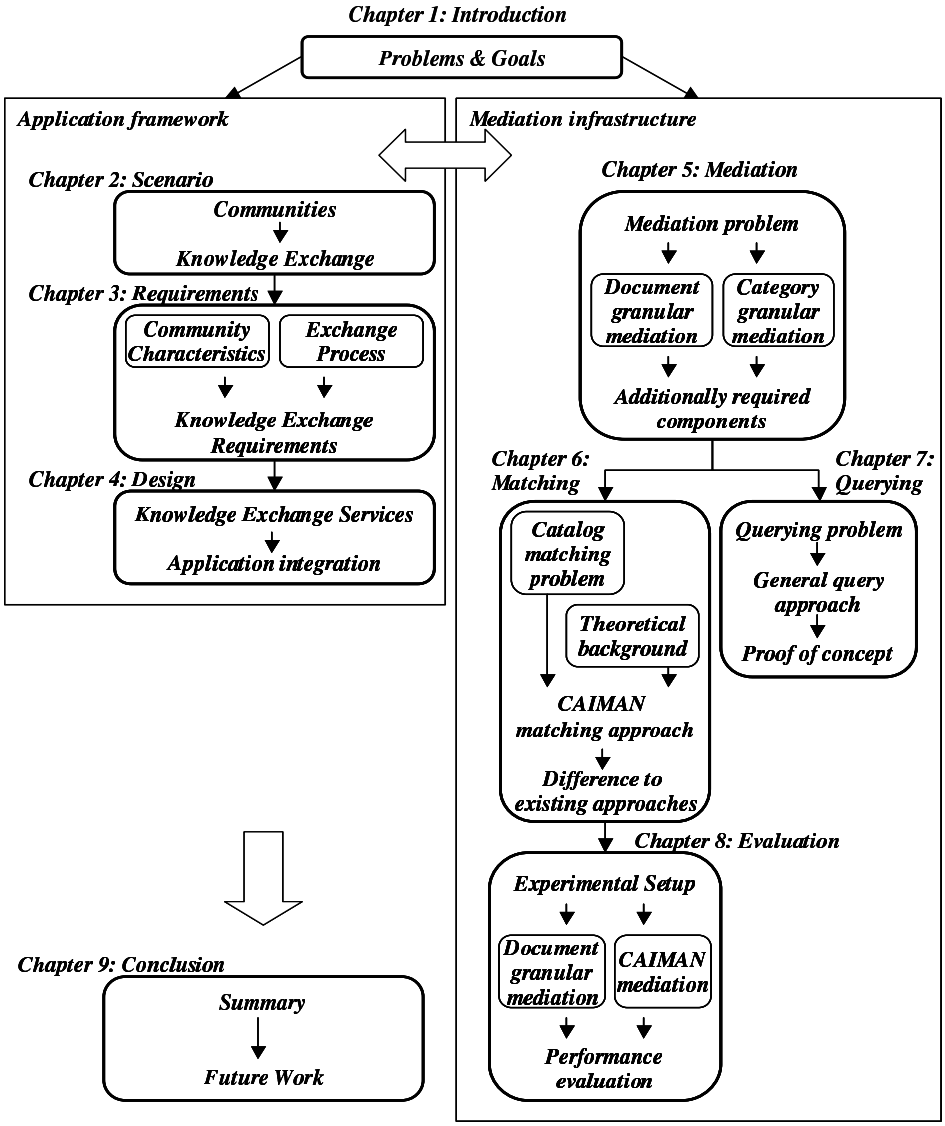


FIGURE 1.2: Structure of this work

The general mediation approach in CAIMAN is explained in Chapter 5. First, we structure the catalog mediation problem by transferring a typical database mediation solution to the catalog scenario. The structure includes a description of mediation sub-problems and in which mediation phase they are solved by which mediator sub-component. Then, we compare two principally different mediation approaches with respect to their effect on the knowledge exchange services. *Document granular mediation* can be realized with state-of-the-art text classification techniques. For *category granular mediation* there are no existing directly applicable techniques, but a number of closely related techniques have been published. We show for both mediation approaches how they can be applied to the catalog mediation scenario. We choose category granular mediation as the approach that we want to pursue, because it has more advantages in our scenario. Finally, we give an overview of the mediation process in CAIMAN and how the different mediation sub-problems are solved by the CAIMAN

matching and query components.

The catalog matching approach for category granular mediation in CAIMAN is presented in Chapter 6. Our catalog matching approach is based on techniques for automated text classification. We first motivate the three consecutive phases of the CAIMAN catalog matching approach. The consecutive phases - classification phase, category similarity phase and structural matching phase - are explained in detail. The catalog matching problem is closely related to the schema matching problem in the database field as well as the ontology matching problem in the AI field. We review existing approaches in those fields and explain what the differences to the CAIMAN approach are.

The catalog querying approach of CAIMAN is presented in Chapter 7. We present a concept for querying catalogs that are represented with different data models. Our concept is based on RDF[1] as the *lingua franca* for all catalog data sources. We provide a data model conversion concept, which allows to model other catalog representation data models with RDF. As a proof of concept for our querying approach, we show in detail, how a Topic Maps based catalog can be represented with RDF. We present how two catalogs, one represented as RDF and the other as Topic Maps, can be jointly queried with our proposed query infrastructure. Our focus in this chapter is on the general joint querying approach and on the necessary data model conversion. Other issues like query mapping are discussed only briefly.

Experimental evaluation results of the mediation quality of the CAIMAN approach are presented in Chapter 8. The experiments compare the CAIMAN approach to a state-of-the-art document granular mediation approach that is based on text classification techniques. We define a performance measure that resembles the quality of service of one of the CAIMAN knowledge exchange services. The mediation quality is evaluated automatically on two different document catalogs, which contain a user catalog and a community catalog each. The results show that the CAIMAN approach is superior to the compared document granular approach and that the mediation quality is high enough to provide a benefit to the user in the knowledge exchange process.

Summary and future work. A summary of the results and contributions of this work is presented in Section 9.1. Future work that can be a useful extension of this work is presented in Section 9.2. More requirements for the knowledge exchange services can be found if all services are fully implemented and evaluated through a user study for different catalogs in a real application context. Another interesting aspect that can be examined is how the knowledge exchange services could support collaborative catalog creation.

In order to include more catalog sources, additional wrapper concepts for different catalog representations are required. A useful extension would also be the automatic wrapper generation, given a high-level description of a data source format, as proposed in [Garcia-Molina et al. 1995].

The catalog matching approach can be expanded in several ways. We have identified the automatic detection of catalog perspectives, cross language catalog matching, the resolution of the problem with different catalog granularities, as well additional hypertext indexing functionalities as useful future extensions to this work.

Finally, we propose some first steps towards an agent based architecture for the CAIMAN framework along with a concept for integration of the CAIMAN framework into an existing general community support framework.

[1] Resource Description Format

Part II

Application Framework

Chapter 2

Application Scenario

2.1 Introduction

In this chapter, we give a detailed description of the application scenario, to which the solutions provided in this work are applicable. Our goal is to support the exchange of knowledge in communities of interest. Communities of interest, informally put, are groups of people that are loosely coupled by a common interest. The members of communities of interest communicate via the Internet, among other communication media. One frequently used means of knowledge exchange in communities of interest are document catalogs. Figure 2.1 gives a coarse overview of the application scenario.

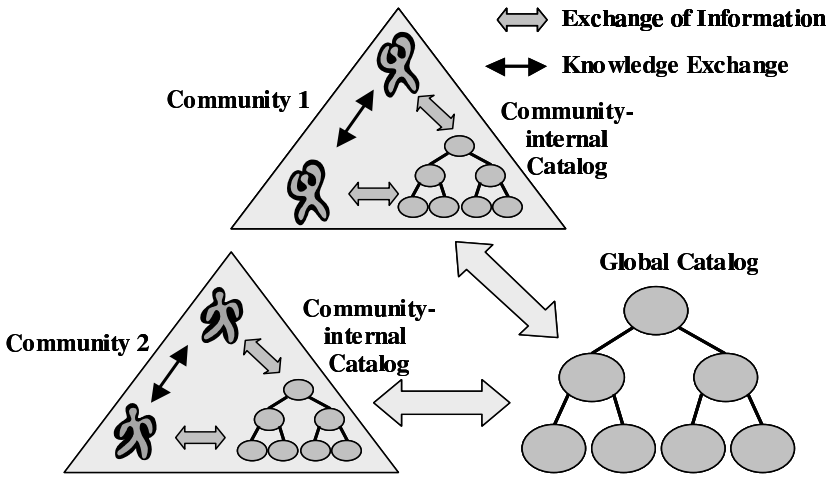


FIGURE 2.1: Overview of the application scenario

It can be seen that a knowledge exchange between community members within a community takes place. This is not a direct knowledge exchange but instead a knowledge exchange that is accomplished by an information exchange via document catalogs within their respective community. Moreover, community members can also exchange knowledge with other community-external entities via document catalogs that are globally accessible and do not necessarily belong to any particular community.

This chapter is organized as follows: we begin with detailed definitions and descriptions of the concepts involved in the application scenario shown in Figure 2.1. In Section 2.2, we define communities of interest more specifically, we describe how the community notion evolved over time and

why community characteristics are important for our application scenario. We differentiate between information exchange and knowledge exchange in Figure 2.1. The difference between knowledge and information and how information can be provided in such a way that knowledge is conveyed is described as part of a detailed description of knowledge exchange in communities in Section 2.3. We introduce and define document catalogs as a means of knowledge exchange in communities in Section 2.3.3.1. In Section 2.3.4, we describe in more detail, who the participants of knowledge exchange are in our scenario. Finally, we look at some inherent catalysts and hindrances for the exchange of knowledge in communities in Section 2.3.5.

2.2 The notion of community

The notion of *community* as a loosely coupled group of people has its roots in sociology [Hillery 1955]. However, the notion of community has become increasingly popular in recent literature from different backgrounds such as CSCW (Computer supported cooperative work) [Ishida 1998a] [Schlichter et al. 1998] [Koch 2002], KM (Knowledge Management) [Wenger and Snyder 2000] [Schmidt 2000] and e-business as well as Internet applications [Bullinger et al. 2002]. KM literature in the early 1990ies proposed a whole new perspective on businesses: the knowledge perspective [Nonaka and Takeuchi 1995] [Probst et al. 1997]. With this new perspective, communities have been discovered as an effective organizational tool to support the flow of knowledge in organizations [Borghoff and Pareschi 1998] [Wenger and Snyder 2000]. However, the even bigger interest in the community notion has been created due to the advent of the Internet in the 1990ies. With the Internet came simple means of communication and cooperation across geographical and organizational boundaries. These electronic means of communication allowed for the first time the formation of global communities around common interests and not only based on geographic proximity, as had mostly been the case before the Internet. Due to the sheer size of such global electronically mediated communities, the positive effects in terms of knowledge creation and exchange can be multiplied. Among the first to mention these electronically mediated, so-called *virtual communities* were [Rheingold 1993] and [Schuler 1995].

Under the broad and ambiguous term *community*, various more specific types of communities have been identified and described. However, due to the different backgrounds of authors in this field, the nomenclature is ambiguous and overloaded. In [Koch 2002], an overview of the community nomenclature and the relationships between the underlying community concepts is presented. Here, we are going to present a timeline of the evolution of the notion of community based on publications from different fields.

2.2.1 Historic evolution of the notion of community

The notion of *community* has first been described from a sociologist perspective as a group of people who share a geographical environment and have some kind of social interaction [Hillery 1955]. This notion has been picked up later in the field of psychology in the context of situated learning [Lave 1991]. In [Lave 1991], the term *community of practice* has been coined for a group of people who share some spatial environment. Later, the positive influence of communities on learning and knowledge flow in organizations has been recognized as useful in a business context. For the first time, communities have been seen as an organizational tool to foster learning and knowledge flow in organizations [Brown and Gray 1995, p. 78]. Communities as organizational tools have later been put into the context of goal-directed Knowledge Management in [Wenger 1998].

The first communities, in which communication among the community members was electroni-

cally mediated, have been termed *community networks* in [Schuler 1994] and *virtual communities* in [Rheingold 1993]. The community networks were mostly still geographically restricted dial-in mailbox systems with no connection to other networks around the world. The exponential growth of the Internet brought with it easily accessible, cheap and high-quality means of asynchronous many-to-many communication. The availability of these means of communication allowed for an important step in the evolution of the notion of community: the shift from the geographic definition of communities to common interests as the defining characteristic. This shift is reflected in the very general definition of community in [Mynatt et al. 1997] as a "*social grouping which exhibits in varying degrees: shared spatial relations, social conventions, a sense of membership and boundaries, and an ongoing rhythm of social interaction.*" The definition of community given in [Mynatt et al. 1997] even states that not necessarily all community members have to know each other. In [Ishida 1998a], it is stated that even frequent interaction among members is not necessary to make a community. The defining characteristics of a community according to [Ishida 1998b] are 1) the existence of the opportunity for the members to communicate over a communication channel and 2) the awareness of the existence of a community. The interaction among community members follows certain implicit social conventions that have developed over time. However, neither social conventions nor relationships among community members are codified administratively. A very broad definition of community can be found in [Schlichter et al. 1998]. There, a community is seen as a loosely coupled set of people that *have something in common* [Schlichter et al. 1998].

A number of works have overloaded the term *community* by using it for technical infrastructures that support communities. In [Mynatt et al. 1997], the term *network community* is coined for a technical infrastructure that *fosters a sense of community among the users*. An Internet-based technical infrastructure that provides a sense of community to the members has later often been referred to as *online community*. In this work, we mean the set of people when we speak about *community*. We call an application that supports communities *community support application*.

The notion of a *virtual community* from [Rheingold 1993] has been picked up and elaborated in [Carotenuto et al. 1999]. Communication in virtual communities is mostly electronically mediated, although members of a virtual community may interact on a face-to-face basis occasionally [Carotenuto et al. 1999]. The categorization of communities given in [Carotenuto et al. 1999] will be the basis of the notion of community for the remainder of this work.

2.2.2 Definition of community for this work

We have presented different attempts over time to define what a community is. For this work, we consider the following four conditions necessary for a community:

- the existence of a commonality and a describing identity among the community members.
- community membership awareness among the members.
- existence of means of communication among the community members.
- most of the interaction between the community members is electronically mediated, i.e. via the Internet.

The last point is the characteristic of a virtual community or network community, as we have seen before. **Thus, although we do not explicitly mention it in the rest of this work, we mean virtual communities when we speak of communities**. This is an important point, since most of this work describes software for community support, which does not apply to non-virtual communities. The

above conditions are necessary but not sufficient in the sense that groups of people characterized by these conditions might as well be teams. There is a gradual transition from teams to communities, as we will see in Section 2.2.6. However, a clear-cut distinction is also not required for this work, as no harm is done by including community-like teams in our community definition. We make no specific assumptions for the identification of our functionality requirements in Chapter 3 that would only apply to teams.

2.2.3 Categorization of communities

As we will show later in Chapter 3, community support applications need to be tailored very carefully to the specific characteristics of the specific community they aim to support. It is thus useful to categorize communities along different criteria in order to be able to derive specific support requirements for each class of community. As could be seen from the historical evolution of the notion of community, there have been several attempts to define different kinds of communities. An important community categorization for this work has been presented in [Carotenuto et al. 1999]. There, communities are categorized firstly by whether they share common interests or common practices. Secondly, communities are categorized by whether the focus of the community on the commonality, which the community members share, is rather tight or more broad. In [Carotenuto et al. 1999], four community categories are defined according to the introduced categorization criteria. The quadrants in Figure 2.2 are named after the four community categories defined in [Carotenuto et al. 1999].

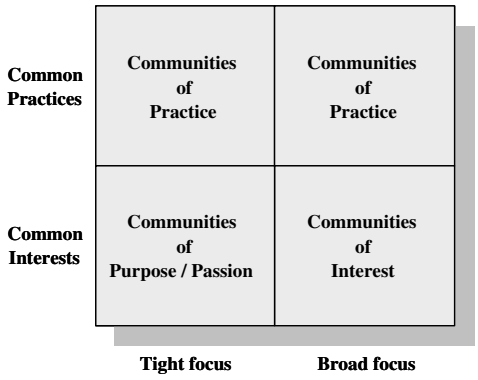


FIGURE 2.2: Community portfolio derived from [Carotenuto et al. 1999]

The four different categories are defined in [Carotenuto et al. 1999] as follows:

- *Communities of interest (COI):* have a broad focus on a set of shared interests. A prime motivation for participation is to gain awareness about what the other community members know [Ishida 1998c] and exchange knowledge.

- *Communities of practice (COP):* have a focus on common activities and practices. The focus is tight in most cases but may be broad as well. The common practices may also result from shared spatial environments [Lave 1991] in the present or in the past [Brown and Gray 1995]. For example, employees of a company that used to have their offices on the same floor in the same building and stay in contact after the company has moved, constitute a community of practice.

- *Communities of purpose:* have a tight focus on a common interest. Members usually share a common desire to forward the interests of the community as a whole.

- *Communities of passion:* are a subset of communities of purpose. Communities of passion usually have a small group of members, which are especially passionate about the pursuit of the purpose of the community.

In practice, there are no strict boundaries between these classes of communities. For example, communities of interest can be communities of practice and vice versa [Wenger 1998]. However, the classification remains useful to derive specific requirements for community support applications, as will be shown in Chapter 3. In this work, we are going to focus on communities of interest, as they are the most frequent kind of communities on the Internet and moreover knowledge exchange is their primary intention for community participation. Communities of employees of an organization, which are spawned for the purpose of developing the company's internal knowledge, are also mostly communities of interest (see also *business communities* in Section 2.2.4).

2.2.4 Community characteristics

Communities are ubiquitous in everyday life [Koch 2002] and can develop in various environments. Some characteristics of communities remain the same, independent of the environment they grow in and some characteristics depend on the community environment.

Voluntary membership. An important characteristic of communities is that their members assemble voluntarily [Carotenuto et al. 1999]. The motivation of community members for participation in communities is purely intrinsic. The members of a community participate as long as they gain something from the participation. However, this does not mean that community members always only act in a selfish way while participating. Depending on the social conventions that grew in a community over time, the rewards for the single participant can be manyfold [Mynatt et al. 1997].

Trust. Another important characteristic of communities is, that a high level of trust among the community members simplifies communication and exchange of knowledge. However, trust among community members is mostly restricted to the focus of the community. For example, members of a Java Programming community trust their peers in programming issues but would certainly not trust each other as much in health issues. Trust among community members develops from the social interaction and exchange the community members engage in [Wenger and Snyder 2000]. This kind of informal trust suffices for an effective interaction among the community members, as the community members are not directly dependent on each other for success [Carotenuto et al. 1999].

Pain / gain balance. A principle that holds in every community is what we call the delicate *balance between pain and gain*. What we mean by this, is that community members participate in the community to gain something from the community. On the other hand, participation requires an effort and compliance with the social conventions of a community. A community member would only stay an active member, if the participation gain was higher than the participation drawbacks. Typically, the social conventions in a community have evolved over time through the community members. The balance between pain and gain is also the reason why communities reject external influences concerning their interaction and social conventions. It is crucial for the success of a community that social conventions are not interfered with and participation in the community purpose is as easy as possible (see also Chapter 3).

2.2.5 Community environments

Communities can grow in different environments with varying groups of people as their potential members. Community environments may have an influence on community characteristics as we will show here.

Personal communities [Wellman 1998] are communities that everybody, who is participating in a regular work life and social life, has around them. The members of a personal community of a person are her friends, workmates and acquaintances. Members of personal communities do not actively decide to become part of a community, but become members of personal communities through their everyday life [Koch 2002]. Personal communities grow in the environment of a person and the voluntary character of a community may become a little less important. For example, you cannot avoid forming some kind of community with your workmates, even if you don't like them. The trust level among members in personal communities may vary a lot, depending on whether they are close friends or just know each other by chance. Personal communities may not react as volatile as other communities to the pain/gain balance, as they usually have the capacity to smooth out temporary changes in the balance.

Group communities [Wellman 1998] are communities that are formed out of common interests. Members of group communities can come from various environments and backgrounds. The communities that have been examined in [Wellman 1998], met and communicated in a free, publicly accessible space, be it virtual or physical. Group communities form without external influence. We consider group communities the typical communities, for which all of the above mentioned characteristics and principles hold.

Business communities [Bullinger et al. 2002] are communities which contribute in some way to a for-profit organization. The core idea of business communities is to deepen the relationship between people more or less loosely associated to a company with the company itself. Typical examples are communities for new potential customers to make them loyal customers, for employees to motivate and qualify them to become high-potentials, and for loosely associated business partners to make them close B2B[1] cooperation partners [Bullinger et al. 2002]. The creation of business communities is principally different from other communities. Usually, communities are created in a bottom-up fashion by the community members. Business communities are created and hosted by an organization in order to bring a benefit for the organization. In business communities, there is a lot more top-down influence of the host organization on the community. This can be harmful to community participation, if the influence is perceived as interfering with the community purpose by the community members. However, this problem does not occur, if the expectations of the host organization comply well enough with the goals of the community members. For example, a community of environmental activists spawned by one of the big oil processing companies would hardly be successful. On the other hand, a customer service community, that the hosting company sees as a cheap source of product improvement hints, may function very well, if the community members gain enough useful insights on how to work around product problems from their participation.

[1] Business-to-Business

2.2.6 Teams and communities

So far, we have characterized what minimum characteristics a group of people has to exhibit to constitute a community. Now we are going to look at some aspects that differentiate communities from teams, the members of which are much more closely coupled than community members.

Differences between teams and communities

Generally, teams can be seen as a subset of communities, since teams also have the constituent community characteristics. However, when we speak about communities in this work, we refer to the kind of communities, which are not teams.

The main differences between teams and communities concern the level of connectedness of the members and the amount of commonalities among the members. These differences are illustrated in Figure 2.3. Figure 2.3 also shows that there is no clear distinction between a team and a community.

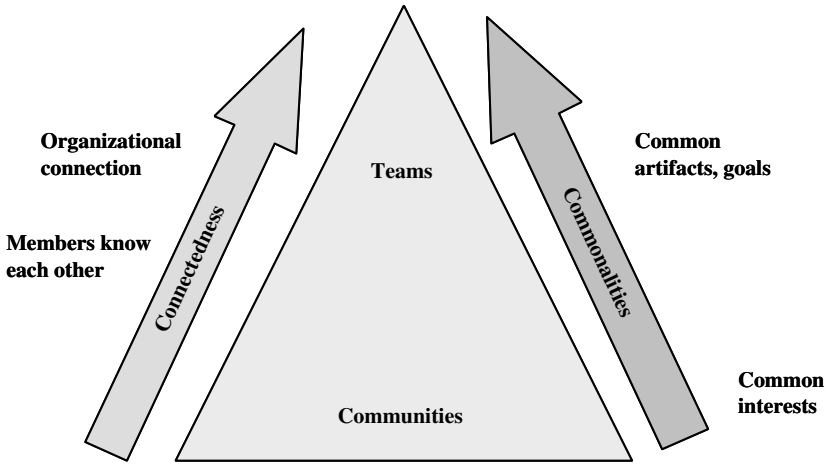


FIGURE 2.3: Communities and teams (see also [Borghoff et al. 2001])

The members of a community are not bound together by a common task, but merely by common interests [Schlichter et al. 1998]. In some cases, there may be an underlying common goal of the community, like for example in communities of passion. However, there is usually no common goal in the sense that there are no external forces that pressure the community members to collaboratively create common deliverables [Carotenuto et al. 1999]. Members of a team usually share common artifacts, which is not necessarily the case in communities. The geographical distribution of members is not a clear criterion for the distinction between teams and communities. Both teams (virtual teams) and communities may be geographically distributed. However, team members usually know each other, i.e. each team member knows all other team members. Community members, on the other hand, usually only know a fraction of their peer community members. The members of a team are tightly connected in their work processes, goals and deliverables and their connection is governed by the organization to which the team belongs. Community members have no such tight organizational connection.

Other criteria for a possible distinction between team and community, which are not depicted in Figure 2.3, are:

- *Group size*: Teams are usually smaller than communities.
- *Frequency of interaction*: Team members interact more often than community members.

- *Focus*: Teams are focused on common goals, communities are focused on common interests.
- *Organization*: Teams often have fixed interior relationships, sometimes in written form.

Relation in a business environment

Communities and teams can co-exist within organizations and across organizations. An individual member of an organization can be a member of several teams in the organization as well as several intra-organizational and inter-organizational communities. In teams, a knowledge flow without restrictions is usually desirable. In communities, on the other hand, restrictions regarding the distributed knowledge may apply. Figure 2.4 shows an example of the relation between organization, team and community.

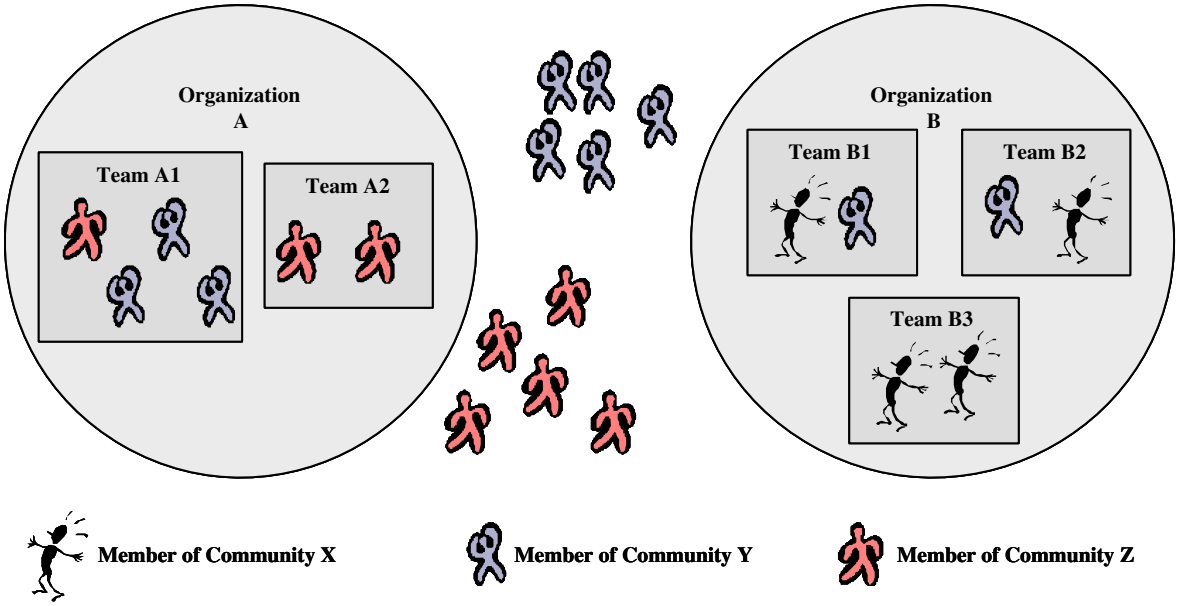


FIGURE 2.4: Relation of organization, team and community

It can be seen that community X is an internal community of organization B within which a flow of knowledge without restrictions is desirable. Community Y has members from organization A and organization B, as well as unaffiliated members. In community Y, the members of the two organizations will likely limit their flow of knowledge in specific, mission-critical areas. Privacy will also play an important role in community Y. Community Z is a business community, which involves organization A's customers into customer service processes, for example. In community Z, a free flow of knowledge is desirable but privacy remains a concern.

As this example shows, there are issues, which may force a restriction of a free flow of knowledge in communities. For the rest of this work, we leave these issues aside and assume that an unrestricted flow of knowledge in communities is desirable.

2.2.7 Example communities and community support applications

To make the description of communities and their characteristics more concrete, we are going to provide some examples for existing communities and how they are supported. In addition to that, we

are going to present a short overview over principal building blocks of community support.

An example for a successful business community is the Epinions community[2]. The purpose of the Epinions community is to allow members and visitors to exchange ratings about consumer products and services. Actually, Epinions could be called a community of communities, since the number of participants is very large, heterogeneous and divided by product groups. Moreover, the Epinions community is divided into members and visitors. Visitors can participate in the community life, but play more of a passive consumer role. The Epinions members are active product reviewers, who get rewarded for their reviewing activity. The community members in the Epinions community communicate mainly indirectly by exchanging product and service reviews and ratings of those reviews. Community members can also actively choose trustworthy partners to build a web of trust. Most of the members also publish their e-mail address for direct communication. Through the ratings and the web of trust, a measure of the trustworthiness of reviewers is calculated. This measure makes it easy for community members and visitors to pick reviews of the most trusted reviewers.

Another successful business community is the LEGO User Group Network (LUGNET [3]). The LUGNET community serves two purposes: on the one hand it is a customer service community to give customers access to resources related to the products they bought. These resources can be online education resources, help and frequently asked questions, event calendars etc. On the other hand, LUGNET is also a community of interest for Lego fans to provide them with a forum to exchange their ideas. Community members can exchange their ideas in Usenet-like forums which are focused on specific design ideas for Lego.

The SIGMOD community of computer science researchers in the field of databases is a very successful non-commercial community[4]. The SIGMOD community is an example of a community that is both virtual and also has regular face-to-face meetings of member subsets of the community that strengthen the social ties between the members. The SIGMOD members have access to a mailing list, a calendar of events and several common research resources that also give a status indication about the research going in the community.

With the above examples, we have shown some basic community support functionalities, which are being used in existing communities. An overview of existing community support functionalities categorized in seven categories is given in [Bullinger et al. 2002, pp. 323-346]:

- **Navigation and help** functionalities such as navigation bars and site maps help the user find the community functionalities she is looking for.

- **Information / Awareness** functionalities provide a sense of what is happening in the community. Such awareness functions include newsletters, member directories, calendars and the provision of online status information.

- **Communication** functionalities include synchronous communication media such as chat, instant messaging (e.g. ICQ, AOL Instant Messenger, Trillian Messenger, etc.) and Internet conferencing using both audio and video (e.g. Microsoft NetMeeting). Examples for asynchronous communication media are news groups, bulletin boards, e-mail and guestbooks.

- **Cooperation** functionalities support community members that are working together more closely. These cooperation functionalities, most of which are used in team support as well, include shared information spaces (see also [Borghoff and Schlichter 2000]), document management

[2] See http://www.epinions.com/
[3] See http://www.lugnet.com
[4] See http://www.acm.org/sigmod/

functionalities, application sharing functionalities such as a shared whiteboard, shared calendars and project management functionalities.

- **Participation** functionalities allow the community members to shape the community itself, for example through home pages, ratings for community content or incentive systems.
- **Transaction** functionalities allow the community members to perform commercial transactions.
- **Administration** functionalities allow for administration of the community member data as well as content data.
- **Recommendation** functionalities have not originally been mentioned in [Bullinger et al. 2002], however, we consider them an important functionality category. Recommendation functionalities exploit the available information about community members and content ratings by the community members to recommend resources to other community members. Examples for recommendation functionalities are book recommendations at Amazon.com, the Knowledge Pump [Glance et al. 1997, 1998] document recommendation functionality (see also Section 4.5), or expertise recommenders like Referral Web [Kautz et al. 1997] and Yenta [Foner 1997, 1999].

Combinations of these functionalities are provided by commercial community support systems. A comprehensive overview of commercial systems can be found in [Bullinger et al. 2002, p. 359].

2.3 Knowledge exchange in communities of interest

In the previous sections we have given an overview of the notion of community and defined what we mean by a community of interest in our application scenario. Now we are going to have a closer look at what we mean by the exchange of knowledge in communities of interest.

2.3.1 Characterization of knowledge

First, we are going to examine what knowledge is. In [Davenport and Prusak 1998], knowledge is defined as

> *a fluctuating mixture of structured experiences, values, contextual information and domain expertise, which comprises as a whole a structured framework for the evaluation and integration of new experiences and information. Creation and application of knowledge takes place in the heads of the knowledge carriers.*

We draw two major conclusions from this definition. The first conclusion is that knowledge is a complex composite of different things which all need to be taken into account when trying to have an effect on processes of knowledge creation and application. This is why we perform a detailed analysis of the knowledge exchange process in Section 3.2.

The second conclusion we draw for this work is that we consider knowledge only to exist in human beings. This contrasts other perspectives, e.g. from the field of *Knowledge Representation* research in Artificial Intelligence [Sowa 2000]. In Knowledge Representation (KR), knowledge is considered to be representable in electronic form in knowledge bases. Later, in Chapter 7, we are going to employ KR techniques for our purposes and thus also use the KR nomenclature concerning knowledge. However, for the rest of this work we consider knowledge to exist in people only.

Data, Information, Knowledge. In contrast to knowledge, we consider everything that can be managed by Information Technology (IT) data or information. To illustrate the relation between data, information and knowledge, we consider a small example: the list of network packet types that arrive at a certain networked computer in a university represents data. If the packets are grouped by their packet types and counted, an administrator can see the information that exceptionally high traffic is caused by a very small number of connections. Based on her experience, the administrator now creates the knowledge that students are running data intensive peer-to-peer file transfers, which should be impeded. Figure 2.5 shows the abstract relation between data, information and knowledge that underlies this example.

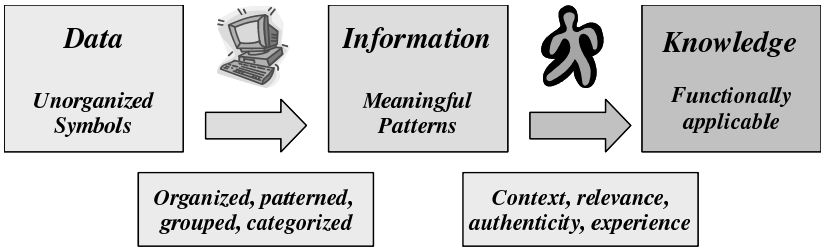


FIGURE 2.5: The relationship of data, information and knowledge

The transition between the different states data, information and knowledge is not always as discrete as in the above example. More often than not the transition is continuous [Probst et al. 1997].

Tacit and explicit knowledge are two important categories of knowledge that have been presented in [Nonaka and Takeuchi 1995]. The distinction of these two types of knowledge actually reaches back to [Polanyi 1974]. Explicit knowledge can easily be detached from the knowledge carrier, whereas tacit knowledge is highly subjective and bound tightly to the knowledge carrier. The differences between tacit and explicit knowledge are summarized in Figure 2.6.

Tacit knowledge is bound to a person, whereas explicit knowledge can be transformed to information, e.g. in the form of a document. In principle, tacit knowledge can be transformed into explicit knowledge and vice versa. However, the transformation may become very costly and lossy. Figure 2.7 shows the four transformation processes between tacit and explicit knowledge described in [Nonaka and Takeuchi 1995].

The four transformation processes shown in Figure 2.7 can be characterized as follows:

- **Socialization** is defined as the exchange of tacit knowledge through direct face-to-face contact in [Nonaka and Takeuchi 1995], if the knowledge can be shared at all.

Tacit Knowledge	Explicit Knowledge
subjective	objective
experience knowledge	intellectual knowledge
practical knowledge	theoretical knowledge
diffuse	precise
bound to its carrier	detachable from its carrier
hard to transform to information	easily transformable to information

FIGURE 2.6: The differences between tacit and explicit knowledge

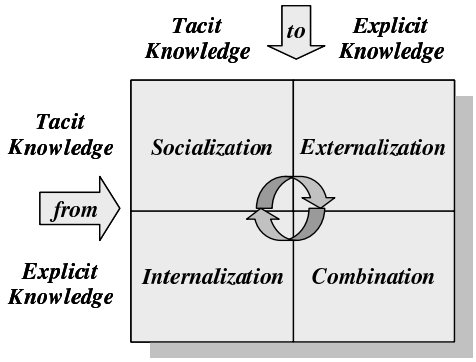


FIGURE 2.7: The transformation processes between tacit and explicit knowledge

- **Externalization** refers to the transformation of tacit knowledge of a person into new explicit knowledge, for example in the form of a document. An example for externalization is the documentation and selection of new best practices among a number of existing work practices in an organization [Borghoff and Pareschi 1998, p. 7].
- **Combination** of knowledge means creating new explicit knowledge through combination of existing bits of explicit knowledge. An example for combination given in [Borghoff and Pareschi 1998, p.7], is the combination of a company's patent directory with market research statistics, to produce knowledge about possible future products.
- **Internalization** refers to the creation of new tacit knowledge from existing explicit knowledge, for example by learning and training [Borghoff and Pareschi 1998, p.7].

More detailed descriptions of the transformation processes between tacit and explicit knowledge can be found in [Borghoff and Pareschi 1998, pp. 6-7] and [Nonaka and Takeuchi 1995, pp. 73-87].

In this work, we are going to concentrate on the exchange of explicit knowledge that can be externalized in a document catalog.

2.3.2 Knowledge exchange model

We have stated in the previous section, that we consider knowledge to exist in people only. Whatever can be stored electronically is considered information. However, that does not mean that an exchange of information does not lead to an exchange of knowledge. As we have learned in the previous section, explicit knowledge can easily be transformed into information in the form of a document or a database. And this transformation works vice versa as well, allowing information in a document to be transformed into explicit knowledge of a person again.

For our knowledge exchange model, we borrow a paradigm from the computer networking domain. The ISO/OSI model [Tanenbaum 1987] defines a stack of communication layers, the topmost of which is the application layer. In this layered network model, the communication between two applications is presented as an exchange of information via a direct connection between the two applications. In fact, the connection between two applications is of a virtual nature, the information is handed down the communication stack and all the communication of electronic signals is physically handled by the lowest layer in the stack, the physical connection layer. Thus, although the physical layer only transfers electronic signals, we can speak of an information exchange via a virtual connection on the application layer. Figure 2.8 shows how we transfer this paradigm to the domain of this work.

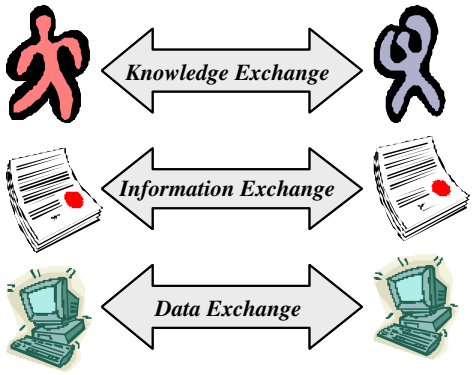


FIGURE 2.8: The model of knowledge exchange in this work

It can be seen in Figure 2.8 that we describe electronically mediated knowledge exchange using a stack of communication layers, just like in the networking example. The top layer describes a virtual knowledge exchange between two people. This knowledge exchange is not performed via a direct connection between two people, but via a connection that allows the exchange of information in the form of documents. The exchange of information in turn is realized by an exchange of data between two applications via a network connection.

The room for creative solutions in this model of knowledge exchange is in what is called *quality of service* in the networking domain. Each of the networking layers offers a certain service to the layer above it, for example the TCP[5] layer in networking offers the service of reliable end-to-end transport of data to the application layer (see [Tanenbaum 1987]). What is meant by quality of service in the networking domain are the quality guarantees for a service that a layer commits to, such as for example bandwidth guarantees, reliability, etc.

In the knowledge exchange model, an example for the quality of the information exchange service is the relevance of retrieved information with respect to a query. Another quality of service example for the information exchange service is the provision of additional context information that is delivered and presented to the user together with a transferred document.

The higher the quality of service of the information exchange, the more effective the virtual knowledge exchange based on the information exchange will be. The core of our efforts in this work aims to improve the quality of service of information exchange in order to ultimately facilitate the exchange of knowledge in and among communities of interest. A more detailed analysis of the knowledge exchange process, in order to discover potential room for improvements, is presented in Chapter 3.

2.3.3 Supporting knowledge exchange with document catalogs

In [Bullinger et al. 2002, p. 246], two principal categories of IT support for knowledge exchange have been presented:

- **Knowledge nets** (also called *knowledge yellow pages* or *knowledge resource maps* [Probst et al. 1997]) focus on supporting the exchange of tacit knowledge. *Knowledge nets* help community members to identify and locate experts as knowledge resources. Knowledge nets are essentially an index to experts in a certain field of expertise[6]. Once the expert is identified, tacit knowledge can be exchanged via personal communication or explicit knowledge can be exchanged via the

[5] Transport Control Protocol

[6] See http://www.experts-exchange.com/ for example

community means of communication, such as chat, newsgroups or e-mail. Knowledge nets often include a system of incentives, e.g. a list that publishes the most knowledgeable experts, to motivate the community members to participate in the knowledge exchange. The kinds of incentives offered vary, depending on the community culture.

- **Knowledge warehouses** (also called *knowledge structure maps* in [Probst et al. 1997]) focus on supporting the exchange of structured explicit knowledge. Knowledge warehouses are resources of explicit knowledge, which is stored in a structured way. Knowledge is exchanged between community members via the knowledge warehouse. Knowledge is first externalized by one community member, stored in the warehouse and then internalized by another community member.

A document catalog is an example of a knowledge warehouse that is frequently used as a means of knowledge exchange in communities of interest. Informally put, a document catalog is an information space in the form of a collection of documents. The documents are categorized to allow for effective navigation in the catalog and effective retrieval of documents from the catalog. Moreover, a category structure provides a context for the stored documents [Chakrabarti et al. 1998]. The sum of all topics of the documents in a catalog is called the domain of the catalog. Document catalogs do not necessarily have to have a restricted domain. Document catalogs can either physically store documents or store document references, which point to the actual physical document location. The latter case is the default in most document catalogs, thus we do not make the assumption that documents are physically stored in catalogs. Typical examples for document catalogs are global catalogs like the Yahoo[7] catalog, the Researchindex[8] collection of computer science scientific publications, and the personal bookmarks of a user.

Related Concepts. The concept of document catalogs for document organization has existed for a long time in computer science. It has originally been picked up from library science, where catalogs have been used to categorize books by their content. The research field of *digital libraries* has its roots in library science and is concerned with the management of large categorized document collections. Commonly, the term *digital library* is used for a document collection that physically stores documents, whereas a document catalog only references documents [Nöhmeier 1998]. Moreover, digital libraries are usually considered large collections, whereas document catalogs can be either small user document collections or large document collections.

The category structure of a catalog is a conceptualization of the real world domain that is represented by the catalog domain. Formal conceptualizations of a knowledge domain are called *ontologies* [Gruber 1992; Noy and Hafner 1997]. On the one hand, simple ontologies have also been used for information management, taking the role of catalog structures [Huhns and Stephens 1999; Clark 1999; Chandrasekaran et al. 1999; Pretschner and Gauch 1999]. On the other hand, the category structure of a document catalog can also be seen as a simple ontology. However, we consider seeing a catalog structure as an ontology not useful for the following reasons:

- The term ontology is used for many other formal conceptualizations and is thus too general for our purposes.

[7] See http://www.yahoo.com/
[8] See http://www.researchindex.com/

- Ontologies have a formal logic definition, which is not the case for catalog structures. In fact, the ambiguity and informal definition of catalog structures is a key advantage, as this informality makes it easier for user to create and work with the catalog structures.

In the following, we are going to give a more formal, however not formal logic, definition of the document catalog model in this work.

2.3.3.1 Document catalog definition for this work

We define a document catalog as a collection of documents, which are categorized according to the catalog structure. The basic catalog structure consists of categories and relations between categories:

- **Categories** represent a specific concept or topic of discourse and contain documents that are in some way concerned with the category topic from the category creator's point of view.
- **Directed relations** between categories signify a *sub-category* relation.
- **Perspectives** represent a categorization criterion of a catalog. Perspectives are mutually intersection-free sub-catalogs of a catalog.

We model catalogs with only one relation type, i.e. *sub-category* relations. In real-world catalogs, other labels may exist in catalogs as well. However, for our catalog mediation approach, it is sufficient to model the *sub-category* relations for the following reasons:

- Relations that are more specific sub-types of the *sub-category* relation can be subsumed under the *sub-category* relation. The additional, more specific semantics of relations is mostly ambiguous and hardly ever correctly formalized in almost all real world catalogs. After all, the ambiguity of relation definitions is one of the great advantages for the users of document catalogs. However, relying on ambiguous or incorrect relations in a mediation approach would hurt more than it would help.
- Relations that are not subsumable under *sub-category*, like for example *related-category*, are mostly relations between categories in different *perspectives* of a catalog. Our mediation approach focuses on the mediation of individual perspectives in a catalog (see Chapter 6), and thus relations between perspectives can be neglected in the catalog model.

Thus, a document catalog can be formally defined as follows:

Definition 2.1 (Document catalog) *A document catalog is defined as* $\Gamma = (\mathcal{G}, r, P, \mathcal{D})$*, where* $\mathcal{G} = (\mathcal{C}, E)$ *is the catalog graph,* r *is the catalog root category,* $P = \{P_1, \ldots, P_n\}$ *is the set of catalog perspectives and* $\mathcal{D}$ *is the set of documents in the catalog. The catalog graph* $\mathcal{G}$ *consists of the set of categories* $\mathcal{C}$ *and the set of edges* $E = \{(c_i, sub-category, c_j) | c_i, c_j \in \mathcal{C}\}$.

A document $d_j \in c_i \subseteq \mathcal{D}$ is not necessarily physically stored in the catalog, but understood as a universally resolvable link to a document.

Catalog perspectives. A catalog contains several *perspectives*, which represent a categorization criterion by which the documents in the catalog can be categorized. A specific document may then be categorized into several perspectives. The perspectives $P_i \in P$ are essentially sub-catalogs of the original catalog with additional constraints:

- The perspective root categories must be a direct sub-categories of the catalog root category.
- No document may occur twice within one perspective.

Member categories of a perspective are defined as the transitive closure of the *sub-category* edges starting from the perspective root category. Categories can only be members in one perspective. A catalog with several perspectives makes sense, since a domain of knowledge can be categorized according to different criteria. The example in Figure 2.9 shows that a collection of documents about software companies all over the world can be categorized by geographic region (first perspective on the left hand side) as well as by software application type (middle perspective) or by the industrial branch of the company's customers (right perspective).

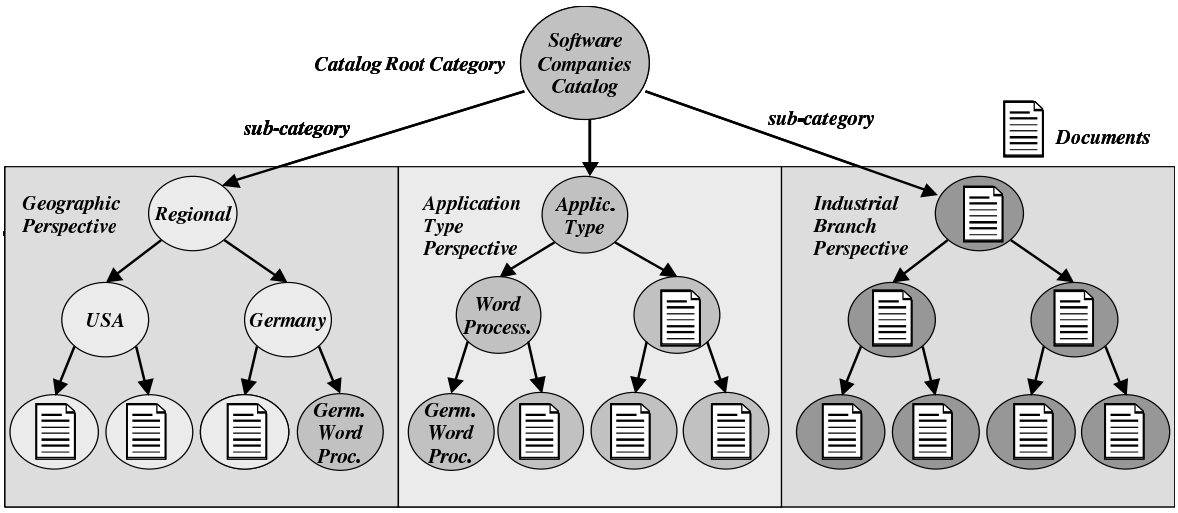


FIGURE 2.9: Example document catalog with several perspectives

There can be categories in a catalog that belong to several perspectives according to their semantics, as illustrated with the *German Word Processing Software* category in Figure 2.9. The category is a *sub-category* of *Germany* and thus belongs to the geographic perspective, but it is also a *sub-category* of *Word Processing* and thus belongs to the application type perspective. It is not possible in our catalog model that a category belongs to two perspectives, hence we assume a copy of the category to be assigned to each of the perspectives. In real world catalogs, categories that are common to two perspectives are usually represented in one of two ways: either the respective category is kept in one perspective and is linked to the other perspective with a *related-category* relation, or the category is a common sub-category of both perspectives.

In our catalog model, all category nodes, including the inner nodes of the trees in Figure 2.9 can contain documents[9].

Catalog differences. The structure of a catalog is highly subjective and represents the personal perspective of the catalog creator onto the domain of knowledge, which is conceptualized as the catalog. Different catalog authors will usually create different catalogs for the same domain of knowledge. The differences can be in the catalog perspectives, the taxonomy (i.e. category names) as well as the

[9]For some nodes in Figure 2.9 this is not visualized in the figure.

overall structure of the catalogs. These catalog differences are the reason, why it can become difficult to understand, search in and categorize new documents in catalogs other than a personal catalog. This work contributes to making the usage of heterogeneous catalogs easier. This is achieved by a semi-automatic mediation between catalogs as we are going to explain in Chapter 4 and Chapter 6.

Different kinds of knowledge can be exchanged via document catalogs. For this knowledge exchange, the user who contributes knowledge has to externalize her knowledge into information and store it in the document catalog. The knowledge recipient retrieves the information from the catalog and transforms it back into knowledge. This exchange process is analyzed in more detail in Section 3.2. The following kinds of knowledge can be exchanged via a document catalog:

- **Document content knowledge**: Explicit knowledge that is stored[10] in a document. i.e. the contents of a document.
- **Document topic knowledge**: Knowledge about the topic of a document, i.e. meta information about a document. A document is assigned to a category, because the contributor considers the document to be concerned with this topic. This knowledge is stored in the connection of a document to a category .
- **Domain structure knowledge**: Knowledge about the structure of the domain of the document catalog. This knowledge is stored in the relation between categories, and the categories themselves. In the catalog example in Figure 2.9, geographic knowledge and knowledge about software systems is stored .
- **Subjective document relation knowledge**: Knowledge about the relation between documents. If two documents are in the same category, it means that they are closely related. However, this close relation may not be obvious from the contents of the documents. We call this document relation a *subjective relation*, as it is based on the knowledge contributor's subjective opinion. The contributor's otherwise exclusive knowledge about the relation between two documents can thus be stored in the document catalog .

Types of catalogs. In and around communities of interest, catalogs are created in various contexts and by various authors respectively author groups. The following three kinds of catalogs are of relevance for this work:

- **Personal Catalogs** are created by a single author for personal information management purposes. Typical examples for personal catalogs are bookmarks, a file system folder hierarchy or a personal information space in a document management system. Personal catalogs cannot be replaced by centralized community catalogs as the community members will want to keep their privacy and control over the catalog for personal information management (see Section 3.4).
- **Community catalogs** serve information and knowledge exchange purposes in a community [Staab et al. 2000a] [Staab et al. 2000b]. Community catalogs are typically created and evolved by a subset of the community members. Community catalogs usually represent a good conceptualization of the domain of the community interest. Typical examples are the repositories of

[10] We use this simplification here for brevity. Actually, knowledge is transformed into information, which can then be stored in electronic form.

document management systems or the Researchindex[11] collection of computer science scientific publications.

- **Global Catalogs** serve as an index to large numbers of documents and are not focused on a special topic. Global catalogs are typically created and evolved by a group of expert editors. Examples for global catalogs are web catalogs like Yahoo[12] or the Open Directory Project[13]. Company-internal intranet catalogs which provide an index to all company resources can be seen as global catalogs.

As the examples above have shown, no clear distinction between community and global catalogs is possible. Some document catalogs can be both community and global catalogs or parts of a global catalog can again be community catalogs. The distinction between community and global catalogs will only be made when of immediate relevance in the rest of this work. Also, the fact that the three presented kinds of catalogs may represent different domains does not mean that references to the same document cannot be stored in several catalogs. None of the three presented types of catalogs is limited in the number of perspectives it can include.

2.3.4 Knowledge exchange partners

Our main objective for this work is to facilitate the exchange of knowledge between community members in communities of interest. However, our derived approach is applicable to the exchange of knowledge within communities and between communities and the global group of Internet users as well[14]. Figure 2.10 shows an overview over the different knowledge exchange partners in our application scenario.

Without loss of generality, we choose one community member as the center of our considerations and improve this member's knowledge exchanges with the knowledge exchange partners depicted in Figure 2.10. Exchange of knowledge among the other participating partners in this scenario can be supported analogously[15], however is not discussed in detail in this work.

2.3.5 Knowledge flow in communities of interest

As the last aspect of our application scenario, we are going to consider some characteristics of the flow of knowledge in communities of interest. These characteristics are inherent with knowledge exchange in communities and are thus constraints that also apply to our application scenario. Some of the issues presented here are going to be reflected in the requirements for support functionalities in Chapter 3. The rest of the issues must either be addressed in a broader community support context or cannot be influenced by software support at all.

The flow of knowledge has been defined in [Borghoff and Pareschi 1998, p. 5] as the exchange of knowledge between different organizational knowledge storages, such as document repositories or communities. The exchange includes the transformation between different representation forms of knowledge in the respective storages.

[11] See http://www.researchindex.com/

[12] See http://catalog.yahoo.com/

[13] See http://www.dmoz.org/

[14] The facilitation quality may not be as high for knowledge exchange among communities, as it is within a community. This will be explained in more detail in Section 4.2

[15] For knowledge exchanges, in which no community member with her personal document catalog is involved, we assume that a document catalog editor performs the interactions, which are necessary for our solution. The necessary interactions are explained in more detail in Section 4.2

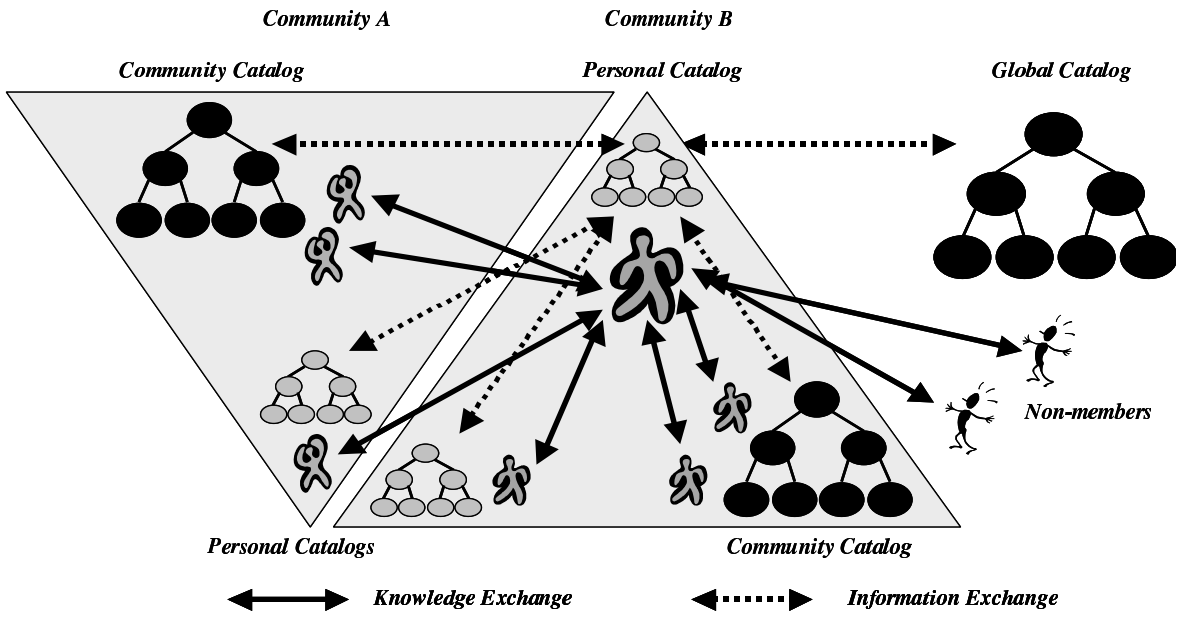


FIGURE 2.10: Knowledge exchange partners

2.3.5.1 Prerequisites for the knowledge flow

One prerequisite for the flow of knowledge in communities is above all the willingness of community members to invest an effort into the exchange of knowledge. This willingness depends on the individual as well as the community culture and general community characteristics.

Common language. Another basic prerequisite for knowledge exchange is that the exchange partners can communicate knowledge in some way. A means of communication is required as well as a representation form of knowledge that both communication partners can understand. In any communication of knowledge, the minimum requirement is that both partners understand the same language. Theoretically, a common language is enough for knowledge exchange between two people. However, knowledge can be exchanged more efficiently, if both communication partners in addition to the same language have a similar mental model of the knowledge domain and thus use similar vocabulary for communication. For example, two lawyers can exchange knowledge about a new law more efficiently than a lawyer could communicate it to a computer programmer. The common language requirement applies to all different ways of knowledge exchange, e.g. face-to-face, via document catalogs or others. In communities of interest, a common vocabulary is developed over time by the community members and used for knowledge exchange within communities. This common vocabulary makes knowledge exchange in communities very efficient.

Trust is another essential prerequisite for knowledge exchange. If a source of knowledge is not considered trustworthy by the knowledge recipient, chances are that the communicated knowledge is not accepted. For example, one would usually hesitate to accept advice from someone who has given wrong advice in the past or from someone unknown whose motives for giving advice are also unknown or even assumed malicious. The same applies if the expertise of the source of advice is unknown. In communities of interest, the community membership helps to build trust among the

members. Trust that grows in communities has two aspects: 1) trust in someone's goodwill or shared intentions, and 2) trust in someone's expertise in a field of knowledge.

Trust in someone's expertise is the smaller problem in communities of interest, since most of the members of a community of interest usually have a minimum knowledge in the domain of interest. Moreover, the expertise of a community member is openly indicated by the participation in the public forums of the community. As a result, community members tend to trust their peer community members in terms of expertise. The same applies for the integrity level. The participation and interaction of the community members on a regular basis increases personal trust among the community members. However, not all community members know all other community members and interact directly. Still, the transitive nature of trust relationships between community members mostly suffices to build a web of trust in communities.

Hindrances to the knowledge flow

Exclusion of non-members. Communities of interest support the flow of knowledge well, but there are also some inherent hindrances to the knowledge flow in communities. Communities develop their own system of values and a community culture over time. The more elaborate the community culture becomes over time, the harder it is for new members of the community to join in. A very strong commitment to the community values can ultimately even lead to exclusion of non-community members from any kind of participation in the community.

Group segregation. If the coupling of community members is too tight, negative effects of group segregation may occur. Group segregation in turn can lead to a diminished knowledge exchange of the community with the outside world. The shared cognition in the community leads to the *not-invented-here* effect. This effect describes the situation that knowledge from outside of the community is not accepted by the community members, just because it is outside knowledge, however useful and relevant it may be.

Harmful value systems. Another problem that can occur in communities is that there are no means to enforce responsibility for knowledge that is distributed within the community. This can lead to the distribution of poor quality knowledge. Moreover, values and incentives in communities are mostly such that only knowledge about successful experiences is exchanged. Mistakes and failures of the community members are more likely to be kept secret.

2.4 Summary

In this chapter, we have presented a detailed description of our application scenario and the concepts involved in the scenario description.

Communities of interest are the core concept in our application scenario. Communities of interest have been defined in Section 2.2 as a loosely coupled group of people with the following characteristics:

- Members have a broad focus on common interests.
- Members share an awareness of community membership.

- The participation of members is based on intrinsic motivation.
- The community members are not directly dependent on each other for success.
- Any measures that interfere with social practices in communities are likely to restrict the community effectiveness.

For the rest of this work, we defined that communities of interest are also virtual communities, i.e. communication among members is electronically mediated. Communities are fragile, self-motivated social constructs, which usually grow in a bottom-up fashion. Ease of fulfillment of the community purpose is essential for the existence of communities.

We gave some examples for existing communities of interest and presented an overview of community support functionalities that are used in existing community support systems.

Knowledge is defined as a complex mixture of experiences, values, contextual information and expertise, in our scenario. We consider knowledge to exist only in people. To support the exchange of knowledge, it is necessary to closely examine the kinds of knowledge to be exchanged as well as the process of knowledge exchange itself.

In this work, we focus on the exchange of explicit knowledge via document catalogs. Since document catalogs can only store information in our scenario, we model knowledge exchange via catalogs as a *virtual* knowledge exchange that is conveyed by a high-quality information exchange.

We presented four different kinds of knowledge that can be exchanged via a document catalog:

- Knowledge represented by the document contents.
- Knowledge about topics of documents.
- Knowledge about the catalog domain structure.
- Knowledge about subjective relations between documents.

Document catalogs have been defined as categorized collections of documents or links to documents in our scenario. A catalog has the following characteristics:

- The categories in a catalog are related to each other by *sub-category* relations.
- A catalog can have several perspectives, which represent categorization criteria. A document must be uniquely categorized within a perspective.
- Common categories are modelled by copies of the common categories in each concerned perspective.

Document catalogs represent the catalog creator's view onto a domain of knowledge and are thus highly subjective. The differences between catalogs need to be overcome for an effective exchange of knowledge.

Knowledge exchange partners. In our scenario, knowledge exchange between a community member and the following exchange partners is supported:

- For direct knowledge exchange with another community member, the exchange takes place via the personal catalogs of the two community members.
- For knowledge exchange with the community as a whole, the exchange takes place via the personal catalog of a member and the community catalog.
- For knowledge exchange with arbitrary community-external people, the exchange takes place via the member's personal catalog and either another personal, community or global catalog.

Finally, we have shown that knowledge exchange in communities is fostered by a culture of trust and shared cognition among the members. On the other hand, there are a number of characteristics of communities that make knowledge exchange more difficult. The relation between community members may become too close, which makes a community impermeable to outside knowledge and influences.

Chapter 3

Community Knowledge Exchange Support Requirements

"Where is the wisdom we have lost in knowledge?
Where is the knowledge we have lost in information?"
From: The Rock, T.S. Eliot (1888 - 1965)

3.1 Introduction

We have defined our application scenario in the previous chapter. We can now analyze the requirements for functionalities that can support knowledge exchange in this particular scenario. Our approach for identifying requirements is based on principles from the fields of Knowledge Management (KM), Computer Supported Cooperative Work (CSCW), psychology and sociology, which we apply to our community scenario. The identified requirements are later used to design an application framework that delivers the required support functionalities. The support functionalities must be integrated into a community support application to be deployed in communities. Therefore, we also provide general guidelines for community support application design. The community guidelines are important for successful knowledge exchange support in communities.

Our goals for this chapter are the following:

- Identify requirements for functionalities of an application framework that can support community knowledge exchange. The requirements should have a sound theoretical basis, which includes psychological aspects involved in the knowledge exchange process.

- Provide guidelines for the general application design of a community support application that includes knowledge exchange support functionalities. These guidelines should consider the sociological characteristics of communities as well as the work context of community members.

- Find out how the available document catalogs should be best used for knowledge exchange in our community scenario. Identify application requirements for knowledge exchange via the available document catalogs.

To achieve our first goal, we are going to apply a process model from Knowledge Management to divide the general knowledge exchange process into more specific steps (see Section 3.2). For each step of the knowledge exchange process, we are going to identify requirements for knowledge

exchange support functionalities. We are going to refine the support requirements based on characteristics of human information processing and knowledge creation.

For our second goal, we are going to look at groupware requirements that have been established by empirical studies in the CSCW field. We are going to transfer these requirements to our community scenario as recommendations for community support application design. We are going to present some additional requirements based on the characteristics of communities presented in the application scenario (see Section 2).

And finally, for our third goal, we are going to discuss the advantages and disadvantages of various approaches to knowledge exchange via document catalogs in communities. Subsequently, we are going to identify additional requirements for the best approach.

3.2 Requirements for supporting the knowledge exchange process

3.2.1 Knowledge Management process blocks

The field of *Knowledge Management* emerged in the mid-nineties of the 20th century from a business administration background. Knowledge Management differs from existing management approaches through a new paradigm: organizations are seen from the knowledge perspective and knowledge is treated as an organizational asset just like financial or real estate assets. There have been many attempts to define Knowledge Management [Beckman 1999] and one of the most cited is the one given in [Probst et al. 1997]:

Definition 3.1 (Organizational Memory and Knowledge Management) Organizational Learning describes the change processes of the organizational knowledge base. Knowledge Management (KM) comprises the formation and direction of these change processes in pursuit of new business value.

Knowledge Management became a big hype in the late 1990ies. A lot of previously well-known techniques and approaches that dealt with knowledge in some way have then been labelled Knowledge Management. As a result, the concept of Knowledge Management became somewhat diluted.

The Knowledge Management literature offers a plethora of theories and case studies that deal with varying aspects of knowledge and organizations, individuals [Reinmann-Rothmeier and Mandl 2000] or society as a whole . For a survey on the state of the art of Knowledge Management primarily from a business administration perspective, the interested reader is referred to [Beckman 1999]. Some aspects of the information technology perspective of Knowledge Management are presented in [Borghoff and Pareschi 1998].

In this section, we use a Knowledge Management process model to structure and analyze the process of knowledge exchange between members of a community of Interest. Having analyzed the structure of the knowledge exchange process, we can derive support functionalities heuristically. The process model we are going to use has been presented in [Probst et al. 1997]. The process model resulted from the analysis and aggregation of experiences from management consulting on aspects of Knowledge Management. The Knowledge Management process model identifies basic classes of processes of a Knowledge Management effort in an organization. Each of the process blocks are general solution approaches for problem blocks that may arise in organizations as part of a Knowledge Management effort. Figure 3.1 shows an overview of the process blocks in [Probst et al. 1997].

The Knowledge Management process model consists of six core process blocks, which can be seen in the lower two thirds of Figure 3.1. The two additional blocks in the upper third of Figure 3.1 allow a strategic management of knowledge in an organization. All of the process blocks can be seen

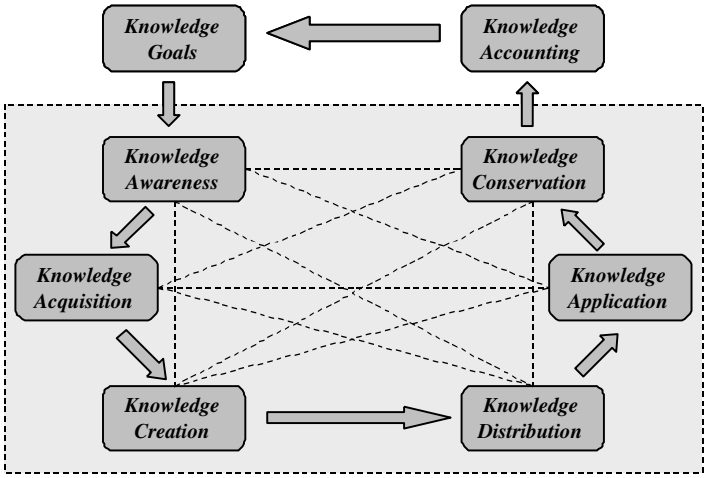


FIGURE 3.1: Knowledge Management problem classes / process blocks in [Probst et al. 1997]

as possible fields for intervention in a management sense, to implement Knowledge Management measures in an organization. The core Knowledge Management process blocks are:

- *Knowledge Awareness* describes the organization-internal and -external transparency about available knowledge resources.
- *Knowledge Acquisition* describes the acquisition of external knowledge when a knowledge gap has been identified.
- *Knowledge creation (and development)* is the intentional production of new capabilities in an organization.
- *Knowledge sharing and distribution* processes assure that existing knowledge is distributed to the right place in an organization.
- *Knowledge Application* processes ensure that the available knowledge is applied in the processes that require it.
- *Knowledge Conservation* processes ensure that knowledge that is worth conserving is conserved using appropriate media and is continuously renewed.

The two additional process blocks that complement the first six to form a strategic management feedback loop are the following:

- *Knowledge Goals* direct the efforts of Knowledge Management processes in the right direction. Knowledge Goals are very important, undirected Knowledge Management efforts can be costly. In [Probst et al. 1997], Knowledge Goals are further subclassified into normative, strategic and operative goals.
- *Knowledge Accounting* measures the success of the operative Knowledge Management efforts with respect to the Knowledge Goals.

3.2.2 Application of the process blocks to the knowledge exchange process

In [Probst et al. 1997], for each of the process blocks, recommendations based on example case studies are given on what Knowledge Management support could look like. These support recommendations cannot be directly applied to the problem of facilitating the exchange of knowledge between two community members. The approaches proposed in [Probst et al. 1997] apply to organizational entities, which are subject to managerial intervention. There are some crucial differences of our application scenario (see Chapter 2) to the organizational scenarios, for which the solutions in [Probst et al. 1997] are targeted:

- Communities are not an organizational entity that is subject to managerial intervention in a straightforward way. Culture, values and social practices of communities are created and evolved by the community members, i.e. in a bottom-up fashion, over time. All these community characteristics represent a balance between the loose coupling of community members on the one hand and the efforts that community members are willing to invest for the pursuit of the community purpose on the other hand. Thus, a community-external attempt to change the culture, values or social practices in communities as part of a Knowledge Management effort, is likely to fail.

- A second difference of communities to the organizational entities described in [Probst et al. 1997] is, that community members are not necessarily in the same administrative business framework. This implies that community members have fewer common goals in their work context and use fewer common resources than team members. As a consequence, the process blocks *Knowledge Goals*, *Knowledge Accounting* and *Knowledge Acquisition* are of very little importance for community support.

These two differences are the main reason why we present a non-intrusive software approach to support knowledge exchange in communities instead of attempting to change sociological characteristics of communities. We aim to support community knowledge exchange with a software application, which supports existing community practices while respecting existing community culture and values.

However, there are exceptions to the differences between communities and other organizational entities, as stated in the list above. A managerial intervention, which attempts to change sociological characteristics of a community, has two prerequisites: 1) there has to be some way of issuing pressure on the community members, if necessary, and 2) there have to be financial resources available to support the managerial measures. Both prerequisites are fulfilled in communities, in which all community members are employees of the same company (aka *Business communities*, presented in Chapter 2). In this work, we do not consider these special cases of communities and thus focus on non-intrusive software support measures for community knowledge exchange.

In contrast to the organization-wide approaches presented in [Probst et al. 1997], we focus on a specific detail: the exchange of knowledge via document catalogs in communities. To see what the remaining process blocks from [Probst et al. 1997] mean to this problem, we are going to define the knowledge exchange process in a more specific way:

Definition 3.2 (Knowledge Exchange Process) *Consider a community in which a knowledgeable member A has a knowledge advantage over another community member B. In the knowledge exchange process, A's surplus knowledge is identified and distributed, made aware to B and re-created in B.*

Figure 3.2 illustrates the knowledge exchange process from left to right. The figure shows that distribution of knowledge from the source is the first step before the knowledge can be made aware to the destination of the exchange process. The available knowledge can then be created at the receiver.

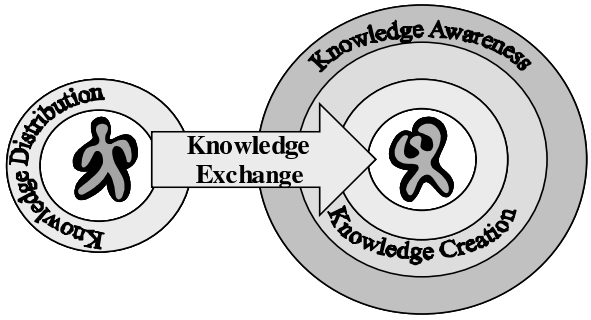


FIGURE 3.2: Steps of the knowledge exchange process with consecutive knowledge application

Knowledge sharing and distribution can be supported by many physical, organizational and technical measures [Probst et al. 1997]. Among the measures proposed in [Probst et al. 1997], we focus on the technical approaches. The goal of knowledge distribution is to assure that existing knowledge is distributed to the right place without changes. Considering whole organizations, it may not be economically viable to attempt to distribute all knowledge to all people. However, we argue that in communities of interest the interest focus is tight enough to make a complete distribution to all members useful. The basis of all electronic knowledge distribution is, that the knowledge is uniquely identifiable, accessible and represented in a processable format. It is suggested in [Probst et al. 1997], that knowledge can either be distributed in a push or a pull mode. In the push mode, the recipient of a knowledge exchange has only indirect control over the knowledge that she receives, e.g. by means of a subscription filter. It is thus crucial for the acceptance of a knowledge push service, that the delivered knowledge is perceived as relevant by the recipient. An important issue concerning the pull mode is, that the effort to pull knowledge on demand must be low enough to sustain the motivation to pull knowledge. With respect to the intrinsic motivation of the community members, it is very important that their personal and organizational privacy needs are respected.

Knowledge awareness of the knowledge recipient has to be established, once distribution of knowledge is taken care of in our knowledge exchange process. *Knowledge Awareness* for our scenario describes the community-internal and -external transparency about knowledge resources. Knowledge awareness can be facilitated by *knowledge resource maps* and *knowledge structure maps* [Probst et al. 1997]. Knowledge resource maps are catalogs of knowledge resources, such as for example skill yellow pages for people and document catalogs for externalized knowledge. Knowledge structure maps characterize the structure of a knowledge domain. These knowledge maps have to be made available to the community members in the context of their work processes. The access to both maps is usually based on a common taxonomy in the case studies presented in [Probst et al. 1997].

Knowledge creation is the next step in the knowledge exchange process, after the knowledge recipient is aware of the availability of new knowledge. According to our knowledge exchange model (see

Figure 2.8), the knowledge exchange between two community members is really a virtual knowledge exchange that is based on the exchange of information in documents. In order to make the knowledge exchange complete, the knowledge has to be re-created by the recipient. A lot of the measures proposed in [Probst et al. 1997] again refer to cultural and organizational changes, which are not applicable in communities, because communities resist any external influence that interferes with their social practices and culture. We are going to explore some specific psychological issues of knowledge creation in Section 3.2.3.

Knowledge application is not part of the knowledge exchange process depicted in Figure 3.2. However, there is significant overlap between knowledge creation and knowledge application, at least in our application scenario. Thus, some aspects of knowledge application will be included in our further considerations as well. Knowledge application is a broad notion, as it typically includes access to knowledge at a knowledge source, request of knowledge, re-creation of knowledge at the recipient and finally acting according to the newly created knowledge. Knowledge application can be hindered by a number of psychological factors and can be supported by extrinsic measures. Besides the many cultural aspects concerning the application of new knowledge in an organization, there are some that can be considered in an IT support infrastructure. In [Probst et al. 1997], it is stated that knowledge sources should be easy to use, able to deliver their knowledge just in time and be ready to connect in the sense that knowledge is stored in a processable format and relevant knowledge is identifiable. The application of knowledge can also be supported by direct integration of knowledge work into the work context. Knowledge is more likely to be applied if it is acquired in an immediate acting context [Forgas 1985].

3.2.3 Psychological aspects of knowledge creation

We are now going to examine some psychological aspects of the knowledge exchange process on the recipient side. The psychological aspect of Knowledge Management is very important, purely technically oriented Knowledge Management efforts failed for the bigger part [Whiting 1999], because they left out other aspects. The aspects discussed here, however, cannot be uniquely assigned to one of the process blocks in Section 3.2. For this section, we are going to consider a situation, in which a community member recognizes a need for new knowledge to perform a task and requests information from the community.

Human information behavior

In [Heinen and Dietel 1991] a model about information required for the fulfillment of a task has been presented (Figure 3.3).

The *subjective information requirement* is the information that the community member considers necessary to perform the task that started the information demand. However, the community member may not be aware of all information she will require for the task. In the model in [Heinen and Dietel 1991], it is assumed, that there is such a thing as an *objective information requirement*, which describes what is the objectively required information to perform the task. Ideally, the community member would possess exactly the amount of information, that's objectively required to fulfill the task. However, not all of the objective or subjective information requirement may be available in accessible information sources. The *information available* is the information that the community member can access at the moment. Depending on the *information behavior* of the community member, she may not consider it worthwhile to request all information that is within her own subjective

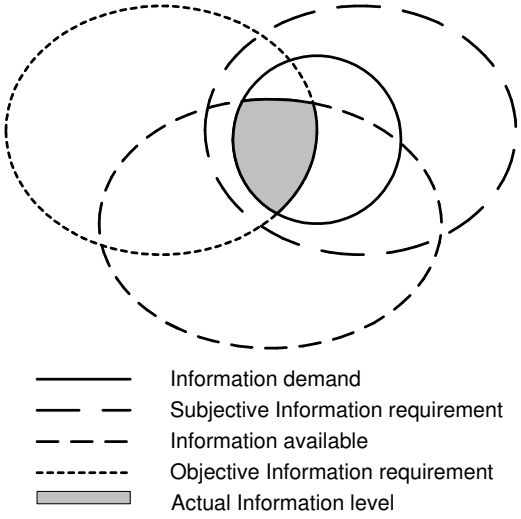


FIGURE 3.3: Information management model in [Heinen and Dietel 1991]

information requirement. The information that is actually requested, depending on the community member's information behavior, is the *information demand*, which is only a subset of what the subjective requirement amounts to. After the information request has been finished, what is left as the *actual level of information* that arrives at the community member, is the small intersection of the four sets of information in Figure 3.3.

These constraints of human information processing leave some room for improvement by a technical knowledge exchange infrastructure. We identify two ways to increase the information level with the user. One way is to make the user more aware of the information available, and thus enlarge the subjective information requirement. The other way is to supply information that has not been demanded, but is assumed to be of relevance. We do not think however, that the third way, determining the objective information requirement for a task, is feasible for an IT system.

Timing of information provision

When a community member requires knowledge for completing a task, it is very important at which stage of the task completion process the required knowledge is received by the user. Research results from social psychology [Hacker 1983] showed that the planning phase of a task is the decisive phase. It is in the planning phase for a task, that people are most receptive for new knowledge. In [Hacker 1983], it is shown that new knowledge that is available in the planning phase of a task, is taken into account and used for the solution of a task. Thus, the more relevant knowledge is available in the planning phase of a task, the more potential problems can be identified, potential mistakes and double work avoided and new knowledge can be continuously created. Ideally, knowledge is supplied proactively for a task.

Information filtering

The knowledge that a community member already has, works like a filter for new information that is received. A negative effect that can not be avoided, is that if new knowledge contrasts an established belief of the user, it is most probably not internalized, i.e. established as new knowledge. On the other hand, the better knowledge is tailored to the user's personal context and preferences, existing

knowledge and education, the better the knowledge is acknowledged and internalized by the user [Forgas 1985].

3.3 General Requirements for community support applications

We have derived some requirements for functionalities that support knowledge exchange in our application scenario. These requirements concern psychological issues of knowledge processing by the single community members. Furthermore, we derived some general requirements from knowledge processes in organizations, which we translated to requirements applicable to support functionalities. In this section, we provide the missing link between the two: general requirements and recommendations for community support software. It has been pointed out in [Grudin 1994], that the design and deployment of software that supports groups, is subject to a number of specific constraints rooted in the social dynamics of groups.

The first part of this section presents some conclusions from research on software that supports teams within organizations in their collaborative actions (aka *groupware*). Generally, these observations about groupware apply to community support software (aka *communityware*) as well. The reason for the applicability to communityware is, the acceptance and active usage of groupware is mostly voluntary, just like in communities. The fact that teams are coupled more tightly from an administrative perspective may have an influence, e.g. when team members are forced to use a certain groupware application. However, since groupware functionalities are typically not crucial for the tasks of the individual team members [Grudin 1994], forcing team members to use groupware functionalities would be counterproductive in most cases.

The second part of this section presents some recommendations specifically for community support software, which are based on community characteristics presented in the scenario chapter (see Chapter 2).

3.3.1 Experiences from groupware design and deployment

In [Grudin 1994], a number of crucial points for groupware success, based on years of experiences in implementation and deployment, have been presented. We are going to focus on the implementation issues and briefly survey the deployment issues. The issues raised here have been adapted from their original groupware context in [Grudin 1994] to our community context here.

It is stated in [Grudin 1994], that achieving successful deployment of software for group support is a lot more difficult than that of single user software or organization-wide large-scale information systems. The design and deployment of single user software does not have to take into account the social and political factors, which play an important role in communities. In fact, communityware or groupware design includes all the human computer interaction challenges of single user software and additionally the challenges that occur with the social dynamics of groups [Grudin 1994]. In contrast to communityware or groupware, large-scale information systems often represent a major investment, which has strong upper level management support. This strong support mostly leads to an adaptation of organizational processes to suit the functionalities of the information system. Members of the organization are virtually forced to use the information system for its intended purposes and adjust their respective work processes accordingly. As communityware cannot be forced on community members, the only way to avoid potential problems rooted in social dynamics of groups is to develop a thorough understanding of the work environments and consider aspects of software introduction to the workplace.

The challenges that arise with social dynamics in groups, are described in [Grudin 1994] as follows. We have adapted some of the issues as well as the solution proposals to the community scenario:

- **Disparity in work and benefit:** This issue resembles what we termed the *balance between pain and gain* in Section 2.2.4. Communityware applications often require additional work from those community members, who do not perceive a direct benefit from using the application. In our scenario, the knowledgeable community members would have to contribute their knowledge for the benefit of new and inexperienced members. The problem can be addressed by reducing the workload for those community members that benefit less, while keeping the benefit of the principal beneficiaries at a high level. In addition, processes, which simplify participation, can be suggested to the community members for application in the community.

- **Critical mass problems:** The benefits from using communityware depend on the fraction of community members who actively participate. Considering our community scenario, if too few community members participate in the knowledge exchange, the benefits for the participating members might be small. In order to avoid lacking participation, the workload for participation has to be reduced, incentives should be connected with participation and the individual and collective benefits should be made apparent to the community members.

- **Disruption of social processes:** Communityware can lead to conflicts with existing social practices or political structures or otherwise discourage key participants. An example in our community scenario is partly automated knowledge exchange. With automated knowledge exchange, the major contributors of knowledge may not receive as much credit for their contribution as in the case of a personal and active contribution of their knowledge. Recommendations given in [Grudin 1994] for addressing the problem are to develop a thorough understanding of the workplace, where the software would be used, and involve representative users in the software design at an early stage.

- **Exception handling:** Communityware is often based on the underlying assumption of fixed standard work processes. However, group activity is often characterized by exceptions and improvisation. In our community scenario, binding standards for the knowledge exchange process or the document catalogs, can be a detriment to the overall purpose. Problems with exceptions can be handled by making communityware customizable and flexible regarding varying work processes.

- **Unobtrusive accessibility:** Communityware features are used relatively infrequently compared to other workplace software features. Thus, access to communityware must not obstruct more frequently used features. On the other hand, community support features should be accessible enough to ensure community participation. One way to satisfy these requirements is to integrate the communityware or groupware features into more frequently used features that are accessed as part of the regular work processes.

In addition to the points mentioned above, which are concerned with the immediate features of communityware and groupware, there are several general problems, which concern the design and deployment of groupware and communityware. It is stated in [Grudin 1994], that evaluation of groupware can be very difficult. The fact, that the complex social context of a groupware application is crucial for its success and that this context is hard to model, makes generalization from evaluation experience unreliable. Furthermore, intuitive design, which is a successful approach in single-user application design, is largely unsuccessful and can even incur additional risk of failure in groupware design and

deployment. In contrast to single-user applications, support for local deployment of groupware is essential for its overall acceptance and success. Local deployment support can include additional consulting services, selection of pilot groups, the introduction of a task force to deal with early problems quickly, before the groupware is rejected.

3.3.2 Special communityware requirements

In addition to the requirements above, which apply to groupware just as well as to communityware, we identify some general requirements specifically for groupware.

Firstly, following the definition of communities in Section 2.2.2, the basic characteristics of communities should be supported by a community support application. The foremost requirement is to provide means for the pursuit of the community purpose. In a community of interest, this translates to means for knowledge exchange among the community members. Furthermore, means of communication among the community members should be provided as well as some indication about membership in the community.

The intrinsic motivation of community members for participation is much more decisive than in the case of groupware. The members of teams may be forced by management to adopt certain work processes and the respective groupware, at least to a certain degree. Communities, however, are more or less immune to outside pressure, since the community members' economic well-being is not dependent on community participation and success[1]. Thus, if the usage of some community support functionality does not incur an immediate benefit for every single community member, independent of the member's work environment, the functionality will not be adopted by the community members.

Moreover, team members have a common goal, which is given by their work environment. The common goals of community members are typically of lower priority than the goals given by their work environments. Thus, participation in the community and the usage of communityware are at risk, if these efforts obstruct the activities in the members' work environment. Specifically for the use of document catalogs for knowledge exchange in communities, this means that 1) services for knowledge exchange should be tightly integrated into the work processes of the community members, and 2) common catalog structures, which are forced onto the community members, are likely not to be accepted.

The issue of catalog structures and coordination of various catalogs in communities is of special interest for this work and is the focus of the following section.

3.4 Requirements for knowledge exchange via document catalogs

As the last part of our requirements analysis, we are going to identify requirements that directly result from using document catalogs for knowledge exchange. However, as there are a number of different document catalogs available for knowledge exchange in communities, there are also different principle approaches of combining the catalogs for an effective knowledge exchange.

Thus, as a first step in this section, we discuss alternative ways of using the document catalogs described in our application scenario (see Chapter 2) for knowledge exchange. Our aim is to identify the approach that is most suitable for the community scenario we described.

[1] This can be slightly different in Business communities, if the community members recruit from employees of only one company, for example. However, even then the community members typically recruit from different lines of business by which they are directly held responsible.

The second part of this section identifies further functional requirements for knowledge exchange support that result from the chosen approach of using catalogs for knowledge exchange.

3.4.1 Comparison of different approaches to using catalogs for knowledge exchange

3.4.1.1 Centralized catalog only

First, we are going to examine a scenario, in which only a centralized community catalog is used for knowledge exchange, as presented in Figure 3.4.

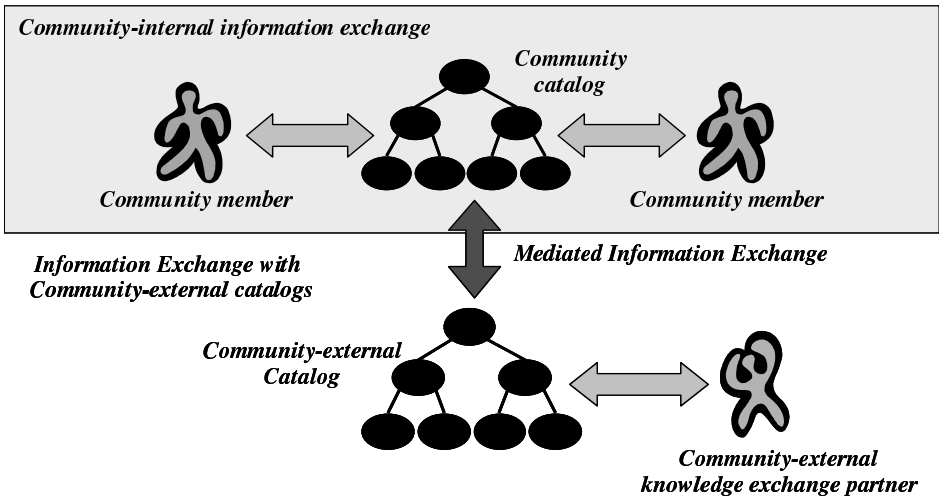


FIGURE 3.4: One centralized community catalog - one centralized perspective on the domain

It can be seen in Figure 3.4 that knowledge is exchanged between the community members through an information exchange via a centralized catalog. The community members may have personal catalogs, but these catalogs are not used for knowledge exchange.

Since catalogs are conceptualizations of a domain, a centralized catalog is a shared conceptualization of the interest domain of the community. The paradigm underlying the centralized approach is, that there exists one objective conceptualization of the community domain, which is representative for all community members.

Different community members may have different subjective, however valid perspectives onto the community domain. Thus, they may also have different ideas what the taxonomy as well as the structure of the centralized catalog should look like. To make knowledge exchange possible, all community members have to understand and adhere to the same perspective onto the community domain, despite their varying subjective perspectives. All community members have to use the centralized catalog for knowledge exchange.

This paradigm is dubbed the *god's eye paradigm* in [Bonifacio et al. 2000], because of the underlying assumption that the author of the centralized catalog knows enough about the domain of interest to make the centralized catalog suitable for all members. The centralized catalog paradigm is for example used in standardized medical catalogs like the *OHSUMED* catalog[2].

Advantages of knowledge exchange using a centralized catalog are shown in the following:

[2] See ftp://ftp.ics.uci.edu/pub/machine-learning-databases/ohsumed/

- There are no ambiguities regarding the categorization of documents. All community members exchange knowledge using the same categorization scheme and taxonomy.
- No complex technical infrastructure for catalog mediation is required. What is still required, however, are basic knowledge exchange support functionalities like notification and retrieval services.
- Through the constant usage of the centralized catalog, the perspectives of the community members can assimilate to each other. This may lead to another increase in knowledge exchange.

Disadvantages of using only a centralized catalog for knowledge exchange are as follows:

- For most domains, there is no such thing as one objectively right perspective along which the domain can be conceptualized. Thus, in a centralized catalog, a lot of aspects of the domain are always neglected.
- The personal perspectives of community members on a domain are not represented in the catalog. Thus, these personal perspectives cannot be made aware to other community members. However, these subjective perspectives are often very useful and can contribute to knowledge creation in the community.
- A community member is typically a member in several communities. Ongoing exchange of knowledge via several centralized community catalogs will require more effort than the intrinsically motivated community member is willing to invest. Thus, the knowledge flow in a community with a centralized catalog may come to a halt.
- A community member must learn and understand the centralized catalog schemes of all communities that she is a member of. Community members may not agree with the catalog structure, but are forced to use the catalog for knowledge exchange.
- The level of distrust in and misunderstanding of the centralized catalog scheme may decrease the categorization quality of information stored in a catalog dramatically. The same applies for the quality of information retrieved from the centralized catalog.
- A centralized catalog must represent the complete community domain and thus requires an authority, which has enough expertise to create and maintain the whole catalog.
- If a community member discovers a new relevant document, she will usually categorize the item in her own catalog. The effort to submit and categorize the document to the community catalog is too high and there is no incentive for doing that.
- If a community member wants to discover new relevant documents in a community catalog, she will browse only through categories, the meaning of which she understands or considers to be the right categories from her perspective. Relevant information in other categories is not discovered.
- If a community member wants to retrieve new relevant documents from the community catalog, she will typically do that with a keyword-based search. This works well, if keywords are not ambiguous and the right keywords have been picked. Relevant documents, which do not include the keywords will not be retrieved.

Integration of personal perspectives. The complete neglect of subjective perspectives is a grave disadvantage of the centralized catalog approach. However, an attempt can be made to integrate the various perspectives into the centralized community catalog. We are not going to explore all possible ways of integrating personal catalogs into the community catalog here. Our aim is to show that integration can be a complex task and requires a human editor.

We consider two different cases for the integration of a user's personal catalog into the community catalog: 1) the catalog perspectives in the personal catalog are pairwise intersection-free with the community catalog perspectives in terms of categorization, and 2) the catalog perspectives in the personal catalog partly overlap with the community catalog in terms of categorization.

In the first case, the personal catalogs can simply be integrated into the community catalog as additional perspectives (see Section 2.3.3.1). However, finding out whether or not the perspectives of the personal catalog overlap with the community catalog can become very difficult and definitely requires a human editor.

In the second case, the integration is not so straightforward. In order to maintain the community catalog consistency, the overlapping perspectives have to be joined with as little loss as possible. If the perspectives have the same granularity and the same scope, the categories will be virtually the same in the personal and in the community catalog. However, if scope or granularity of the personal and the community catalog differ, a complex integration has to be performed by a human editor.

Both of the described cases require a human editor for the integration of a personal catalog into the community catalog. The integration of personal perspectives incurs the following **advantages**:

- It is easier for the community members to use and understand the centralized catalog, because their perspectives onto the domain of interest are partly integrated into the centralized catalog. Documents may be categorized more consistently and retrieval may yield more accurate results.
- Integration of the community members' personal perspectives into the community catalog increases the community members' trust in and understanding of the centralized catalog.

The **disadvantages** of an integration of various personal perspectives are:

- Integration of different user perspectives in one community catalog is a very hard task [Lenat 1995] that may become too complex for only one central authority, i.e. one human editor. Even if the integration can be managed, a significant level of distrust with the centralized catalog can remain.
- Integration of different personal catalogs will make the centralized community catalog more complex and thus harder to understand for community members. The community catalog may include details that are only of interest for a few community members and deter other members from working effectively with the catalog.
- It is hard to integrate new or expanding personal perspectives that are created after the initial catalog has been created.

Bottom-up catalog construction. A centralized catalog that is created by a central expert authority may create distrust in the catalog among the community members. An alternative approach is that all community members participate in the catalog construction in a bottom-up fashion. This approach also automatically leads to an integration of some of the personal perspectives of the community members and further increases trust in the centralized compared to the top-down approach. However, the bottom-up approach also incurs the following **disadvantages**:

- Since the centralized catalog is constructed by all community members, agreements for the integration are required. Chances are, that no agreements can be found among members with different perspectives. Moreover, any agreement found will always render the catalog impractical for some part of the community.

- The distributed way of catalog construction may lead to inconsistencies in the centralized catalog.

3.4.1.2 Personal Catalogs only

An alternative to the centralized catalog approach is to use personal catalogs only for knowledge exchange between the community members. This scenario is shown in Figure 3.5.

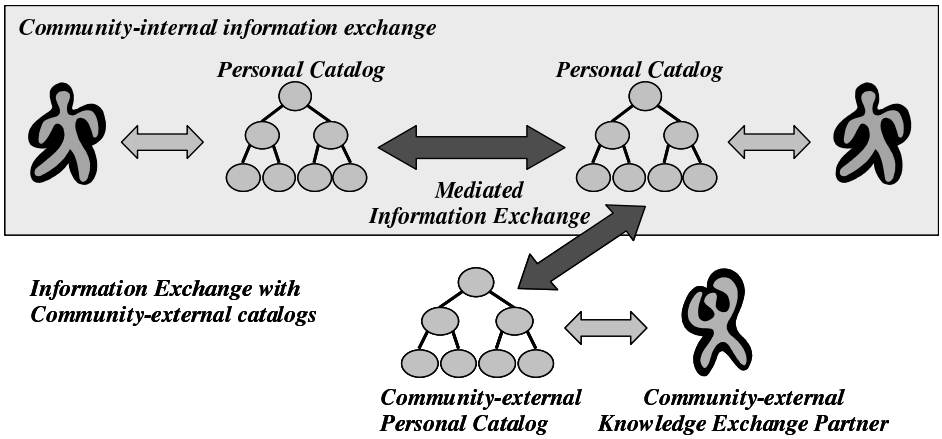


FIGURE 3.5: Using personal catalogs only for knowledge exchange

Most community members organize their documents in a personal catalog anyway, such that no additional effort is necessary for catalog creation. However, knowledge exchange in this scenario is not straightforward. One way to achieve knowledge exchange is , that each community member has to manually map her catalog to the catalogs of the other community members. The effort for this for each community member would undoubtedly be too high and lead to a halt of the knowledge exchange. The alternative is a (semi-) automatic mapping of the catalogs. The semi-automatic mapping requires a significant technical effort to map the different personal catalogs onto each other. Using personal catalogs with an automatic catalog mapping has the following **advantages**:

- For community members, knowledge exchange incurs no additional effort over the organization of documents in their personal catalog.

- The community members can keep their perspective on their community domain as well as on all other community domains and even global domains.

- Together with the categorization of documents in the personal catalog, each community member's perspective onto the community domain is conserved as well. This perspective can be of value for the whole community.

- The trust problem with centralized catalogs disappears. Trust is essential for participation in communities.

- The browsing and retrieval precision of documents may be increased, because the community members know their personal catalogs best and know best the semantics of the different categories.

On the other hand, the personal catalog approach incurs some disadvantages as well:

- A complex catalog mediation infrastructure is required.
- The direct mediation of personal catalogs incurs the risk of a low quality of the mapping between catalogs [Takeda et al. 2000]. This low precision is due to the fact that the personal catalogs may have too little conceptual overlap.
- The community knowledge is hardly accessible from outside the community without a centralized community catalog.

3.4.1.3 Centralized Catalog and Personal Catalogs

Another approach to support knowledge exchange in communities is to use both the personal catalogs of community members as well as a centralized community catalog (see Figure 3.6). If available, community-external catalogs such as global catalogs are used as well. This approach combines the advantages of the two other approaches while getting rid of most of their disadvantages.

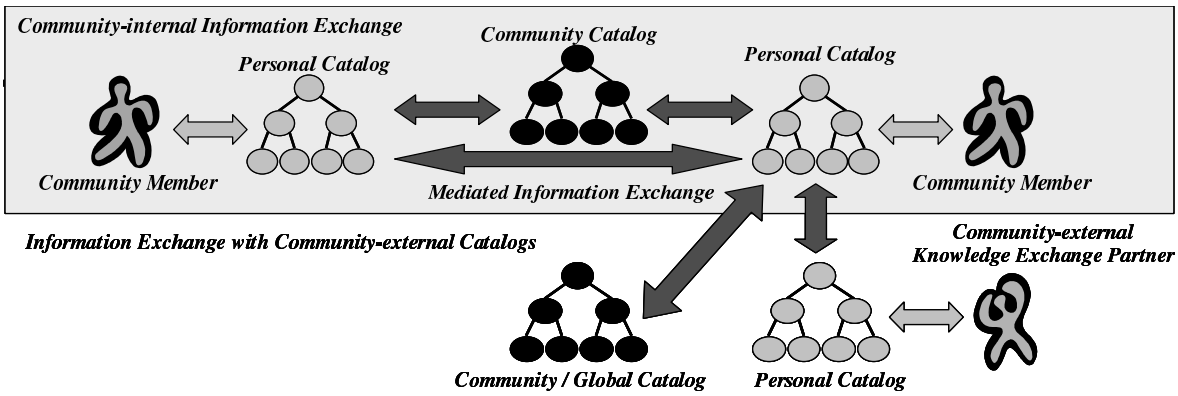


FIGURE 3.6: Using both personal and community / global catalogs to support knowledge exchange

It can be seen in Figure 3.6 that the community members see the rest of the world from the perspective of their personal catalog. The personal catalog of a community member resembles a pair of glasses, through which the other catalogs are seen. The community members store and retrieve information from their own personal catalog. The personal catalogs in turn exchange information with other catalogs within and outside of the community. In order to perform a meaningful exchange of information between the participating catalogs, the exchange needs to be mediated, i.e. a mapping between the catalog structures has to be found. Thus, in communities, a component is required, which performs a largely automated mediation between the catalog structures.

Theoretically, some community members may choose to have several personal catalogs, one for each community membership, for example. In that case, the knowledge exchange would be performed via several personal catalogs. No changes, however, would be required for the mediation approach, as several personal catalogs can simply be interpreted as several perspectives of the same catalog.

However, as additional personal catalogs would incur extra effort for the user to manage the catalogs and have no substantial benefit, we assume that all community members only use one personal catalog.

Using all catalogs for knowledge exchange combines the advantages of the two other approaches. The particular **advantages** of this approach are:

- Through the ongoing exchange of information between the catalogs, a balance between the catalogs can be achieved. Ideally, all documents from other catalogs that are of interest to a community member will eventually end up in her personal catalog by some knowledge exchange functionality. This would render extensive searches for information in other catalogs superfluous.
- Knowledge exchange incurs almost no additional effort for the community members over the management of documents in the personal catalog.
- The community members can keep their own personal perspective, which is represented by their personal catalog, for access to all document catalogs of interest.
- Membership in several communities does not incur extra effort for community members compared to membership in just one community. No additional community catalogs have to be accessed and understood, unlike in the other cases, if not all catalogs are used.
- Together with the categorization of documents in the personal catalog, each community member's perspective onto the community domain is conserved as well. This perspective can be of value for the whole community.
- The trust problem with centralized catalogs disappears.
- The browsing and retrieval precision of documents may be increased compared to the centralized catalog approach , because the community members know their personal catalogs best and know best the semantics of the different categories.
- All relevant documents from different catalogs are discovered, if mediation of the catalog structures has been successful.
- As the conceptual overlap between the personal catalogs and the community catalog is high, mediation precision can be expected to be high [Takeda et al. 2000].
- The community knowledge is easily accessible through the community catalog.

The **disadvantages** of using all catalogs are the following:

- A complex technical infrastructure for catalog mediation is required.
- The construction of the community catalog requires expertise and effort of an editor or the community members.

Using all catalogs works best in our community scenario, because the advantages of this approach outweigh the disadvantages by far. Allowing the users to keep their perspective and use their personal catalogs for all knowledge exchanges, is best suited to the flexibility and loose coupling that is required in a community. Thus, for the rest of this work, we choose to use all catalogs for knowledge exchange in our scenario.

3.4.2 Requirements for knowledge exchange using all catalogs

We have seen that using all available catalogs for knowledge exchange in communities works best. Now we can identify an additional set of requirements for knowledge exchange functionalities in our scenario:

- To support knowledge exchange, services are required, which perform an exchange of documents in a way that respects the community members' privacy, the constraints given by the sociological character of communities and the psychological aspects of human information processing.

- For a meaningful exchange of documents between the document catalogs with minimum effort by the community members, a mechanism to mediate (semi-)automatically between the involved catalogs has to be provided. Mediation in this context means, that related categories in different catalogs can be identified and used for the knowledge exchange services mentioned above. The mediation mechanism should exhibit a high enough accuracy such as to provide a benefit for each of the community members. In particular, this means that the required user interaction for mediation or correction of the mediation results must be significantly lower than if the mediation was performed manually.

- The minimum effort requirement for the mediation mechanism also leads to the requirement that the different involved catalogs have to be accessible and processable by the catalog mediation mechanism. This should involve the ability to process different formalisms for the representation of document catalogs.

3.5 Summary

In this chapter, we have identified requirements for functionalities, that can support knowledge exchange as part of a general community support application in our application scenario.

Supporting knowledge exchange. Using a process model from the Knowledge Management field, we subdivided the knowledge exchange process into three steps and included knowledge application as an important additional step. We identified the following requirements for knowledge exchange support for the four steps:

- Knowledge distribution
  - Knowledge[3] should be uniquely identifiable, accessible and represented in a processable format.
  - Information that is pushed to a community member should be highly relevant to the user's information requirements to ensure acceptance of the push distribution.
  - Information pull should incur very low effort for the user.
  - Personal and organizational privacy aspects should be respected.
- Knowledge awareness

[3] More exactly: information that represents explicit knowledge

  - The awareness for available knowledge can be increased through the provision of knowledge resource maps as well as knowledge structure maps. Increased awareness leads to a greater subjective information requirement and demand.

- Knowledge creation
  - Provision of knowledge should be personalized to the recipient's personal context to increase the acceptance of new knowledge.
  - Knowledge required for a task should be supplied pro-actively in time and assumed relevance.

- Knowledge application is not part of the knowledge exchange in our exchange model. However, since knowledge application is closely connected with knowledge creation, we have included application here as well.
  - Knowledge sources must be easy to use and ready to connect.
  - Knowledge must be available just in time and in an immediate acting context, i.e. just when the knowledge is required for the accomplishment of a task.

Community support requirements. In order to be deployed in a community, the knowledge exchange functionalities have to be integrated into a community support application. Such a community support application has to fulfill the following requirements to maintain community participation:

- The disparity in work and benefit that results from community participation has to be as small as possible.
- The benefit of community participation should be immediately apparent to all community members to avoid critical mass problems.
- Social practices and motivational aspects have to be conveyed as well as possible or at least not interfered with.
- The community support functionality should flexibly adapt to different work contexts of the community members and not rely on standardized underlying work processes.
- A sense of membership and boundaries has to be provided to the community members.
- Means of pursuit of the community purpose have to be supplied. The pursuit of the community purpose has to be facilitated enough as to keep up the intrinsic participation motivation for members.
- Means of communication among the community members have to be supplied.

Catalog usage. Our application scenario comprises personal, community and global catalogs. We compared different approaches to using the available catalogs for knowledge exchange in communities:

- Using a community Catalog only does not require a complex technical infrastructure and works well, if community members willingly comply with the categorization of the catalog and invest some extra effort for manual exchange of documents between personal and community catalog. The latter is not the case in most communities.

- Using only the personal catalogs of community members for knowledge exchange requires no additional effort from the community members. Their subjective perspectives on the community domain can be captured in their personal catalogs and stored. However, the mediation quality for the personal catalogs may be low and community knowledge is very hard to access from outside the community.
- Using both community and personal catalogs is the best of both worlds. This approach has the advantages of the personal catalogs only approach and additionally has good accessibility of community knowledge from outside the community. However, a complex mediation infrastructure is required.

By choosing the approach of using all catalogs for knowledge exchange, we incurred some additional requirements:

- Knowledge exchange services are required, which respect the requirements identified here.
- A document catalog mediation approach is required as the basis of knowledge exchange services. The mediation quality must be high enough to allow the knowledge exchange services to provide a benefit for all community members.
- A way of accessing and processing document catalogs represented in different formalisms is required.

Chapter 4

Application Level Design of the CAIMAN Framework

4.1 Introduction

This chapter describes the application level design of the CAIMAN framework (abbreviation for *"***C***ommunity knowledge exch***A***nge v***I***a* **M***edi***A***tio***N** *of document catalogs"*). CAIMAN is a conceptual framework that supports the exchange of knowledge in communities of interest via document catalog mediation. We are going to introduce the CAIMAN *knowledge exchange support services*, which have been designed to fulfill the requirements identified in the previous chapter.

We have the following goals for this chapter:

- Design knowledge exchange support services that support the exchange of the different kinds of catalog knowledge identified in the application scenario.
- Show how the individual services fulfill the requirements from Chapter 3.
- Show how the knowledge exchange support services can be integrated into an existing community support application.

In Section 4.2 we are going to introduce the general concept of how we aim to support knowledge exchange in our community scenario. We use all available catalogs for knowledge exchange. The exchange of catalog information is then enabled by a pairwise mediation of the catalogs, performed by the CAIMAN framework components.

In Section 4.3, we are going to introduce the CAIMAN *knowledge exchange support services*. We have designed the following services:

- The **information publication service** supports the distribution of the user's knowledge.
- The **related information retrieval service** supports knowledge awareness, creation and application and is concerned with document knowledge.
- The **category discovery service** also supports knowledge awareness, creation and application, but is concerned with domain structure knowledge.

We are going to describe the CAIMAN functionalities and the resulting requirements for the technical realization of the mediation infrastructure. We also present user interaction concepts, because the interaction design is an important part of the knowledge exchange support services.

In Section 4.4, we are going to present an example for the integration of CAIMAN services into an existing community support application.

Section 4.5 concludes this chapter with an overview of other frameworks that support the exchange of knowledge in communities. We are going to compare the advantages of the CAIMAN framework with the existing approaches.

4.2 CAIMAN knowledge exchange principle

The CAIMAN framework offers functionalities that can support the exchange of knowledge between a community member and her peer community members. The CAIMAN framework is targeted for our community application scenario, where document catalogs are used as a medium for knowledge exchange. We use all available document catalogs for knowledge exchange, since this approach has the most advantages.

The basic idea behind the CAIMAN framework is to allow every individual community member to keep her personal perspective on the knowledge domains of other catalogs. The user catalog is like a pair of glasses, through which the user looks at and interacts with other catalogs. To the user, it seems as if all catalog contents were integrated into the user catalog. This *virtual integration* of the other catalogs into the user catalog is technically realized with the CAIMAN mediation infrastructure. The mediation infrastructure ensures that for documents in one catalog, the "right" categories in another catalog are found, and a meaningful exchange of documents can take place. Figure 4.1 shows a conceptual overview of the CAIMAN framework.

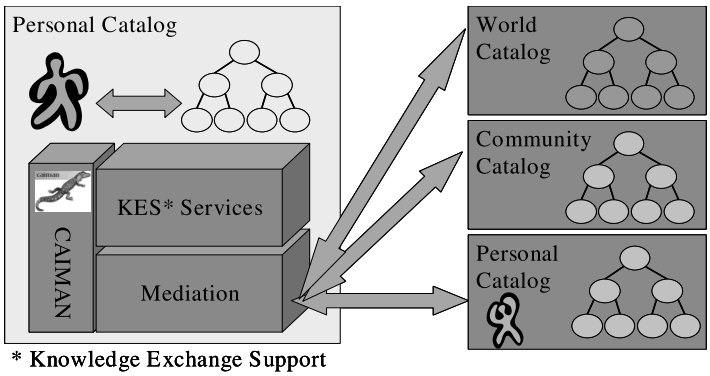


FIGURE 4.1: Conceptual overview of the CAIMAN framework

Knowledge exchange support services. The virtual integration of document catalogs is, at a first glance, an exchange of information. However, the CAIMAN knowledge exchange support services exchange and present information to the community members in such a way that a knowledge exchange is fostered. Each of the individual steps of the knowledge exchange process defined in Section 2.3.2, i.e. knowledge distribution, awareness and creation, is supported by the exchange services. In addition to knowledge exchange, the services also promote knowledge application.

The knowledge exchange support services foster the exchange of all four different kinds of knowledge stored in a catalog, i.e. *document content knowledge*, *document topic knowledge*, *domain structure knowledge* and *subjective relation knowledge* (see Section 2.3.3.1).

CAIMAN is a conceptual framework and not a full-fledged community support application. Some of the requirements from Section 3.3 can only be fulfilled, if the CAIMAN services are integrated into a community support application that provides catalog management functionalities. It is essential for the success of the CAIMAN functionalities in communities that the functionalities are integrated into the existing work processes of the community members and thus integrated into existing applications. Introduction of a new proprietary "CAIMAN application" in a community would likely be unsuccessful. All CAIMAN services and proposals for their application integration presented in this section respect the general requirements of community support from Section 3.3.

4.3 Knowledge exchange support services

We present the services in the order in which they appear in the knowledge exchange process (see Section 3.2). We begin with a service that fosters knowledge distribution and afterwards present two services that foster knowledge awareness, creation and application. For each service, we provide a conceptual overview, a user interface design concept and an account of the requirements that are fulfilled by the service.

The user interface design concepts show a fictitious document catalog management application with CAIMAN functionalities. The design concepts serve to illustrate the user interaction design of the CAIMAN services and to show how the services should be integrated into an existing application. Figure 4.2 shows the user interface concept for access to a user's personal catalog.

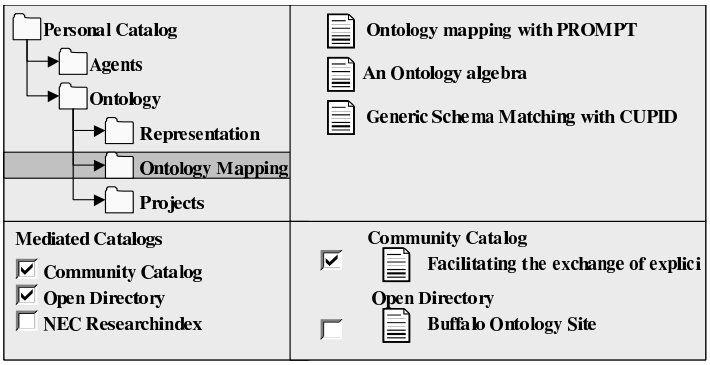


FIGURE 4.2: User interface for the personal catalog (design study)

On the top left hand side of Figure 4.2, the hierarchy of user catalog folders can be seen. The example shows that the user categorized her domain of knowledge into *Agents* and *Ontology* on the top level, where *Ontology* has the sub-categories *Representation*, *Ontology Mapping* and *Projects*[1]. The top right part shows the contents of the currently selected category, called *Ontology Mapping*. In the bottom left part, the user can choose catalogs that should be used by CAIMAN for mediation with the user catalog. In the bottom right part of the user interface, the mediation results are shown. These result are, for example, documents that are related to the currently selected category.

Requirements that are fulfilled by all CAIMAN services are:

- All CAIMAN services can operate in either **pull** mode, i.e. the user initiates the service, or **push** mode, i.e. the service is automatically initiated by the system.

[1]The category and document names in this example have been chosen for illustration purposes only and have no special importance for the service concept illustrated.

- The *disparity in work and benefit* is kept as low as possible by integrating the service functionalities into the user's work processes such that knowledge exchange incurs no extra effort. Moreover, all services have been designed to make the benefit of using them apparent to the user.
- Integration of the CAIMAN services into existing applications allows for flexible adaptation to different work contexts.
- We claim that the CAIMAN services facilitate the exchange of information and knowledge in a community of interest enough to sustain the intrinsic participation motivation of the community members.

4.3.1 Information publication

Concept. The *information publication* service allows a community member to publish a document to the community document catalog or other catalogs simply by assigning it to a category in her own personal catalog. For example, if a community member bookmarks an interesting web page into a folder in her bookmark collection, the new document will be published[2] to the categories in mediated catalogs that have been chosen by the mediation component. The publication of the newly added web page does not incur extra effort by the user.

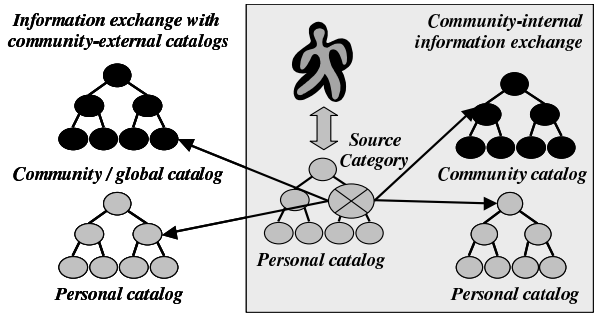


FIGURE 4.3: Conceptual overview of the information publication service

The conceptual overview of the information publication service in Figure 4.3 shows the personal user catalog in the center. The arrows from the currently chosen user category to categories in the other catalogs signify the exchange of documents or document references. The destination categories in the mediated catalogs are calculated by the mediation infrastructure. It can be seen that documents are published to community-internal user and community catalogs as well as community-external catalogs.

A user interface design study of the *information publication service* can be seen in Figure 4.4. The upper part of Figure 4.4 shows the categories and documents in the user catalog. The list of destination catalogs as well as one destination catalog that is currently chosen for display are shown in the lower part of Figure 4.4. The lower part also shows the mediation result, i.e. to which category in the destination catalog the documents from the currently chosen user category are published. The user can manually override the mediation result and choose the destination catalogs for the publication service. Not all of the categories in the user catalog are automatically published. The user can

[2] In all CAIMAN services, the term *document* actually refers to a valid document reference, for example in form of a URL. Only in case a document is not globally accessible it is physically transferred.

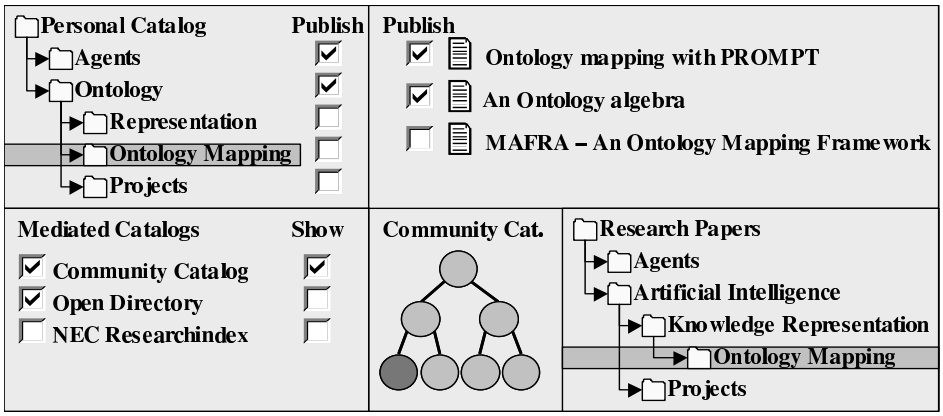


FIGURE 4.4: User interface of the information publication service (design study)

generally exclude categories or single documents from publication or choose to be consulted every time a category or document is published. Except for the choice of resources that may be published, no extra effort is incurred by the user through using the publication service.

Requirements basis. The *information publication* service has been designed to support *knowledge distribution* in the knowledge exchange process. The user's privacy is respected by letting her choose, which parts of her catalog may be published and which must not be published. It is crucial that the publication service maintains a high mediation accuracy, otherwise the service will not be accepted by community members[3]. The knowledge sources of the publication service are made public, such that the destinations can see, where the published documents come from. Thus, frequent knowledge publishers gain a reputation as an expert in the community, which is an immediately apparent benefit for publishers. The *information publication* service supports the distribution of document content knowledge, topic knowledge and subjective relation knowledge (see Section 2.3.3.1).

4.3.2 Related information retrieval

Concept. The *related information retrieval* service finds related documents for a category in the user catalog. The user chooses a personal catalog category and one or more source catalogs. The related retrieval service then retrieves documents that are found to be related to the chosen category.

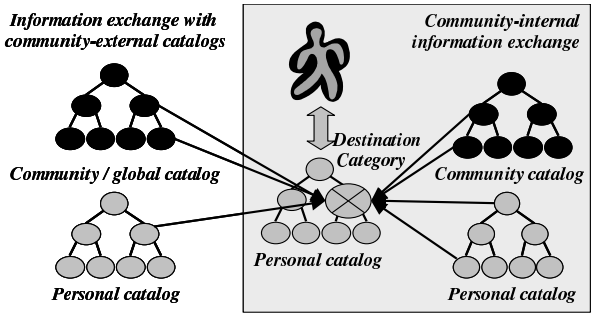


FIGURE 4.5: Conceptual overview of the related information retrieval service

[3]We are going to show in Chapter 8 that the CAIMAN mediation approach performs well enough to ensure that using the CAIMAN services incurs a benefit for community members.

The arrows in the conceptual overview of the *related information retrieval* service in Figure 4.5 signify the document transfer from mediated catalogs to the currently chosen user category. Which documents from the source catalogs end up in which category of the user catalog is determined by the mediation infrastructure. In addition to documents, information about the categories, from which documents are retrieved, is transferred.

A user interface design study for the *related information retrieval* service can be seen in Figure 4.2, which has served as an introductory example as well. Figure 4.2 shows the categories and documents in the user catalog in the upper part and the chosen mediated catalogs in the lower part. On the lower right hand side, the retrieved related documents along with their source catalogs are shown. The user can manually override the mediation result and decide which of the retrieved documents she wants take over into her user catalog. Default permissions can be given to accept all incoming related documents or none. Except for choosing the documents that may be integrated into a user category, no extra effort is incurred by the user through using the related retrieval service.

Requirements basis. The *related information retrieval* service supports *knowledge awareness and creation* in the knowledge exchange process and additionally supports *knowledge application*. The *related retrieval* service increases knowledge awareness by providing the user with what is called a *knowledge resource map* in [Probst et al. 1997]: the knowledge available in other catalogs is made available for navigation and access in the user's own personal catalog. The creation of new knowledge is facilitated by a personalized presentation of information, i.e. the virtual integration of mediated catalogs in the personal catalog. Moreover, the retrieved documents can be provided in a pro-active push fashion or in an unobtrusive pull fashion. All of the CAIMAN services can be integrated into any catalog management application that the user works with on a day-to-day basis. Through this integration into existing work processes, knowledge application can be supported. Given that the accuracy of the retrieval service is high enough, the service benefit is immediately apparent to the community members[4].

The *related information retrieval* service supports knowledge awareness, creation and application for document content knowledge, document topic knowledge as well as subjective relation knowledge (see Section 2.3.3.1).

4.3.3 Category discovery

Concept. The *category discovery* service allows the user to discover the relationships between categories in her personal catalog and categories in mediated catalogs. While the user browses through the categories of her personal catalog, closely related categories and their neighborhoods in mediated catalogs are retrieved and displayed.

The conceptual overview of the *category discovery* service in Figure 4.6 shows the user's personal catalog in the center. For each category in the user catalog that the user selects, the mediation infrastructure finds categories in the mediated catalogs that are related to the user category. The arrows connect a category in the user catalog with a related category in a mediated catalog. In case no sufficiently accurate relations can be found, no connections are shown. The shaded circles around categories in the mediated catalogs denote the category neighborhoods, which are also displayed while the user browses through her catalog. This service does not offer the transfer of documents by default,

[4]We are going to show that the CAIMAN mediation accuracy is high enough in Chapter 8.

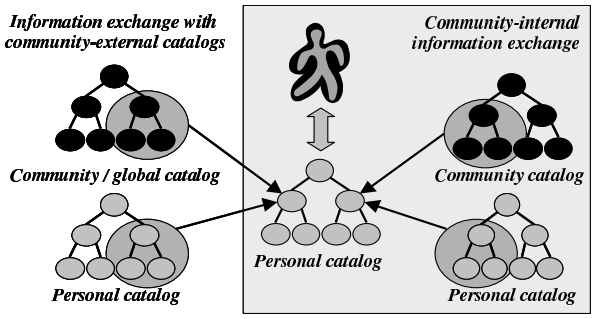


FIGURE 4.6: Conceptual overview of the category discovery Service

as the purpose of the service is the discovery of the structure of a domain of knowledge. The retrieval service can be used for document transfer, once a relevant category has been discovered.

A user interface design study for the *category discovery* service can be seen in Figure 4.7. The top leftmost part of the figure shows the user's personal catalog, the rest of the figure shows the structure and contents of the mediated catalogs. In the second column, a graphical display of sub-catalogs of two mediated catalogs can be seen. The categories that are most closely related to the currently selected user category are highlighted. In addition to the graphical structure, a category navigation pane, which displays the category names, is shown in the third column. The fourth column displays the contents of the selected categories from the mediated catalogs. The user can navigate both in her personal catalog and the mediated catalogs - the respective counterpart catalog display will refresh to show the respective related categories.

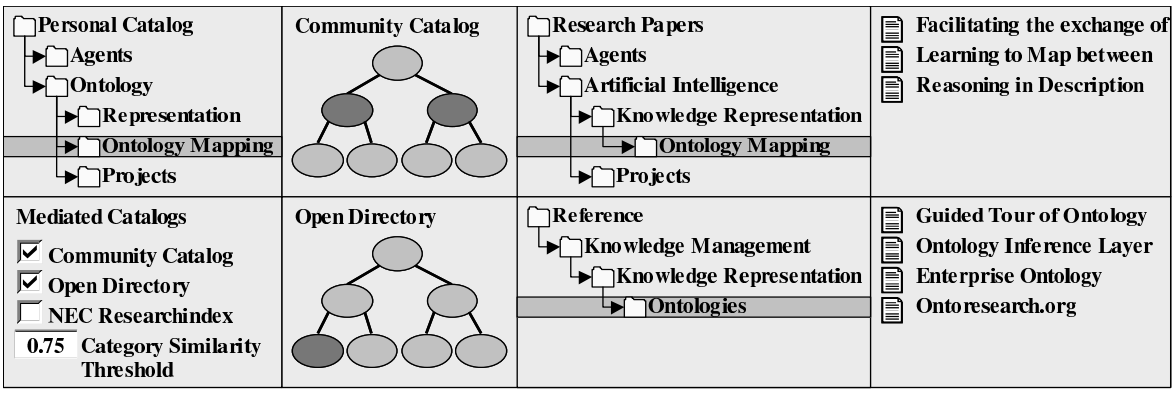


FIGURE 4.7: User interface of the category discovery service (design study)

Requirements basis. The *category discovery* service supports *knowledge awareness and creation* in the knowledge exchange process and additionally supports knowledge application. The *category discovery* service increases knowledge awareness with the user by providing an overview of the structure of a domain of knowledge from different perspectives - this is called a *knowledge structure map* in [Probst et al. 1997]. The creation of knowledge is supported by the personalized presentation of the knowledge structure map: categories in mediated catalogs are always presented together with their connection to the user's personal categories. Thereby, the service makes no difference between cate-

gories that a user has already seen and new, previously unseen categories. The discovered information is presented to the user in a pro-active, unobtrusive way. The integration of the category discovery service into the user's work processes can support knowledge application.

The *category discovery* service is the default service that is started along with the application within which the CAIMAN services are integrated. This way, there is a constant, pro-active but unobtrusive push of information about potentially relevant knowledge that is available through mediated catalogs. This increased knowledge awareness with the user leads to an increased subjective information requirement and information demand as has been explained in Section 3.2.3.

The accuracy with which related categories are found by the category discovery service has to be balanced with the amount of categories presented by the service. All very closely related categories must be retrieved and it is no disadvantage, if some remotely related categories are retrieved, as this further increases knowledge awareness.

The *category discovery* service supports the exchange of domain structure knowledge (see Section 2.3.3.1). Moreover, the user can see how the other catalogs' structure relates to her own personal catalog structure. Thus, unfamiliar categories in the community catalog, for example, are "explained" through correspondences with categories in the personal catalog.

4.3.4 Semi-automatic mediation

The user can manually override the mediation proposals made by the CAIMAN services. If the user accepts or corrects what the services propose, the respective documents are assigned to the destination category. If a proposed document or category is rejected, it only has the effect that none of the proposed documents are actually categorized in the category that has been proposed by the respective CAIMAN service. We chose this semi-automatic approach in order to avoid patronizing the user too much. Especially since the automatic mediation results cannot be expected to be perfectly personalized to the user's preferences, it would certainly not contribute to community participation to force the system's recommendations onto the user.

No new categories are automatically introduced. Similarly to why we avoid completely automatic mediation, we also avoid patronizing the user in the creation of new categories. New categories are normally inserted by the user into her personal catalog and by respective editors into the community and global catalogs. Theoretically, the mediation results of the CAIMAN services could be used as the basis for automatic category creation, if the creation of new categories seems adequate. The creation of new categories may be adequate, for example, if the destination catalog of a service is of coarser granularity than the source catalog. However, as errors in the category creation are possible, automatic creation would likely have a negative effect on community participation.

4.3.5 Application requirements for the mediation infrastructure

In the description of the knowledge exchange services we have seen that the technical mediation infrastructure has to fulfill the following requirements:

- The mediation infrastructure must be able to convey all four kinds of knowledge in document catalogs.
- The mediation quality must be high enough to bring an immediate benefit to the community members.

- Catalog contents and structures can change. The mediation infrastructure must be able to cope dynamically with changing catalog structures and contents while keeping up mediation quality.

- The mediation infrastructure must be able to scale well with large catalogs in terms of quality and mediation performance.

4.4 An example community support application with CAIMAN services

To provide an impression of the integration of CAIMAN services into an existing community support application, we show an example integration concept here. The example application we chose is the *Community-Items-Tool (CIT)*[Koch et al. 2001], a shared bookmarks and bibliographic references management tool, which is based on the *Cobricks* community support architecture (see [Borghoff et al. 2001; Koch et al. 2001]). The CIT has a web interface for easy access from different clients. The documents and references managed with the CIT are called *items*. Items can be categorized in community-global categories as well as in a hierarchical folder structure individually created by each user. These two categorization schemes can be regarded as a community catalog and personal user catalogs and thus the integration of the CAIMAN services in a meaningful way is straightforward. In addition to general catalog management functionalities, the tool incorporates community services like notification about newly added items. Two screenshots of the web user interface of the *Community-Items-Tool* can be seen in Figure 4.8.

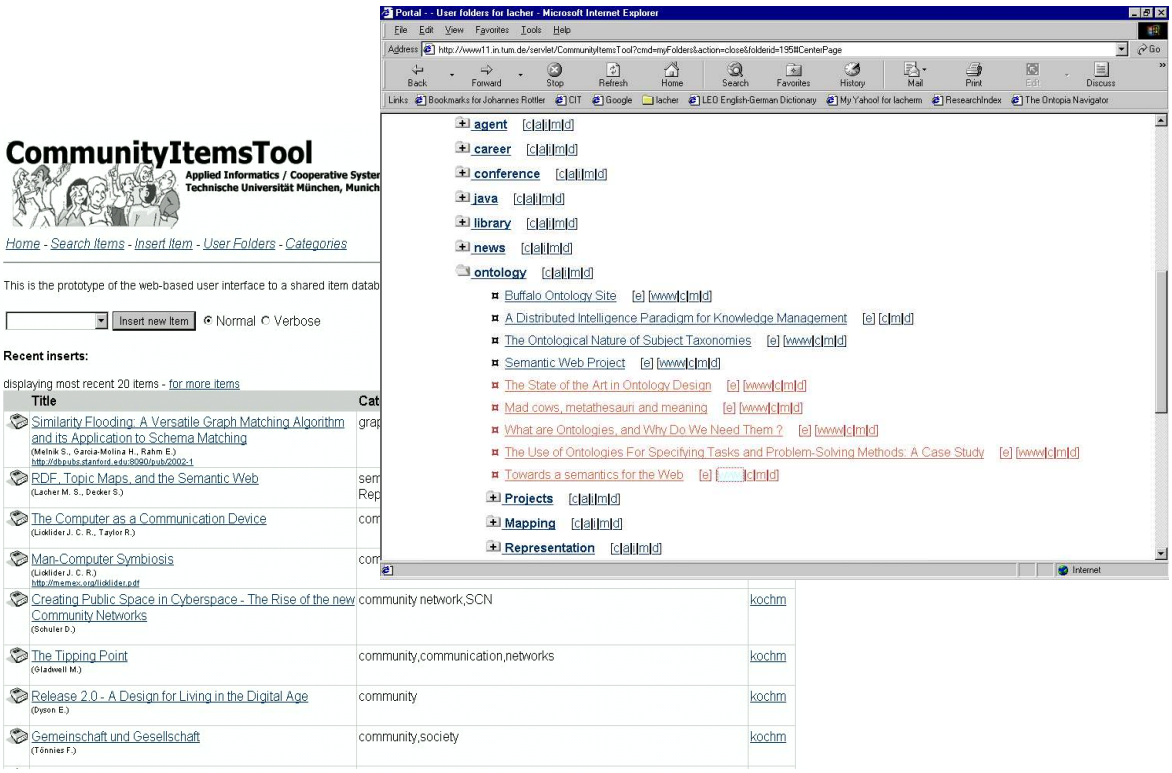


FIGURE 4.8: The CommunityItemsTool [Koch et al. 2001] with CAIMAN retrieval service

On the left hand side of Figure 4.8, we see the startup screen of the *Community-Items-Tool*, which lists the most recently added items. On the right hand side of Figure 4.8, we see how the personal user folders can be browsed in the CIT. The CAIMAN framework can calculate mediation results for the different catalogs managed by the CIT and can integrate an additional global catalog that suits the scenario, like the *Researchindex* catalog[5]. The right hand side of Figure 4.8 shows how the *related retrieval* service of CAIMAN can be integrated into the CIT. The documents displayed in the *ontology* folder are shown in different shades of grey: documents that are already contained in the user's personal catalog are presented in a dark shade. Documents from other CAIMAN-mediated catalogs are shown in a lighter grey shade. The other CAIMAN services can be integrated into the CIT in a similar fashion.

4.5 Related work

The *Knowledge Pump* system [Glance et al. 1998] aims to support effective and efficient knowledge sharing in communities. Just like in CAIMAN, the medium for knowledge exchange is a document catalog. However, document catalogs in the Knowledge Pump system are organized according to a centralized hierarchical scheme of categories, which are called *communities*. There is no possibility for individual users to retain their personal perspective onto a domain. We claim that even though concepts of incentives for participation in the knowledge exchange have been integrated into the Knowledge Pump, participation will likely suffer from lacking personalizability. The web based Knowledge Pump application offers a service that recommends documents to the user, which have been added and evaluated by other users. Recommendations are given on a per category basis based on predicted document relevance calculated with collaborative filtering algorithms. There are no means for the users to exchange subjective relations between documents nor knowledge about the domain structure. The Knowledge Pump allows to connect and access other external document catalogs such as web catalogs. In fact, the concept of the Knowledge Pump explicitly includes the independence of specific document catalogs and stresses the applicability across various catalogs. Knowledge Pump users can personalize the recommendations they receive by picking other users as their advisors, such that documents submitted by advisors are recommended preferably. Knowledge Pump is one of the few document management systems, which have been explicitly designed for communities, considering participation and user motivation issues. The key difference to CAIMAN is that no personal catalogs are allowed in the Knowledge Pump, which can lead to a somewhat thinner flow of knowledge and possibly decreased user participation compared to the CAIMAN concept.

In the *OBIWAN* project[6], ways for personalized information retrieval have been explored. The project goal was to increase precision and recall of web searches. To achieve this improvement, queries are expanded, search results are re-ranked and filtered, based on result relevance with respect to a user interest profile. The user profile is based on the hierarchy of the Excite web catalog. In [Pretschner and Gauch 1999], it has been shown that search personalization can increase search results quality. In contrast to CAIMAN, the OBIWAN application is targeted for an individual user and not a community of users. Moreover, the user profile, which is employed to personalize search results is not created by the user, but instead based on a global catalog. The user interest is inferred by observing web page visits. The only service that is offered by the OBIWAN framework is an information retrieval service.

[5] See http://www.researchindex.com/

[6] See http://www.ittc.ukans.edu/obiwan.

The *Siteseer* project offers personalized navigation of the web [Rucker and Polanco 1997]. Siteseer is a web page recommendation system that uses an individual's bookmarks and the organization of bookmarks within folders for predicting and recommending relevant pages. The usage of personal catalogs for knowledge exchange support services in CAIMAN is very similar to the usage of the user's bookmarks for recommendations in Siteseer. In both systems, documents are retrieved based on already existing documents in a category and not based on category name label similarity. However, the information retrieval service is the only service that Siteseer offers. Moreover, web page recommendations are performed on a peer-to-peer basis only, which incurs a smaller domain overlap of the user catalogs, thus risking a low query recall. The risk of low recall is increased even more by the way in which related categories are found: through an exact URL match of the contained documents. This matching approach leads to low recall, because categories with documents with similar content from different web pages would not be recognized as related. The community members with whom a document exchange is performed are chosen on a per-category basis: the more common documents two users have in one of their categories, the more relevant these two categories of these two users are to each other. The per-category approach prevents users from exchanging important knowledge about the domain structure. Moreover, the automatic choice of knowledge exchange partners in Siteseer compared to the explicit user choice in CAIMAN creates potential problems of trust and privacy. In Siteseer, new users suffer from a cold start problem, as no recommendations can be given if there are no existing contents.

In the *Macadam* project [Dourish et al. 1999b], a system that mediates between personalized versions of a centralized community document catalog has been presented. The Macadam prototype makes use of the Presto document management infrastructure [Dourish et al. 1999a], which has been developed in the larger *Placeless Documents* project. The Presto infrastructure provides a flexible document and meta-data repository, which is the technological basis for a centralized document repository with a centralized categorization scheme. The Macadam system provides means of personalization of the repository structure by individual users or communities. As a result, each user can have her own personal perspective on the repository. Moreover, an approach to mediate between different perspectives is proposed. Adaptations to the categorization scheme are not actually committed on the centralized scheme, but recorded as so-called contexts in a way to make them re-applicable for other users. Mediation between two user perspectives resolves to a sequence of backward and forward applications of several such contexts. Although the Macadam system also provides for personalized perspectives, there are fundamental differences to CAIMAN. Macadam requires a common reference catalog to start with and personal catalogs can only be derived from the reference catalog. This prevents a dynamic turnover of members in communities, as all community members would have to begin working with the same catalog from the beginning on. Additionally, there is no clear distinction between private catalogs and community catalogs, which entails privacy issues and is likely to diminish community participation. The Macadam system is more targeted for teams than for communities as it violates additional community requirements, such as integration into everyday work processes.

The *Haystack* system [Adar et al. 1999] is a personal information management system that provides personalized access to information resources. The Haystack system is integrated into the user's personal desktop, observing the user's information management behavior to use the collected information for personalized information access. Haystack allows the integrated management of different types of information like documents, appointments or e-mail. The Haystack system has recently been

Compared Systems						
Feature	KP†	OBIWAN	SiteSeer	M/PD‡	Haystack	CAIMAN
Exchanged Knowledge						
Document topic	√	×	√	√	×	√
Domain structure	×	×	×	×	×	√
Subjective relation	×	×	√	√	×	√
Exchange Peers						
User	√	×	√	√	×	√
Community	√	×	×	√	×	√
Global	√	√	×	×	√	√
Community Design						
Personal Catalog	×	×	√	×	√	√
Work/benefit balance	√	√	√	×	√	√
Privacy	×	√	×	×	√	√
Knowledge Exchange Support						
Knowledge publication	√	×	√	√	×	√
Knowledge awareness	√	×	√	√	√	√
Knowledge creation	√	√	√	√	√	√
Heterogeneous catalogs	√	√	×	×	×	√
Cold start immune	×	√	×	√	×	√
Different Information types	×	×	×	×	√	×

† Knowledge Pump ‡ Macadam / Placeless Documents

TABLE 4.1: Related Work compared to CAIMAN

re-implemented based on a repository of semi-structured RDF[7] data [Huynh et al. 2003]. The flexible data management architecture allows for the creation of personal and dynamically adaptable catalog structures. The focus in [Huynh et al. 2003] is on simple access and management of information through semantically rich user interfaces and a specialized data modification language. As aspects of collaboration have been completely left out and are mentioned as future work for Haystack, CAIMAN and Haystack complement each other.

A summary of the differences between the presented systems and CAIMAN is shown in Table 4.1.

4.6 Summary

In this chapter we have presented the application level design of the CAIMAN framework. CAIMAN supports the exchange of knowledge among community members via document catalogs. CAIMAN differs from other existing concepts in that the CAIMAN functionalities have been systematically based on characteristics of the knowledge exchange process. CAIMAN supports knowledge exchange for the following knowledge types and knowledge exchange building blocks:

- **Knowledge types:** Document contents, document topics, catalog domain structures, subjective document relations.

[7] Resource Description Format

	Knowledge exchange process building block			
Knowledge type	Distribution	Awareness	Creation	Application
Document content	IP	RIR	RIR	RIR
Document topic	IP	RIR	RIR	RIR
Domain structure		CD	CD	CD
Subjective relation	IP	RIR	RIR	RIR

IP: Information publication, CD: Category discovery, RIR: Related information retrieval

TABLE 4.2: Assignment of support services to process blocks and knowledge types

- **Knowledge exchange process steps:** Knowledge distribution, knowledge awareness, knowledge creation and additionally, even though not included in the exchange process, knowledge application.

We introduced the CAIMAN knowledge exchange support services, which support the exchange of the different knowledge types mentioned above. All of the CAIMAN services have been designed for community support, i.e. respect the requirements from Chapter 3. We have introduced the following three knowledge exchange support services:

- The **information publication** service allows the user to publish documents from her personal catalog to other catalogs without additional categorization effort.
- The **related information retrieval** service retrieves related documents from mediated catalogs for a given category in the personal user catalog.
- The **category discovery** service allows the user to discover new relevant categories in mediated catalogs by browsing through her own personal catalog.

A summary of which building blocks of the knowledge exchange process are supported for which types of knowledge by the respective services can be seen in Table 4.2.

The key enabler of the CAIMAN knowledge exchange is a semi-automatic mediation of community-internal and -external document catalogs.

We also presented a first concept for the integration of CAIMAN services into an existing shared bookmark management application for scientific research communities.

The CAIMAN framework concept presented here has a number of advantages over existing related concepts. We reviewed systems that provide for the exchange of knowledge via document catalogs and summarized the key features and differences to CAIMAN in Table 4.1.

Part III

Mediation Infrastructure

Chapter 5

CAIMAN Document Catalog Mediation Principle

5.1 Introduction

In this chapter, we are going to motivate the CAIMAN catalog mediation principle, which is the basis of the knowledge exchange services presented in chapter 4. Mediation in our scenario refers to the virtual integration of document catalogs. We refer to what we present in this chapter as a *mediation principle*, because we describe a mediation approach in an overview fashion, which does not include all details involved in catalog mediation. Based on existing techniques that can be used to calculate the virtual integration of catalogs, we differentiate mediation principles by how the virtual integration is performed:

- **Document granular mediation** performs mediation on a per-document basis, without regard to the categories in which the documents were contained.
- **Category granular mediation** performs mediation on a per-category basis.

We compare both mediation approaches with respect to the knowledge exchange services. We show that a category granular approach has more advantages in our scenario. The, we describe the category granular CAIMAN mediation principle. Among the various aspects of mediation, we focus on a catalog matching approach (see Chapter 6) and a querying approach for heterogeneous catalogs (see Chapter 7) in this work. The mediation principle briefly describes other aspects of mediation, how the different CAIMAN components work together and what role the user plays in the mediation process.

5.1.1 Specification of the mediation problem

As a basis of the services, a way to query the user catalog and the mediated catalogs in an integrated fashion is required. The mediation problem is illustrated in Figure 5.1. The mediated catalogs on the right hand side of Figure 5.1 have a different category structure and are represented with different data models than the user catalog on the left hand side of the figure. Consider, for example, the *related information retrieval* service, invoked on a category c_i. The service needs to retrieve all documents in category c_i from the user catalog and all related documents from the mediated catalogs. First, we have to find out, which documents in the mediated catalogs can be considered related to c_i. Second, the mediated catalogs may be represented with a data model that is not compatible with the query

language and data model of the user catalog and we have to find a way to overcome these differences. Finally, the query results from the different mediated catalogs have to be integrated.

A virtual integration of the involved catalogs, i.e. providing one integrated view on all involved catalogs, allows the knowledge exchange services to jointly query and update[1] all catalogs. Thus, we refer to virtual integration of catalogs here, when we speak of mediation.

5.1.2 Application requirements

The CAIMAN mediation principle has to fulfill the following application requirements of the knowledge exchange services:

- **Knowledge types:** The mediation approach should allow for an exchange of all different types of knowledge that can be exchanged via a catalog: document knowledge, topic knowledge, domain structure knowledge and knowledge about subjective document relations.
- **High quality mediation:** the mediation quality should be high enough to bring an obvious benefit for the user in a semi-automatic mediation process.
- **Dynamic mediation:** the mediation approach should allow for a dynamic mediation environment with changing catalog structures and contents.
- **Large catalogs:** the mediation approach should allow for mediation of large catalogs, even up to the size of existing web catalogs, while keeping user interaction fluent and mediation quality high.

The usefulness of the CAIMAN framework depends on the mediation quality provided by the mediator. This leads to an additional application requirement: to calculate the virtual integration result, i.e. which documents from a mediated catalog are related to which category in the user catalog, we can theoretically use all data that is available in the catalogs. Among these data are category name labels and name labels of relations between categories. We argue that in our application scenario, category and relation labels in catalogs can be highly subjective and thus misleading. This argument is backed by a case study in [Bonifacio et al. 2000]. There, it is suggested that in document catalogs created by different people, category and relation names vary too much to be useful for finding correspondences. In order to maintain a high mediation quality, we add this additional application requirement:

- **No category or relation name labels can be used** for calculating the catalog matchings in the mediator.

5.1.3 Investigated problems

In this chapter, we provide an overview of the CAIMAN mediation principle. We investigate the following sub-problems concerning the mediation principle:

- How can the mediation problem be broken down into smaller, well-defined problems that can be solved by the components of the CAIMAN mediation infrastructure (Section 5.2).
- We investigate whether a document granular or a category granular approach is better suited as mediation approach with respect to the knowledge exchange services in CAIMAN (Section 5.3).

[1] Since updates are symmetric to queries in all cases that are relevant to our scenario, we are not going to differentiate between the two in the rest of this work.

- We want to find a coarse mediation principle that describes how the different mediation components interact with each other, with the knowledge exchange services and with the user to solve the mediation problem in our scenario (Section 5.4).

5.2 Document catalog mediation principle

Mediation in a database scenario

Related work on mediators has mostly been published in the context of heterogeneous databases. *Mediators* have been defined in [Wiederhold 1995] as *"domain-specialized components, which bring source information into a common form"*. In [Garcia-Molina et al. 1995], this definition is picked up and refined to: *"A mediator is a system that refines in some way information from one or more sources and embeds the knowledge that is necessary for the process"*. In [Borghoff and Schlichter 1996], mediators are defined as *"components, the main functionality of which is the selection of the information source, which can satisfy a query"*. As can be seen, the definitions vary, depending on the context, in which mediators are used. What is common to all mediator definitions, however, is the scenario of heterogeneous information sources, the virtual integration of which is achieved by the mediator [Borghoff et al. 1996; J.Bayardo et al. 1997; Garcia-Molina et al. 1995].

The virtual integration of heterogeneous databases can generally be achieved in two phases:

- Preparatory mapping phase:
  - A schema for the integrated database is created.
  - Correspondences between the integrated schema and the schemas of the heterogeneous source databases are identified (**schema matchings**). The process of finding these schema matchings is called **schema matching process**.
  - Views that map the integrated schema to the schemas of the heterogeneous source databases are created. This process is called **schema mapping**.
- Integration phase:
  - Queries over the integrated schema are mapped to queries over the heterogeneous source schemas (**query mapping**) and the query results are integrated by a **query processor**.
  - The individual queries over the heterogeneous source schemas are converted between different query and data representations by **wrappers**.

The schema matching and mapping as well as the query mapping processes are typically integrated in a central **mediator** component along with the query processor. The **conversion** of query and data representations is typically performed by decentralized **wrappers** at the heterogeneous source databases.

Mediation in a catalog scenario

To structure the catalog mediation problem, we transfer the database scenario to our catalog scenario. Just like in the database scenario, catalog mediation is understood in this work as a virtual integration of catalogs under a common catalog structure. Specifically, virtual integration of catalogs in the user catalog means that for a query over the user catalog, the mediator returns all relevant document results from all mediated catalogs.

In a catalog mediation scenario, the common schema is represented by the user catalog structure. The mediator maps queries over the user catalog to queries over the mediated source catalogs and integrates the results. Catalog wrappers convert the different catalog data representations into a representation that is queryable by the mediator. A conceptual overview of mediation in our catalog scenario can be seen in Figure 5.1.

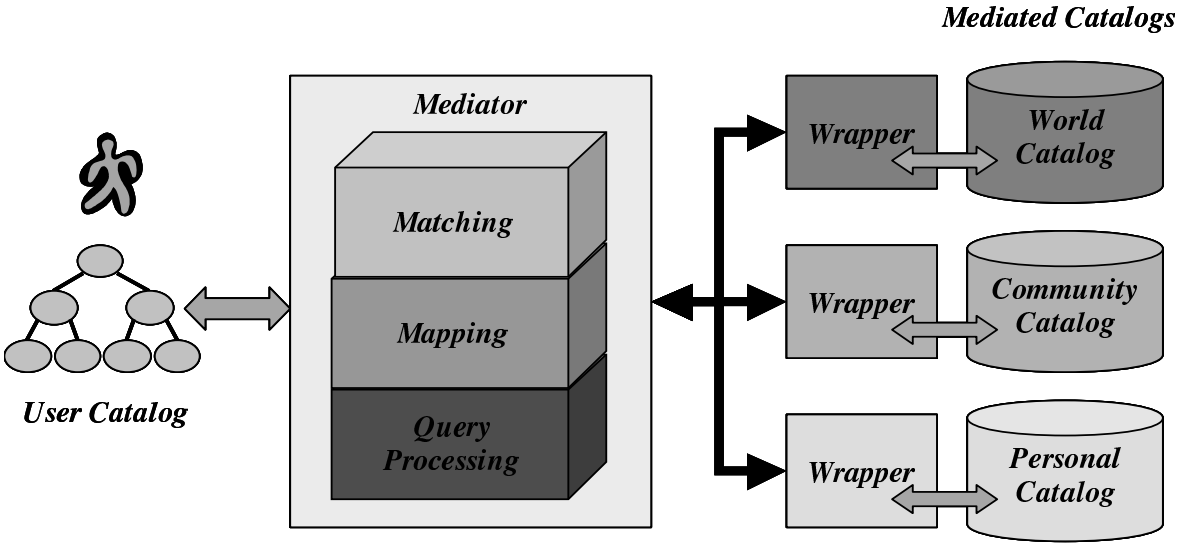


FIGURE 5.1: Mediation scenario for heterogeneous document catalog sources

In the left part of Figure 5.1, we see the user document catalog, under which the virtual integration of the different heterogeneous source catalogs is performed. Mediation in our catalog scenario has the same phases as in the database scenario:

- In the **preparatory mapping phase**, a **catalog matching** is calculated by a matching component. The catalog matching is used to prepare the **catalog mapping** information that is later required to map queries over the user catalog to queries over the mediated catalogs.
- In the **integration phase**, a query that is posed over the user catalog is sent to the mediator. The mediator performs a **query mapping** of the query to a new set of queries that are posed over the mediated catalogs. Since the different mediated catalogs may additionally have different query and data representation languages, the conversion between the different representations is performed by **wrappers**.

Thus, to make catalog mediation feasible in our scenario, we need to specify the following components and how they work together:

- A **matching** component, which calculates a catalog matching as the basis of virtual catalog integration.
- A **mapping** component, which prepares the required catalog mapping information and maps the queries posed over the user catalog to a set of queries over the other mediated catalogs.
- A **query processing** component, which can process catalog queries and integrate the results.
- **Wrapper** components, which can provide access to document catalogs in a way that makes them queryable by the query processing component.

5.3 Comparison of mediation approaches

5.3.1 Document granular mediation

Document granular mediation is an atypical mediation approach, because the categories of the different catalog sources are not matched. We have included document granular mediation here, because a document granular approach can be realized in a straightforward way using existing techniques for automated text classification.

Mediation principle. Document granular mediation approaches take advantage of the fact that the documents themselves contain a lot of data that can be used to take a shortcut approach to mediation. Instead of matching categories and creating mapping information according to the catalog matchings, we can simply take an individual document from a mediated catalog and categorize it over the user catalog. The categorization result is the mediation result for this specific document, which is calculated independently of the other documents from the same community category. We call this a document granular approach, because the integration decision is taken for each individual document, independent of the categories. A document granular mediation approach still requires all components of the mediation infrastructure, but the mediation process for finding all documents related to a category c_i is somewhat different from the approach suggested in Section 5.2:

1. In the **preparatory mapping phase**, the mediation result can be pre-computed by retrieving and categorizing all documents and storing the results locally as document meta-data.

2. In the **integration phase**, all documents that have been categorized into category c_i have to be retrieved from all mediated catalogs. If the mediation result has not been pre-computed, the complete set of documents from all catalogs has to be retrieved for categorization.

Existing approaches. There are many existing text classification techniques that can be used for document granular mediation in our scenario (see [Sebastiani 2002] for an overview). For **automatic text classification, a text classifier needs to be trained** on the destination catalog and can subsequently classify previously unseen documents from a source catalog into the "right" category in the destination catalog[2]. Most published text classification results consider classification into a flat list of classes, without any hierarchical structure. Among the most popular text classification techniques are Support Vector Machines [Joachims 1997] and the Naive Bayes approach [Lewis 1998]. Very good classification results have been reported for hierarchical classification in [Koller and Sahami 1997]. However, these results applied for a very small catalog with only 10 very distinct classes.

In [Agrawal and Srikant 2001], another document granular mediation approach has been presented. The approach in [Agrawal and Srikant 2001] uses an adapted Naive Bayes text classification technique, which takes documents in the same source category into account for classification. The adapted Naive Bayes classifier is used to achieve a very high classification accuracy on synthetically created catalogs. The catalogs that are mediated are identical to each other with the difference that in one catalog some random noise documents are added. For the integration of existing large catalogs on the Web, the achieved accuracy has been significantly lower than for the synthetic catalogs. Still, a significant improvement compared to generic Naive Bayes classification could be achieved in all cases. However, for the experiments in [Agrawal and Srikant 2001], no internal document categories in the catalog tree have been considered and instead all documents have been joined into leaf categories. This results in a somewhat unrealistic distribution of documents in categories.

[2] See Section 6.3 for details on how automatic text classification works.

High accuracy. The performance results for classification techniques that can be used in document granular mediation indicate a high classification accuracy (see [Sebastiani 2002] for an overview). However, the techniques have often been tailored to the special kinds of catalogs, which have been used for evaluation. Performance results for real world catalogs look a lot less encouraging. We have shown that classification results on real world catalogs, such as for example the Researchindex[3] collection of research papers, are not sufficient for an effective semi-automatic mediation approach [Lacher and Groh 2001].

Structural independence. Since no category matching is performed in document granular mediation, the document matching accuracy is not susceptible to the existence of structural differences between the mediated catalogs. For a document granular approach, the granularity of the different catalogs, i.e. how many categories there are and how the documents are distributed among them, is virtually irrelevant[4].

Asymmetry. Document granular mediation calculates a mediation result for each individual document of a catalog, which makes the mediation result inherently asymmetric. If we consider, for example, the related information retrieval service (see Section 4.3.2), we see that for each individual document from all catalogs that function as sources of the retrieval process, the "right" destination category in the user catalog has to be calculated. This mediation solution for the retrieval service is not of any help for the publication service, though. For the publication service, we need to calculate the "right" categories in all destination catalogs for all user documents, independently of the previous calculations. Thus, for document granular catalog mediation, one mediation solution needs to be calculated for every catalog that serves as a destination catalog of one of the knowledge exchange services.

5.3.1.1 Information publication service

To start the mediation process, the user chooses a set of catalogs with which documents should be exchanged. The information publication service then publishes those documents from each user category c_i, for which the user allows publication.

Costly classifier training. Using a document granular mediation approach for document publication, one classifier has to be trained for each of the destination catalogs. One possibility to train the classifiers for the destination catalogs, would be to transfer all documents from all destination catalogs to the user's workstation and train the classifiers there. This option can be ruled out, because the amounts of data to transfer would be too large and the approach would not remain scalable. The other option is to assume that the classifiers are trained at the locations of the involved destination catalogs. On the one hand, this may be a realistic assumption, because the destination catalogs benefit from the publication service. On the other hand, not all hosts of community or global catalogs may be willing to spare processing time for classifier training and classification for all users accessing the catalog. Hence, all knowledge exchange services, for which the destination catalog is a non-cooperative catalog, can not be realized with a document granular mediation approach.

[3] See http://www.researchindex.com/

[4] With the exception of *context aware classification* (see Section 6.5.3.1).

High user workload. We proposed in Chapter 4 that all knowledge exchange services should be designed to perform a semi-automatic mediation, allowing the user to correct or reject recommendations of the services. With document granular mediation, the correction of the mediation results has to be performed for each individual document and may thus incur significant workload for the user. However, this problem may not be significant in the publication case, as the user is likely to have little interest in correcting the destinations of documents in other catalogs.

Loss of subjective relations among documents from the same source category is another issue with document granular retrieval. The creator of a catalog puts a set of documents into the same category because she sees a relation between the documents. This subjective relation may not be reflected in the document text. Document granular approaches cannot transfer this relation to another catalog, thus this potentially valuable piece of knowledge is lost in the mediation process (see example in Figure 5.2).

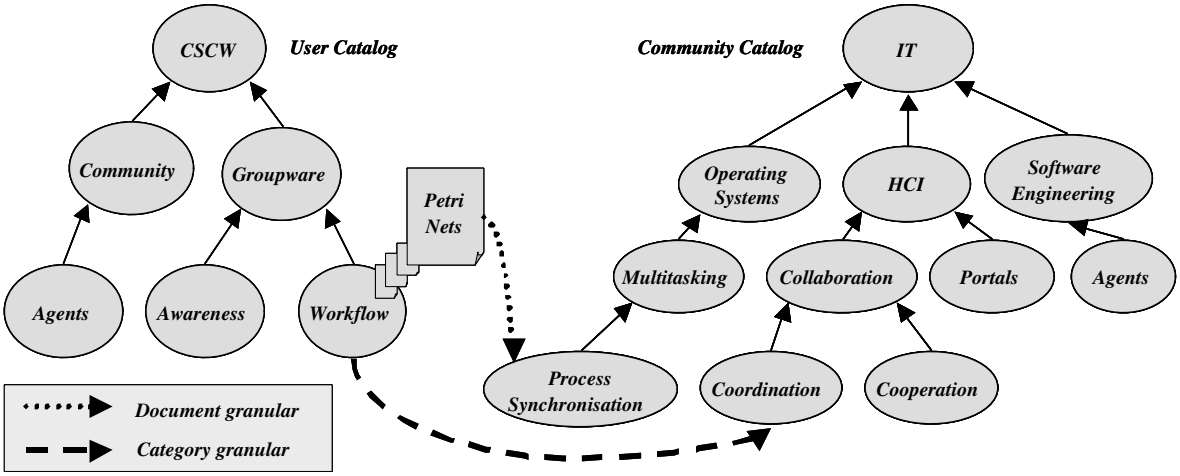


FIGURE 5.2: Subjective relations among documents and mediation result with document granular vs. category granular matching

Figure 5.2 shows an example for the loss of subjective relations with document granular catalog mediation. The user catalog on the left hand side conceptualizes the domain of *Computer Supported Cooperative Work* (CSCW). The user knows that *Petri Nets* provide a useful theoretical background for *Workflow* systems. Consequently, she categorizes a document about Petri Nets in the *Workflow* category. The relation between the document about *Petri Nets* and the rest of the documents about *Workflow* systems is what we call a subjective relation. As the subjectively related documents have different topics, the relation cannot be seen from the texts alone.

Now we consider a scenario, in which the two catalogs in Figure 5.2 are mediated and the information publication service publishes new documents from the user catalog to the community catalog. The document about Petri Nets has just been newly added by the user and is thus subject to publication. A document granular matching approach will take the individual document and classify it into the community catalog, using text classification techniques, which rely on document term statistics. As can be seen in Figure 5.2, the *Petri Nets* document will most probably be assigned to the *Process Synchronization* category, which holds other documents about *Petri Nets* and the subjective relation is lost. A category granular matching approach would instead match the whole *Workflow* category

in the user catalog to the *Coordination* category in the community catalog. Along with the category matching, all documents from the *Workflow* category would be assigned to the *Coordination* category and the subjective relation could be conserved as well in the community catalog.

Cold start problems. The phase, in which a catalog has just been created and starts to get filled with documents is called cold start phase[5] of the catalog. If one of the destination catalogs of the publication service is in its cold start phase, the catalog can be assumed to have an insufficient number of documents for text classifier training. Thus, the publication service will likely deliver low quality results and make little sense. A classification confidence threshold may increase the service precision at the cost of its recall. If the source catalog instead of the destination catalog is in a cold start phase, the results quality will not suffer.

Another constraint is that a document must not be assigned to multiple categories within one catalog perspective. Thus, documents that already exist in a destination catalog are removed from the set of matching documents.

5.3.1.2 Related information retrieval service

We assume the user chooses a category c_i in the user catalog. The related information retrieval service then recommends documents from different remote source catalogs that are related to the documents in category c_i.

Using a document granular mediation approach, only one classifier has to be trained for the user catalog. The decision whether a document should belong to c_i or not, can only be taken locally by the trained classifier. Thus, all documents from all of the involved remote source catalogs have to be transferred to the user's workstation first. Only then can they be checked for relevance to the category c_i by the classifier. This may cause large network and processing load at the user's workstation.

The correction of recommendations of the information retrieval service is of much more importance for the user than for the publication service. However, with document granular mediation, the correction of the mediation results has to be performed for each individual document and may thus incur significant workload for the user.

Documents are retrieved independently of their categories. Thus, just like for the publication service, the subjective relations of documents are lost in document granular retrieval (see example in Figure 5.2).

In the user catalog cold start phase, the catalog can be assumed to have an insufficient number of documents for text classifier training. This leads to the same problem as for the publication service. Again, only a classification confidence threshold may increase the service precision a limited amount at the cost of its recall.

To avoid double classification, only documents that are not already categorized in the user catalog are retrieved.

5.3.1.3 Category discovery service

The category discovery service can be realized in two different ways: either the documents retrieved by the *related retrieval* service are taken as indicators for category matches, or the category matches are calculated the same way as in a category granular approach. If the first option is adopted, the category discovery service inherits the problems of the related retrieval service introduced above.

[5]The term *cold start* is often used in groupware and communityware in a collaborative filtering context.

Moreover, the only way to find related categories is to consider all those categories in mediated catalogs *related* that are source categories of documents retrieved by the related retrieval service. Given that one document is enough to establish a relation between two categories, this document granular approach is very error-prone in addition to the disadvantages incurred by using the *related retrieval* service.

5.3.2 Category granular mediation

Category granular mediation is a mediation approach that is typically used with databases and is reflected in our motivating problem scenario in Figure 5.1. We consider category granular mediation especially interesting, because it has a number of advantages over document granular mediation and successful applicable matching techniques have been published in related fields.

Mediation principle. In a category granular mediation approach, categories are treated as atomic, indivisible entities and catalog integration is performed on a per-category basis. Thus, either all documents from a source category are integrated into the destination category, or none. A catalog matching of the user catalog with each mediated catalog is required as the basis of category granular mediation.

The goal of the matching process here is to find matching categories in mediated catalogs for every category c_i in the user catalog. Text classification techniques can be used in catalog matching approaches for category granular mediation. The classification results can be used as an indicator for the similarity of categories. The CAIMAN matching approach uses text classification techniques (see Section 6), thus in the following comparison, we assume a classification based approach as well.

Existing catalog matching techniques that can be used for category granular mediation are essentially solutions to concrete instances of an abstract problem class: the problem of matching two conceptual structures. Conceptual structures consist of the definition of abstract classes and their relations on the one hand and instances of the defined classes on the other hand. In document catalogs, the categories represent classes and documents represent their instances. Another example for conceptual structures other than document catalogs are databases, where the schema entities represent the classes and data tuples represent instances. Furthermore, *ontologies* [Gruber 1992], which are also a more general form of conceptual structures. A number of matching approaches have been presented for databases [Rahm and Bernstein 2001] and ontologies [Doan et al. 2002]. However, most of them use category name labels and name labels of relations between categories and instances, to establish a catalog matching. As name labels cannot be used in our scenario, because of their subjective nature, a large fraction of the existing matching approaches can not be directly applied in our scenario.

5.3.2.1 General characteristics

Symmetry. Matching techniques that are used in category granular mediation estimate the similarity between categories. The notion of similarity of two categories is inherently symmetric: we can generally say that if category c_i is similar to category c_j, then category c_j is also similar to category c_i[6]. Category similarities are calculated based on text classification techniques and training of only one classifier can be sufficient due to matching symmetry. However, due to the requirements of the knowledge exchange services, a matching may still become asymmetric. Consider, for example, a user category c_i, for which the publication and the related retrieval service are started. For the related

[6]There are exceptions to this assumption, but we claim that in our scenario, this is a realistic assumption

retrieval service, c_i may have several matching categories in one mediated catalog, from which documents are retrieved. For the publication service, however, there can only be one matching category, because any document can only be in one category in a catalog perspective. We can see that training one classifier for all catalogs can be enough, but the matchings have to be calculated individually for each service.

Category subset transfer. In a category granular mediation approach, once a category match is established, we know the mediation solution for all contained documents. Unlike in the document granular case, not all documents have to be transferred before the mediation solution can be established. However, depending on the specific matching approach, the transfer of some documents from remote catalogs to the user's workstation may be required, in order to establish a category match.

Choice of classifier training catalog. There are two options for training the required classifier in a category granular approach. Training the classifier on the community catalog or other mediated catalogs would incur the same potential problems with non-cooperative catalogs, as presented for document granular approaches. These problems can be avoided by training the classifier on the user catalog. However, classifier training requires a minimum amount of training documents to work correctly and may thus lead to cold start problems.

Structural dependency. In contrast to document granular mediation, different catalog granularities do have a significant influence on category granular approaches. If two catalogs with different category granularities are matched, the categories will not match exactly. Depending on the direction in which the knowledge exchange service exchange documents, different granularities have more or less influence as we will see in the discussion of the services. Unlike in a document granular mediation approach, the mediation result has to be recalculated, if new categories are created in a catalog, due to the dependency of category granular matching on the catalog structure.

Conservation of subjective relations. As the documents in one category are always kept together, when using a category granular mediation approach, subjective relations among documents are conserved and can be transferred by the knowledge exchange services.

Low user workload. The user workload for correcting mediation results is not as big a problem as for the document granular case: the corrections can be made on a per-category basis instead of on a per-document basis, which reduces the user workload significantly.

5.3.2.2 Service-specific characteristics

Information publication service. After the catalogs for mediation have been chosen, the information publication service publishes those documents from each user category c_i, for which the user allows publication.

We have mentioned that different catalog granularities have a strong influence on a category granular mediation approach. If the user catalog has a finer granularity, several user categories will likely be matched to an individual destination category, which is a correct solution. However, in case the destination catalog has a finer granularity, a user category will possibly match with several destination categories, among which one final match has to be picked. Whichever category is picked among the match candidates, the choice will be wrong for a significant fraction of the documents in the user

category. This disadvantage cannot be avoided and has to be taken into account for catalogs with granularity differences.

Related information retrieval service. The related information retrieval service for a chosen user category c_i recommends documents from different remote source catalogs that are related to the documents in category c_i.

Again, granularity differences in the catalogs have a strong influence here. If the user catalog has a finer granularity, several user categories will possibly match with one source category and one final match has to be picked. Again, independent of the category that is chosen, the choice will be wrong for a significant fraction of the documents in the source category. However, in the case that the source catalog has a finer granularity, a user category will likely match with several source categories, which is a correct solution.

Category discovery service. A category granular mediation approach can take advantage of a more accurate category matching for the *category discovery* service: the category matches are not based on single documents, as in the document granular case.

5.3.3 Summary and discussion

We have illustrated the advantages and disadvantages of document granular mediation with respect to the requirements of the knowledge exchange services.

- **Document granular mediation** has the following **advantages**:
  - A high accuracy has been shown for classification techniques that can be used for document granular catalog integration. However, these results have been achieved on small or special purpose catalogs.
  - The mediation accuracy is not susceptible to structural differences and different granularities of mediated catalogs.
- **Document granular mediation** also incurs the following **disadvantages**:
  - One classifier has to be trained for each involved catalog. This can impede knowledge exchange with non-cooperative community or global catalogs.
  - Significant network and processing load is incurred by the necessary complete transfer of the source catalog of a service to the destination.
  - Significant workload for the user is incurred in the semi-automatic mediation process.
  - Subjective relations among documents are lost.
  - If the destination catalog of a service is in a cold start phase, the mediation quality will be low. However, if the source catalog of a service is in a cold start phase, the mediation quality will not suffer.

With regard to the application requirements, we can say the following regarding **document granular mediation**:

- **Knowledge types:** Only document content knowledge is exchanged; topic knowledge and subjective document relations are lost. Exchanged domain structure knowledge is likely erroneous.

- **High quality mediation:** The accuracy of integration on certain catalogs has been demonstrated to be high enough to bring an obvious benefit for the user.
- **Dynamic mediation:** Structure and granularity independence allow for dynamic mediation. However, the required complete transfers of large catalogs constrain dynamic mediation.
- **Large catalogs:** Insufficient scalability with large catalogs due to the required complete transfers of large catalogs.

We have further illustrated the advantages and disadvantages of category granular mediation with respect to the requirements of the knowledge exchange services.

- **Category granular mediation** has the following **advantages**:
  - For a matching approach that is based on text classification, only one text classifier for the user catalog has to be trained.
  - Only a fraction of the documents in each category of the mediated catalogs needs to be transferred to the user catalog to establish the matching.
  - The workload for the user to correct or adjust matchings is low.
  - Subjective relations among documents are conserved.
  - The cold-start problem can be avoided by switching the text classification training catalog.
- **Category granular mediation** also incurs the following **disadvantages**:
  - The matching accuracy is very susceptible to differences in the catalog structures. Assigning documents from a coarse granular catalog to a finer granular catalog will likely result in errors.

With regard to the application requirements, we can say the following regarding **category granular mediation**:

- **Knowledge types:** All four types of knowledge can be exchanged.
- **High quality mediation:** There are no published evaluations for existing category granular catalog integration approaches that are applicable to our scenario.
- **Dynamic mediation:** Structure and granularity dependence can hinder dynamic mediation. However, no large catalog transfers are required for matching.
- **Large catalogs:** Good scalability with large catalogs due to the fact that only catalog fractions need to be transferred and user interaction effort can be kept small.

Community membership cold start. None of the two presented approaches are influenced by a community membership cold start situation, i.e. if there are not many members in a community yet. There may be fewer documents to exchange between community members, but the quality of the knowledge exchange depends on the quality of the mediation and not on the number of participating community members.

Influence of domain overlap. We have assumed, that both document granular and category granular approaches are based on techniques for automatic text classification. As text classification techniques are in turn based on term statistics, a text-classifier-based mediation approach will deliver more accurate results with increasing overlap of the domains of the mediated catalogs. To show what mediation quality can be expected, we consider examine the situations that a user catalog is mediated with:

- **a catalog in the same community.** As members of a community of interest share a common interest by definition, we can arguably assume that within a community, the catalogs have sufficient domain overlap to achieve a high mediation quality.
- **a world catalog.** By definition, a world catalog has a large domains of interest, and thus it is likely to have some domain overlap with the user catalog. However, the catalog granularities and vocabulary may differ and thus a medium mediation quality can be expected.
- **a catalog in a different community.** Catalogs from other communities are likely to have little domain overlap with the user catalog. Even if there is overlap, granularity differences are likely as well. The expected mediation quality is going to be lower than for the previous two cases.

An overview of the expected mediation quality for different catalogs can be seen in Figure 5.3. Experimental evidence that our assumptions about mediation quality are correct has been presented in [Takeda et al. 2000].

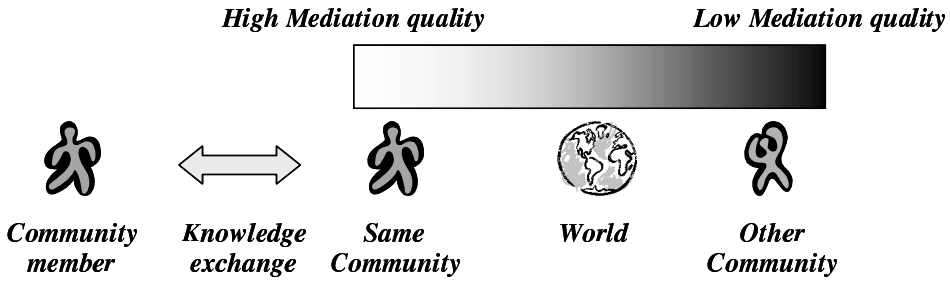


FIGURE 5.3: Mediation quality depending on the catalog used for mediation

Conclusion. The only existing approach that is directly applicable to our scenario is a document granular mediation approach based on text classification. However, we have shown that a document granular approach does not fulfill a number of the application requirements of the knowledge exchange services. Thus, we consider a document granular approach not suitable for our purposes.

The category granular approach fulfills the application requirements much better than a document granular approach. Thus, the CAIMAN mediation infrastructure uses a category granular mediation approach.

5.4 Overview of the CAIMAN mediation approach

The CAIMAN framework follows a category granular mediation approach that employs text classification techniques to establish catalog matchings. For the calculation of catalog matchings, no category or relation name labels are used. Other aspects of mediation can for example be found in [Garcia-Molina et al. 1995; Borghoff et al. 1996; J.Bayardo et al. 1997].

5.4.1 Preparatory mapping phase

Catalog matching. Catalog matches are calculated by the CAIMAN matching component. The CAIMAN matching approach is based on text classification and does not use any catalog structure name labels for the matching calculation.

As we have described in our catalog model in Section 2.3.3.1, a catalog can have several perspectives under which documents can be categorized. It makes little sense to try and match two perspectives that represent different categorization criteria in the mediation process. In CAIMAN, the user has to manually pick the catalog perspectives that should be matched.

One text classifier is trained on the user catalog and a subset of the documents in each category of each mediated catalog is transferred to the user, such that the classifier can classify them with respect to the user catalog. The classification results serve as the input for the category matching calculations. The final catalog matching identifies for each category in the user catalog a set of matching categories in the respective mediated catalog. Depending on the knowledge exchange service, for which the matching is calculated, there may be more than one matching category. Consequently, one matching has to be calculated for each knowledge exchange service. The CAIMAN matching approach will be described in more detail in Chapter 6.

Training the text classifier on the user catalog only can lead to a cold start problem. We avoid the cold start problem by temporarily delegating the classifier training to the mediated catalog sites, if they are willing to perform the training. As soon as the user catalog cold start phase is over, the classifier training is performed at the user catalog site again. A disadvantage of swapping the training catalogs is that the matching is calculated based on classifiers trained on a community catalog, and thus resembles the perspective of the community instead of the user's perspective. However, this disadvantage is by far outweighed by the advantage that swapping makes a matching possible that could otherwise not be calculated or would be erroneous.

Catalog and query mapping. In order to be able to prepare the mapping information that is required for later query mapping, the user has to supply some information about the involved catalogs.

In our catalog model in Section 2.3.3.1, we have defined that categories are linked to their sub-categories by means of *sub-category* relations. However, in different real life catalogs, the *sub-category* relations may have different name labels, such as *sub-class*, for example. In order to able to query the mediated catalogs, the mediator needs to know which label the sub-category relation has in the mediated catalogs. Moreover, the mapping component needs to know how documents are assigned to categories in the respective data models. This information has to be supplied by the user as well.

Preparing the mapping information for categories is straightforward in our case. A category is simply mapped to its matching categories according to the calculated catalog matching.

Finally, using the collected mapping information, rules for the mapping of queries have to be generated. In the CAIMAN mediation principle, we consider this mapping generation in a simple way that serves as a proof of concept only.

5.4.2 Integration phase

Knowledge exchange takes place in the integration phase. If the user invokes one of the knowledge exchange services, the respective queries over the user catalog are mapped to queries over the mediated catalogs using the mapping information gathered in the preparatory mapping phase. The mapping of queries and the integration of the query results is performed by the CAIMAN query processing component using pre-defined mapping rules. The generation of mapping rules and query mapping

is discussed in this work in an overview fashion. We provide a simple proof-of-concept solution for query mapping, but a full-blown query mapping solution is not within the focus of this work. The mapping of queries and related issues have been extensively discussed in [Halevy 2001].

All queries and query results are represented in RDF[7] in CAIMAN. However, the various mediated catalogs may be represented in other data formats that cannot be handled by the CAIMAN query processing component. A wrapper converts between the CAIMAN-internal data format RDF and other data formats. The query processing and wrapper approaches are described in more detail in Chapter 7.

After the query has been processed, the integrated results are presented to the user by one of the knowledge exchange services. The user can now manually accept, correct or reject the recommendations made by the knowledge exchange services.

5.5 Summary

The problem that has to be solved by a mediator in our scenario is the problem of virtual integration of document catalogs. We transferred a database mediation principle to our scenario and identified the necessary components of a mediator: a *matching* component, a *mapping* component and a *query processing* component. Additionally, various *wrapper* components are required to convert between source catalog data formats. In addition to the application requirements from Chapter 4, no name labels of the catalog structure can be used for the calculation of matchings in the mediation process.

We have shown that catalog mediation can be realized with two different matching approaches: document granular matching and category granular matching. Document granular mediation can be realized with state of the art text classification techniques. A large number of matching techniques for database schemas and ontologies have been published. Although principally applicable to catalog matching, most of these approaches use name labels for matching. We compared the two general approaches with respect to the knowledge exchange services and found arguments in favor of each approach:

- **Document granular:** high mediation quality can be assumed, catalog structure differences are irrelevant.
- **Category granular:** only one text classifier to train, fewer document transfers over the network, less user workload, subjective document relations preserved, cold start problem can be completely avoided.

In terms of the overall fulfillment of the application requirements, a category granular mediation approach is superior to a document granular approach.

We introduced a coarse category granular mediation principle. We have described the consecutive steps of the mediation process and which roles the different mediation components as well as the user play in the process. Among the different aspects involved in catalog mediation, we focus on matching and a querying principle. Query mapping, however, is not the focus of this work.

[7] Resource Description Format

Chapter 6

CAIMAN Document Catalog Matching Approach

6.1 Introduction

In this chapter, we present the novel CAIMAN matching approach for document catalogs, as required for category granular catalog mediation. The matching approach is the basis for a matching component in CAIMAN, which is one of the components required for the solution of the mediation problem introduced in Chapter 5. We begin with a specification of the catalog matching problem that our matching approach solves. The catalog matching problem is closely related to the schema matching problem in the database field as well as the ontology matching problem in the AI field. We review existing approaches in those fields and their applicability to the catalog matching problem in Section 6.2. Since our catalog matching approach is based on techniques for automated text classification, we introduce the necessary theoretical background in Section 6.3. In Section 6.4, we give an overview of the principles and different consecutive phases of the CAIMAN catalog matching approach. The individual phases of the matching approach are the document categorization phase (Section 6.5), the category correlation phase (Section 6.6) and the structural matching phase (Section 6.7).

6.1.1 Specification of the catalog matching problem

We have described in Chapter 5 that category granular document catalog mediation requires a catalog matching component. We have informally defined the catalog matching process in Section 5.2, which we are going to formalize here. In the process of catalog matching, we calculate, which categories from the user catalog Γ^U match with which categories from the community catalog Γ^C in the sense that they can be considered to represent similar topics. A document catalog has been formally defined in Section 2.3.3.1. The outcome of the matching process is called a *matching*[1]. For the rest of this section, we assume the community catalog to be the matching partner catalog of the user catalog.

Based on the catalog matching, the knowledge exchange services transfer documents from a source catalog to a destination catalog. Figure 6.1 shows an overview of the relation between the mediated catalogs, the matching process and the catalog matching as the outcome of the matching process.

[1]Our definition of a *matching* is different from definitions in existing graph matching literature. However, this choice of terminology was a necessity to avoid other more confusing terminology overlaps.

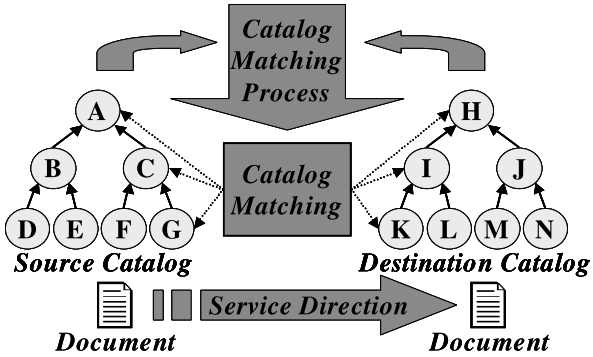


FIGURE 6.1: Catalog matching process in CAIMAN.

Figure 6.1 shows that the result of the *catalog matching process* is a *catalog matching*, which consists of matching pairs of related categories from the matched catalogs. The catalog matching process calculates for every category from the user catalog a set of best-matching categories in the community catalog. The service direction is the direction in which documents are transferred between matching categories by the respective service. Note that, depending on the service (see Section 4.3), either the user catalog can take on the role of the source catalog and the community catalog can take on the role of the destination catalog or vice versa. The calculated catalog matching is also service-specific, as we are going to explain later in this section. To formalize what a category matching is, we need to define a category similarity measure:

Definition 6.1 (Category similarity function) *A category similarity function $\sigma : \mathcal{C}^U \times \mathcal{C}^C \rightarrow [0,1]$ is a measure for the similarity of category pairs (c^U, c^C) with $\mathcal{C}^U$ the set of user categories and $\mathcal{C}^C$ the set of community categories.*

Now we can define a catalog matching generally as:

Definition 6.2 (Catalog matching) *A catalog matching M is defined as $M \subseteq \mathcal{C}^U \times \mathcal{C}^C$, such that for all $m_i \in M$, the value of $\sigma(m_i)$ is defined and can be calculated.*

We have mentioned above that the characteristics of the category matching depend on the respective knowledge exchange service, for which the matching has been calculated. The characteristics of a matching are enforced by a service-specific category match filter function. The service-specific characteristics are expressed as constraints of the following types:

- Cardinality constraints on the number of matching partner categories for a category.
- Minimum similarity constraints for a match pair.
- The Stable Marriage constraint, which enforces that the match pairs are picked in a globally optimal[2]way.

The constraints are explained in more detail in Section 6.7.3. We are first going to define the filter function in a general way.

[2]There can be several valid notions of optimality for this matching problem, and Stable Marriage optimality is just the one we use here.

Definition 6.3 (Category match filter function) *A category match filter function* $\mathcal{F}^K : \mathcal{P}(\mathcal{C}^U \times \mathcal{C}^C) \rightarrow [T, F]$ *for a knowledge exchange service* K *and a catalog matching* M *has a value of* $\mathcal{F}(M) = T$ *if the constraints of the service* K *hold for* $M \in \mathcal{P}(\mathcal{C}^U \times \mathcal{C}^C)$ *and a value of* $\mathcal{F}(M) = F$ *if the service constraints do not hold.*

$\mathcal{P}(\mathcal{C}^U \times \mathcal{C}^C)$ symbolizes the power set of $\mathcal{C}^U \times \mathcal{C}^C$. Using the category match filter function, a final catalog matching for a specific knowledge exchange service can be picked from the set of all possible matchings. Thus, we define the final catalog matching as:

Definition 6.4 (Final Catalog Matching) *Given two catalogs,* Γ^U *and* Γ^C*, a similarity function* σ *and a match filter function* $\mathcal{F}^K$ *for a knowledge exchange service* K*, a final catalog matching for the service* K *is defined as a best effort matching* $M \subseteq \mathcal{C}^U \times \mathcal{C}^C$*, for which* $\mathcal{F}^K(M) = T$ *and the following holds:*

$$\forall m_i \in M, \quad \forall m_j \in (\mathcal{C}^U \times \mathcal{C}^C) \setminus M : \qquad \sigma(m_i) \geq \sigma(m_j) \tag{6.1}$$

Definition 6.4 basically says that a final matching is a matching for which all the service-specific constraints hold and that includes the category match pairs $m_i = (c_i^U, c_i^C) : c_i^U \in \mathcal{C}^U, c_i^C \in \mathcal{C}^C$ with the highest similarity values $\sigma(m_i)$ among all category match pairs. The filter function takes care that the catalog matching makes sense for the respective service and that no constraints of the catalog are violated. For the publication service, for example, a catalog matching with several destination categories for one source category would violate the constraint that a document must be uniquely classified in a catalog. We are going to explain the service constraints in more detail in Section 6.7.3.

Categories, that are not included in a match pair of the final catalog matching are also not included in the document transfer that is initiated by a knowledge exchange service.

The CAIMAN matching approach is concerned with estimating the similarity function σ in a way a human would do it and picking the right category matching for a knowledge exchange service.

6.1.2 Optimal catalog characteristics

Theoretically, the CAIMAN matching technique can be applied to all kinds of document catalogs. However, due to the fact that we use text classification techniques and that we treat categories as an atomic entity, the CAIMAN approach works best on catalogs with certain characteristics. All other related work referenced in Section 6.2 relies on the same catalog characteristics.

- **One perspective**: the matching approach works best, if each of the involved catalogs consists of one single categorization perspective. To ensure this, we let the user manually choose perspectives for mediation.
- **Same perspective**: the matching approach works best, if each of the involved catalogs is categorized along the same perspective.
- **Similar level of detail**: the matching approach works best, if the involved catalogs exhibit a similar category granularity.
- **Domain overlap**: the domains of the involved document catalogs should exhibit a significant amount of overlap, otherwise matching makes no sense.
- **Similar vocabulary**: the vocabulary used in the document collections in different catalogs should have a common subset in order to make statistical text classification techniques applicable.

- **Language**: All documents in the catalogs have to be written in the same language.

The CAIMAN matching approach is still applicable to catalogs, which do not adhere to the above characteristics. However, the matching correctness is likely to deteriorate in this case.

6.2 Existing matching techniques

Catalog matching is a special case of matching of conceptual structures. The matching of conceptual structures plays an important role in the database field as well as in AI. In the database field, elaborate solutions have been published for the matching and integration of relational database schemas or other structured or semi-structured data sources. We are going to give an overview of existing approaches to schema matching in Section 6.2.2. In AI, an equally large number of solutions for *ontology matching* have been published. Ontologies are formal conceptual structures, which often have a formal logic definition [Gruber 1992]. We are going to present an overview of existing ontology matching approaches in the following section.

6.2.1 Ontology matching

Ontologies are defined as a *"formal specification of a shared conceptualization"* in [Gruber 1993]. This very general definition allows almost every formal conceptual structure to be understood as an ontology. Consequently, there are many different applications and application-specific definitions for ontologies. Further details on different definitions of ontologies can be found in [Gruber 1992; Noy and Hafner 1997; Chandrasekaran et al. 1999; Clark 1999; Staab et al. 2000c; Noy and McGuiness 2001]. Ontology based applications are described, among others, in [Huhns and Singh 1997; J.Bayardo et al. 1997; Fensel et al. 1999; Huhns and Stephens 1999; Pretschner and Gauch 1999; Staab et al. 2000b]. For our purposes here, we consider ontologies to be definitions of classes, which resemble categories in a document catalog, and their relations to each other. Ontologies may include instances, which resemble documents in a document catalog, or not. A more formal analysis of the connection of ontologies and document catalogs has been presented in [Welty 1998].

Ontology matching approaches are largely based on name labels of classes. The reason for this is that not all ontologies necessarily have instances and the matching process still has to generate a result. An ontology without instances would resemble an empty catalog structure without documents in the categories. Since class labels in typical ontologies mostly consist of one word, the choice of which is highly subjective, it is hard to generate accurate matchings. Class label based matchings are often subject to low accuracy.

The SMART [Noy and Musen 1999] and PROMPT [Noy and Musen 2000] systems are based on class names and use a simple structural matcher which propagates matches in the ontology graph.

The objective of FCA-Merge [Stumme and Maedche 2001] is to merge two ontologies. A prerequisite of FCA-Merge is the availability of a set of documents that are known to be relevant to both source ontologies. FCA-Merge extracts instances of concepts, which are common to both source ontologies, from the set of documents. Based on the common instances, a merged ontology is generated. It is assumed that it is already known which documents are relevant to both source ontologies. In the case of catalog matching, it is the objective to find this set of documents.

In [Mitra et al. 2000], the *ONION* toolkit for graph-oriented ontology matching is presented. In ONION, the formal logic definitions of concepts are used to express concepts in one ontology through the use of concepts in another ontology. The construct that defines the relations among the two source ontologies is called an *ontology articulation*. The articulation is created in a process,

in which a domain expert plays the most important role, but may also be supported by automated matching techniques [Mitra and Wiederhold 2002]. The automated matching is generated by concept label based techniques and simple structural matching techniques. Moreover, a corpus of documents is collected for each source ontology. The corpus is then used to attempt to extract descriptions of the concepts in an ontology from the corpus, using the concept labels. The descriptions are matched to retrieve a basic similarity measure. In our scenario, we cannot use these label based matching techniques, since label based matching has been excluded for reasons we have stated in Section 5.1.2.

Other ontology-based approaches for the integration of heterogeneous information sources in general are presented in [Wache et al. 2001].

6.2.2 Database schema matching

A large number of approaches for automated schema matching and integration has been published. Here, we briefly look at the approaches with respect to their applicability to the catalog matching problem. More details about the techniques proposed for matching can be found in [Rahm and Bernstein 2001].

Cupid [Madhavan et al. 2001], DIKE [Palopoli et al. 2000] and MOMIS [Bergamaschi and Beneventano 1999] use a purely structural approach to matching and integration. The structural matcher of Cupid [Madhavan et al. 2001] employs bottom-up propagation of similarity between classes in the schema tree. In catalog matching, we cannot rely on a purely structural approach, as the catalog structure alone does not say much about the meaning of a category.

SEMINT [Li and Clifton 2000] and DELTA [Clifton et al. 1997] use instance data only for integration. However, database instance data, i.e. database tuples are not comparable to documents. Moreover, we claim that relying exclusively on instance data will not allow for sufficiently accurate matching results.

The LSD [Doan et al. 2001] system uses a complex multi-strategy approach to integration. The LSD system operates on structured instance data, whereas the documents in a catalog constitute unstructured data. LSD also requires some manually mapped training data, i.e. documents for which the mediation result is already known. We cannot generally assume that this training data is available in a community scenario.

None of the existing matching techniques is directly applicable to our catalog mediation scenario. We are going to introduce the novel CAIMAN catalog matching approach in Section 6.4. The CAIMAN approach is based on text classification techniques, which is why we are first going to present some theoretical background of text classification.

6.3 Text classification techniques used in CAIMAN

In this section we introduce the theoretical background of the text classification techniques applied in the CAIMAN catalog matching approach. We are going to use the terms *text* and *document* interchangeably. More specifically, we consider a *document* to contain *text* that is enriched by additional (meta-)information. However, the difference between the two is not of interest for us in this section. Also, the terms *classification* and *categorization* are used interchangeably in existing literature [Sebastiani 2002].

Information Retrieval (IR) techniques are models and algorithms for retrieving textual information from document repositories [Manning and Schütze 1999]. In contrast to data retrieval, the data re-

trieved by information retrieval techniques does not adhere to a fixed structure like a database schema. IR techniques determine which of the documents from a document repository are relevant with respect to the query. Some IR techniques deliver only exactly matching results, whereas others deliver ranked lists of results ordered by relevance. For large document repositories, ranked lists have proven to be more useful than exact matches for most applications, as the final decision of relevance is up to the user.

Ad-hoc queries are queries entered by the user in an ad-hoc fashion when she requires information. Ad-hoc queries are difficult to answer, as the amount of information supplied in the query is usually very small. More information can be supplied in the query, if some example results of the query are already known in advance. The characteristics of these known results can be learned and posed as a query. A field that is closely related to IR, which learns characteristics of known query results, is text classification or text categorization.

Text categorization approaches take advantage of documents that have been previously assigned to a set of categories. Taking the information retrieval perspective, each category can be seen as a query with the document contents as the query description. New, previously unseen documents are only assigned to a category, if they are relevant to the query that is represented by the category's document contents. *Filtering* and *routing* are closely related to text classification, for cases with only two categories [Manning and Schütze 1999]. Filtering techniques decide, which documents are relevant and which are not, whereas routing techniques deliver a ranked list of results in order of relevance. In the CAIMAN matching approach, we use text categorization techniques that deliver both binary relevance feedback as well as ranked lists. More general theoretical background of IR is discussed in detail in [van Rijsbergen 1979], [Baeza-Yates and Ribeiro-Neto 1999], [Manning and Schütze 1999] and [Sebastiani 2002].

We can apply text categorization techniques to the catalog mediation problem, since catalogs with pre-categorized documents are available. With the problem specification from Section 6.1.1 in mind, we are going to present an overview of how automatic text categorization can be performed with machine learning techniques. This section is largely based on [Sebastiani 2002], [Manning and Schütze 1999], [Baeza-Yates and Ribeiro-Neto 1999] and [van Rijsbergen 1979]. We define automated text categorization in a way that has been adapted from [Sebastiani 2002]:

Definition 6.5 (Automated text categorization) *Consider a Boolean function $\Phi : \mathcal{D} \times \mathcal{C} \rightarrow \{T, F\}$ over a domain of documents $\mathcal{D}$ and a set of predefined categories $\mathcal{C}$, that describes how documents ought to be correctly classified. Automated text categorization approximates the unknown target function Φ by means of a function $\hat{\Phi} : \mathcal{D} \times \mathcal{C} \rightarrow \{T, F\}$ called classifier, such that Φ and $\hat{\Phi}$ coincide as much as possible.*

The function Φ maps a document-class-tuple (d_i, c_j) to a truth value, which signifies whether the respective document d_i belongs to class c_j or not. How the coincidence of the classifier $\hat{\Phi}$ and the correct classification is defined and measured, is described later in this section, when we talk about performance measures. A document may be assigned to several categories, i.e. the classifier function may have a truth value of T for several tuples (d_i, c_j).

For semi-automatic application scenarios like ours, a ranked list of class candidates is preferable over a boolean value of class containment as a result of the classification process. In practice, most classification techniques calculate a measure $\hat{\phi}(d_i, c_j)$ for how likely it is that a document d_i belongs to a class c_j. In order to be able to interpret this measure as a probability, we need to make some

further definitions. First, we assume that each document $d_i \in \mathcal{D}$ belongs to exactly one class, as given by Φ. Now we can interpret the classification of a document d_i as a random experiment with C as the random variable for the chosen category and ϕ as the discrete density function. We define:

$$\phi_(d_i, c_j) := P(C = c_j|d_i) \tag{6.2}$$

$$P(C = c_j|d_i) := \begin{cases} 1 & \text{if } \Phi(d_i, c_j) = T \\ 0 & \text{otherwise} \end{cases} \tag{6.3}$$

We can now say that if $\sum_{j=1}^{|\mathcal{C}|} \hat{\phi}(d_i, c_j) = 1$, then $\hat{\phi}(d_i, c_j)$ approximates $\phi_{d_i}(c_j)$ and thus

$$\hat{\phi}(d_i, c_j) \approx P(C = c_j|d_i) \tag{6.4}$$

We interpret $\hat{\phi}(d_i, c_j)$ as the probability that document d_i belongs to class c_j.

Different techniques have been applied to construct classifiers $\hat{\Phi}$ for automated text classification:

- The Knowledge Engineering approach is based on manually created rules, the creation of which is labor-intensive and inflexible with regard to changing document corpora.
- Machine learning approaches automatically induce classifiers from a set of pre-classified training examples. Even if a set of pre-classified training documents is not available and has to be manually created by an expert first, machine learning techniques are less labor intensive than knowledge engineering approaches [Sebastiani 2002].

Automatic evaluation of machine learning approaches

Machine learning approaches to document classification require one corpus of pre-classified example documents in order to be able to learn the characteristics of the classes from these examples. Moreover, for performance evaluation of the effectiveness of the trained classifier, another corpus of pre-classified documents is required to be able to compare the classification decision of the classifier to the actual classification. These two document sets have to be strictly intersection free in order to avoid unrealistically good performance results if the classifier is tested on the same corpus on which it has been trained.

To put it in a nutshell, we need a corpus $\mathcal{D}_p \subset \mathcal{D}$ for which all values of $\Phi(d_i, c_j)$ are known for every pair $(d_i, c_j) \in D_p \times \mathcal{C}$. This pre-classified corpus is divided into a training set $\mathcal{D}_{tr} \subset \mathcal{D}_p$ and a test set $\mathcal{D}_{te} = \mathcal{D}_p \setminus \mathcal{D}_{tr}$. The classifier $\hat{\Phi}$ is trained on $\mathcal{D}_{tr}$ and the test set $\mathcal{D}_{te}$ is classified using the trained classifier. An effectiveness measure is calculated based on how often the classification predictions of the classifier $\hat{\Phi}$ on $\mathcal{D}_{te}$ match the known values of Φ on $\mathcal{D}_{te}$. After having evaluated the performance of the classifier on the test set, the classifier is usually retrained on the complete set $\mathcal{D}_p$ to improve its performance. The calculated performance measure is thus always a lower bound of the actual performance of the classifier.

This approach, to divide the pre-classified corpus into train and test set is called *train-and-test* approach. An alternative approach, the performance estimate of which is closer to the actual performance of the fully trained classifier is the *k-fold cross validation* approach [Sebastiani 2002]. In the k-fold cross validation approach, the pre-classified corpus $\mathcal{D}_p$ is divided into k disjoint test sets $\mathcal{D}_{te}^i$. The train-and-test approach is then iteratively applied, using the document set $\mathcal{D}_p \setminus \mathcal{D}_{te}^i$ for training and $\mathcal{D}_{te}^i$ for evaluation. The performance results of the k differently trained classifiers are then aggregated to result in an overall performance measure of the approach.

Many machine learning approaches have parameters which can be tuned for classification of a certain corpus. For parameter tuning, the original training set $\mathcal{D}_{tr}$ is again subdivided into a parameter

tuning training set $\mathcal{D}'_{tr}$ and a tuning validation set $\mathcal{D}'_{te}$. The classifier can then be trained on $\mathcal{D}'_{tr}$ and its performance be evaluated on $\mathcal{D}'_{te}$ repeatedly with different parameter settings until good values for the parameters are found. The parameter tuning cannot be performed on the original training and test set, because in that case the parameters would be tuned to the same document set they would later be evaluated on. To yield more realistic performance results, one would like to evaluate the performance on a corpus that has not been previously seen by the classifier.

Next, we are going to introduce the three sequential steps that are performed in machine learning approaches to text classification: 1) corpus indexing, 2) classifier construction, and 3) performance evaluation.

6.3.1 Document corpus indexing and feature weighting

In order to make machine learning techniques applicable to natural language text documents, the text has to be transformed to a representation that can be used for classifier deduction. This transformation process is called *indexing* in IR terms. For text classification purposes, documents are usually represented as vectors in a term space, in which the different terms represent dimensions. A document is thus represented as a vector of term or *feature weights* $\overrightarrow{d_i} = (\omega_{1i}, \ldots, \omega_{|\mathcal{T}|i})$, where $\mathcal{T}$ is the set of terms that occur at least in one of the training documents from the training corpus $\mathcal{D}_{tr}$. The weights $0 \leq \omega_{ki} \leq 1$ represent how much term t_{ki} contributes to the semantics of document d_i [Sebastiani 2002]. This document vector scheme is common to all indexing approaches we consider. The different schemes only differ in the understanding of what a term is and how the term weights are calculated.

The simplest way to define terms is to associate them with words. This *set of words* approach completely disregards any compositional semantics of words in sentences. However, more complex term definitions have largely been shown not to improve the classification performance [Sebastiani 2002]. Thus, we are going to use the *set of words* approach for text categorization in CAIMAN.

Before feature weights are calculated, it has been shown to be useful to perform some transformations on the collection of terms of a document. One of these transformations is the removal of *stop words* or *function words* such as articles, prepositions, conjunctions etc. [Sebastiani 2002] as these terms will not be useful for discrimination between documents.

Another useful transformation to be done is *stemming*, where words are reduced to their morphological root. A simple, but effective stemming algorithm is the *Porter stemmer*, which reduces words to their assumed morphological roots by truncating suffixes [Sebastiani 2002]. The Porter stemmer is sufficient for most IR applications.

The most popular way to calculate feature weights is the *tfidf (term frequency inverse document frequency)* function [Sebastiani 2002], defined as

$$tfidf(t_k, d_i) = tf(t_k, d_i) \cdot log(\frac{|\mathcal{D}_{tr}|}{df(t_k, \mathcal{D}_{tr})}) \tag{6.5}$$

where $tf(t_k, d_i)$ is the number of times term t_k occurs in document d_i and $df(t_k, \mathcal{D}_{tr})$ is the number of documents in $\mathcal{D}_{tr}$, in which the term t_k occurs at least once. The *tfidf* function is based on two intuitions. The first intuition is that the more frequent a term is in a document, the more representative this term is for the content of this document (*tf term frequency*). The second intuition is that if a term occurs in a lot of documents of the collection, it is less discriminating than a term that occurs only in a few documents (*idf inverse document frequency*). The *tfidf* weighting scheme completely neglects any importance of the order of terms in a document. To calculate feature weights ω_{ki} from the *tfidf*

function, the *cosine normalization* is computed:

$$\omega_{ki} = \frac{tfidf(t_k, d_i)}{\sqrt{\sum_{s=1}^{|\mathcal{T}|}(tfidf(t_s, d_i))^2}} \tag{6.6}$$

where $\mathcal{T}$ is the set of all terms that occur in any of the documents in $\mathcal{D}_{tr}$.

6.3.1.1 Indexing HTML documents

The structure of HTML documents and hypertext documents in general is very different from plain text documents. The contents of one unstructured plain text document are usually distributed over several hypertext pages. Thus, HTML documents require special indexing techniques [Yang et al. 2002]. Although we are not going to use HTML indexing techniques in the CAIMAN prototype, we would like to give the reader an impression of how the structure of hypertext documents can be exploited for text categorization. The main focus of this work and the CAIMAN prototype implementation, however, are standard indexing techniques, which make no assumptions about structure or other higher level document semantics. We aim to show how the same standard IR techniques perform on different document catalogs, in order to make the results comparable.

As the contents of a hypertext document entity are typically distributed over several hypertext pages, there is often very little text on an HTML page, which could be used for automated classification. We have run some preliminary experiments that showed that the classification performance using standard indexing techniques for HTML pages was significantly reduced compared to catalogs with unstructured documents.

A survey of state of the art indexing and categorization techniques for hypertext documents has been published in [Yang et al. 2002]. The presented approaches exploit the hypertext structure of HTML pages for classification. As an illustrating example of how hypertext structure can be exploited, we are going to present a brief overview over one hypertext categorization approach presented in [Attardi et al. 1999]. The approach in [Attardi et al. 1999] effectively classifies HTML pages by context instead of content. The context of an HTML document d_i is defined as all pages which link to d_i. The intuition behind the technique presented in [Attardi et al. 1999] is, that a web page which refers to an HTML document d_i, usually contains concise information about d_i's contents. Especially the hyperlink itself usually contains a brief synopsis of the linked page. Moreover, other elements in pages that link to d_i, such as for example the title element will also hold valuable information about d_i. Context information from pages that link to d_i is then used for classification of d_i. The classification approach presented in [Attardi et al. 1999] is thus based on the following assumptions:

- a web page which refers to a page p contains information about p's contents
- elements that are nested around a link to p hold information about p's content
- the information from referring pages is sufficient to classify p

The classification results presented in [Attardi et al. 1999] are promising and an integration of hypertext classification techniques into the CAIMAN matching approach is a worthwhile field for future research (see Section 9.2).

6.3.2 Feature vector dimensionality reduction

The total size of the vocabulary or *dimensionality of the feature space* $|\mathcal{T}|$ can become a problem in automatic text classification[3]. The problems that occur with high dimensionality of the feature space are:

- One problem is the sheer complexity of operations that have to be performed for classifier construction.
- Another problem is, that the document vectors are typically very sparse vectors in a high-dimensional space and some classifier deduction algorithms are not suited for that kind of training data.
- A third problem is *overfitting* of deduced classifiers. Overfitting is the phenomenon, when a classifier deduction technique adapts too much to the specific characteristics of the training examples and does not generalize enough to accept documents, which deviate from the training examples.

All three problems that occur with high-dimensional feature spaces can be minimized, if the dimensionality of the feature space is reduced in some way before the classifier deduction. Several applicable techniques have been proposed and shown to perform effectively. However, dimensionality reduction bears the risk of neglecting features that are important characteristics of the respective training document set. Thus, the performance of a classifier will decrease, if the dimensionality is reduced too much.

Generally, there are two approaches to dimensionality reduction:

- **Local dimensionality reduction** approaches choose a subset of terms $\mathcal{T}'_j$ for each training class c_j, such that typically, $10 \leq |\mathcal{T}'_j| \leq 50$.
- **Global dimensionality reduction** approaches choose a set of terms $\mathcal{T}'$, such that $|\mathcal{T}'| \ll |\mathcal{T}|$, which applies for classification for all classes c_j.

The dimensionality of the feature space can be reduced by:

- **Term selection** techniques choose a subset $\mathcal{T}'$ of the original terms $\mathcal{T}$ based on varying decision criteria.
- **Term extraction** techniques generate a new term set $\mathcal{T}'$ with $|\mathcal{T}'| < |\mathcal{T}|$. However, the new terms in $\mathcal{T}'$ are not necessarily terms from $\mathcal{T}$, but instead have been generated from the terms in $\mathcal{T}$ by term extraction techniques.

A simple decision function for global term selection is the document frequency function $df(t_k, \mathcal{D}_{tr})$, introduced in Equation 6.5. Only the terms that appear in a lot of the documents in the training set $\mathcal{D}_{tr}$ are kept, the other terms are discarded. It has been shown that it is possible to reduce the dimensionality by a factor of 10, while not incurring any loss [Sebastiani 2002].

[3]Typically, for English documents, feature spaces have roughly $10000 - 20000$ dimensions.

Term set selection with information gain is an elaborate term set selection technique, which outperforms the simple document frequency approach. The measure of *information gain* is based on the notion of *entropy* from the field of information theory [Manning and Schütze 1999]:

Definition 6.6 (Entropy) *The entropy of a discrete random variable C with a discrete set of symbols $\mathcal{C}$ is defined as*

$$H(C) = -\sum_{c \in \mathcal{C}} P(C = c) \cdot log_2 P(C = c) \tag{6.7}$$

For this definition we also define that $0 \cdot log_2 0 := 0$. We use the symbol C here, because we are going to use the definition of entropy with the set of categories in a catalog. However, the above definition of entropy is a general definition, which does not only apply to the special case with categories. An intuitive description of the meaning of entropy is the *expected number of bits that are required to encode the outcome of the random variable C*. Thus, entropy is a measure of the amount of information in a random variable [Manning and Schütze 1999]. The *information gain (IG)* is defined in the following way in [Manning and Schütze 1999],[Yang and Pedersen 1997]:

Definition 6.7 (Information Gain) *The information gain between two discrete random variables C and T is defined as :*

$$IG(C, T) = H(C) - H(C|T) \tag{6.8}$$

The information gain measure can be interpreted as the amount of information that can be gained for the outcome of the random variable C, if the outcome of the random variable T is known.

We now apply the information gain measure to text classification. Consider a random experiment in which a document d is to be assigned to a class, based on the information that a certain term occurs in that document. Let the random variable C denote the class to which d is assigned and T be the term, about which we know that it occurs in d. The information gain $IG(C, T)$ can now be interpreted as a measure for how valuable the information, that a certain term T occurs in d, is for the decision to which class to assign the document d. After some transformations of definition 6.7 we get [Sebastiani 2002]:

$$IG(c_j, t_k) = \sum_{t \in \{t_k, \overline{t_k}\}} \sum_{c \in \{c_j, \overline{c_j}\}} P(c, t) \cdot log \frac{P(c, t)}{P(c) \cdot P(t)} \tag{6.9}$$

where c_j signifies the event that document d is assigned to c_j and t_k signifies the event that term t_k is known to occur in d; $\overline{c_j}$ and $\overline{t_k}$ signify the respective inverse events. $P(c, t)$ signifies the probability that for a random document d_i, the term t occurs in d_i and d_i belongs to category c. $P(c, t)$ can be estimated by counting documents for which the above holds among documents in the training set.

$IG(c_j, t_k)$ is a measure for how valuable the term t_k is for the classifier decision whether or not a document belongs to class c_j. For dimensionality reduction, terms with a low information gain are discarded. As can be seen from Equation 6.9, $IG(c_j, t_k)$ is a dimensionality reduction function that is local to category c_j. To derive a global dimensionality reduction function, which is category independent, either one of the following three combinations of the category specific local information gain functions is computed [Sebastiani 2002]:

$$IG_{sum}(t_k) = \sum_{i=1}^{|\mathcal{C}|} IG(c_i, t_k), \tag{6.10}$$

$$IG_{wsum}(t_k) = \sum_{i=1}^{|\mathcal{C}|} P(c_i) IG(c_i, t_k), \tag{6.11}$$

$$IG_{max}(t_k) = max_{i=1}^{|\mathcal{C}|} IG(c_i, t_k) \tag{6.12}$$

Global dimensionality reduction by a factor of 100 with the information gain measure has been shown to incur no decline in classification effectiveness or even improve effectiveness [Sebastiani 2002]. We are going to use global dimensionality reduction based on information gain in CAIMAN.

Term extraction techniques are not used in CAIMAN, however, we give a brief overview because term extraction techniques are popular in existing IR publications. In contrast to the term selection techniques we have presented so far, term extraction techniques for dimensionality reduction do not pick a subset of the original terms, but generate a set of synthetic terms which best represent the chosen document corpus. The intuition behind the creation of synthetic terms is that the original terms may suffer from problems with polysemy, homonymy and synonymy and might not be the optimal independent dimensions for the document feature space [Sebastiani 2002]. Term extraction techniques generally consist of two steps: 1) the automatic extraction of the new synthetic terms and 2) the conversion of the original document vectors into vectors with respect to the new term basis. Two classes of techniques are presented in [Sebastiani 2002]: *term clustering* techniques and *latent semantic indexing*. Term clustering techniques have been reported to exhibit a mere 2% classification effectiveness loss with a 1000-fold reduction of terms. Latent semantic indexing has been shown to outperform term selection techniques [Sebastiani 2002]. For our catalog mediation technique we do not employ term extraction techniques. The performance of term selection techniques is sufficient for our purposes and term selection techniques are easier to implement. For more detailed information on term extraction techniques, the interested reader is referred to [Sebastiani 2002] and other sources cited there.

6.3.3 Construction of text classifiers

A large number of techniques for automatic classifier construction have been proposed in the IR literature. We picked three of these techniques due to their distinctive features: 1) the *Naive Bayes* technique, because the intuition behind it is straightforward, it is easy to implement and has good runtime complexity characteristics, 2) the *Rocchio centroid* technique, because the learning model allows to represent both categories and documents as a feature vector and thus allows for interesting combinations, and 3) a *Support Vector Machine (SVM)* approach, because this technique generalizes quickly from few examples and has been shown to perform well on small feature spaces as well as on large feature spaces. The SVM approach, however, is very complex to implement and has a high runtime complexity. All three of the chosen classifier construction techniques are among the best in terms of classification effectiveness. Each of three chosen techniques is going to be introduced in more detail. For a more recent survey on classifier construction techniques, the interested reader is referred to [Sebastiani 2002].

6.3.3.1 Naive Bayes Classifier construction

Naive Bayes Classifier construction is a probabilistic approach. The main intuition of Naive Bayes classification is that classification consists of two dependent random experiments with the random variable D, which signifies a document feature vector and C, which signifies the class chosen for classification. In the first random experiment, a document feature vector $\overrightarrow{d_j}$ is chosen, in the second random experiment, the class c_i to which $\overrightarrow{d_j}$ belongs, is chosen. The Naive Bayes classifier estimates the conditional probability $P(C = c_i|D = \overrightarrow{d_j})$ that document d_j belongs to class c_i. This probability for classification of new documents (*a posteriori probability*) can be estimated using term statistics in documents of the training corpus (*a priori probability*), based on the assumption that occurrences

of different terms in the training documents are independent from each other[4]. Although this assumption is not realistic for natural language documents, it has been shown not to hurt the qualitative classification performance of the Naive Bayes classifier.

In the following, we are going to abbreviate the assignment of a value to a random variable simply by the value itself, i.e. $P(C = c_i)$ will be abbreviated by $P(c_i)$. The basis of the Naive Bayes classifier is Bayes' theorem, through which the required probability can be expressed as

$$P(c_i|\overrightarrow{d_j}) = \frac{P(c_i) \cdot P(\overrightarrow{d_j}|c_i)}{P(\overrightarrow{d_j})} \tag{6.13}$$

where $P(c_i|\overrightarrow{d_j})$ is the conditional probability that if the term vector of document d_j is $\overrightarrow{d_j}$, d_j belongs to category c_i. We now need to calculate the terms on the right hand side of Equation 6.13 from the training data to get the desired probability on the left hand side of Equation 6.13.

To estimate $P(\overrightarrow{d_j}|c_i)$ from the training documents, the *term independence assumption* is necessary: the document term vector $\overrightarrow{d_j}$ is known to be in class c_i. The term independence assumption says that the occurrences of single terms $t_{jk} \in \mathcal{T}'$ for $k = 1, \ldots, |\mathcal{T}'|$ in a document d_j are statistically independent from each other.

We also assume the single terms in a document to be multinomially distributed. The multinomial model along with the multivariate Bernoulli model are term distribution models, which allow to take into account multiple occurrences of terms in one document. These models have been shown to outperform plain binary occurrence binomial event models [McCallum 1998]. The multinomial model sees a document as an ordered sequence of word events, which are drawn from the same term set $\mathcal{T}'$. It is assumed that the document lengths are independent of the category, to which the documents belong. The term independence assumption also includes that multiple occurrences of the same term are independent from each other. We can express the probability estimate for the occurrence of a certain document, given its category and assumed multinomial term distribution as presented in [McCallum 1998]:

$$\begin{aligned} \overrightarrow{d_j} &= (d_{j1}, \ldots, d_{jk}, \ldots, d_{j|\mathcal{T}'|}) \\ \hat{P}(\overrightarrow{d_j}|c_i) &= P(|\overrightarrow{d_j}|) \cdot |\overrightarrow{d_j}|! \cdot \prod_{k=1}^{|\mathcal{T}'|} \frac{\hat{P}(t_k|c_i)^{d_{jk}}}{d_{jk}!} \end{aligned} \tag{6.14}$$

where d_{jk} is the weight or term coefficient of term $t_k \in \mathcal{T}'$ and $\mathcal{T}'$ is the reduced term set of the training document set after dimensionality reduction.

$P(|\overrightarrow{d_j}|)$ is the probability that a random document consists of $|\overrightarrow{d_j}|$ terms in total. If there are n documents of length $|\overrightarrow{d_j}|$ in the training set , we can estimate $\hat{P}(|\overrightarrow{d_j}|) \approx n/|\mathcal{D}|$. Finally, the probability of a term occurrence given a class membership of a document can be estimated by a Laplace estimate as shown in [McCallum 1998]:

$$\hat{P}(t_k|c_i) = \frac{1 + \sum_{j=1}^{|\mathcal{D}|} d_{jk} \cdot P(c_i|d_j)}{|\mathcal{T}'| + \sum_{m=1}^{|\mathcal{T}'|} \sum_{j=1}^{|\mathcal{D}|} d_{jm} \cdot P(c_i|d_j)} \tag{6.15}$$

Note that $P(c_i|d_j)$ for a given d_j is either 1 if d_j belongs to c_i or 0 if not. Thus, informally put, Equation 6.15 approximates the number of occurrences of term t_k in class c_i divided by the number

[4]The term independence assumption is the reason for the qualification of the Bayes classifier as "naive".

of all term occurrences in class c_i. Only $\hat{P}(c_i)$ remains to be estimated from the training data and is given in [McCallum 1998] as:

$$\hat{P}(c_i) = \frac{\sum_{j=1}^{|D|} P(c_i|d_j)}{|D|} \tag{6.16}$$

Equation 6.16 represents the fraction of documents in the training set that are in class c_i.

We have now estimated all terms on the right hand side of Equation 6.13 except for $P(\overrightarrow{d_j})$ in the denominator of Equation 6.13. The probability $P(\overrightarrow{d_j})$ is the same for all c_k and thus does not make a difference in the classifier decision. Consequently, $P(\overrightarrow{d_j})$ is commonly not estimated for Naive Bayes classifiers. However, this also means that the resulting estimate of Equation 6.13 cannot be interpreted as a probability anymore. Similar to Equation 6.4 and with Equation 6.13, we re-define the Naive Bayes classifier as:

$$\hat{\phi}_B(d_j, c_i) = \hat{P}(c_i) \cdot \hat{P}(\overrightarrow{d_j}|c_i) \tag{6.17}$$

$$\hat{\Phi}_B(d_j, c_i) = \begin{cases} T & \text{if } c_i = c'_k : \hat{\phi}(d_j, c'_k) \geq \hat{\phi}(d_j, c_k) \quad \forall c_k \in \mathcal{C} \\ F & \text{otherwise} \end{cases} \tag{6.18}$$

$\hat{\phi}_B(d_j, c_i)$ can not be interpreted as a probability. The classifier $\hat{\Phi}_B$ then simply chooses the category c_i, for which $\hat{\phi}(d_j, c_i)$ has the maximum value among all arguments c_k.

The Naive Bayes approach is easy to implement and has a low runtime complexity, while maintaining good classification performance [McCallum 1998]. A disadvantage of the Naive Bayes classifier is that it is very much prone to overfitting. Overfitting means that a classifier learns the exact characteristics of the given examples and does not generalize enough to recognize unseen texts as relevant [Sebastiani 2002]. In the Naive Bayes approach, the training and classification phase cannot be separated.

6.3.3.2 Support Vector Machine classifier construction

The Support Vector Machine (SVM) technique is a machine learning technique, which has first been used for text classification in [Joachims 1998]. The geometric interpretation of SVM learning is finding a hypersurface in a $|\mathcal{T}|$-dimensional space, in which the documents are represented as vectors, such that the hyperplane separates the positive training examples in the best way from the negative training examples. To explain this interpretation in more detail, we present an example with $|\mathcal{T}| = 2$, which can be seen in Figure 6.2 (see also [Sebastiani 2002]).

Figure 6.2 shows a 2-dimensional space with positive training examples represented by a $+$ symbol and negative examples by a o symbol. The straight lines represent the possible *decision surfaces*, $|\mathcal{T}| - 1$-dimensional hyperplanes, which separate the positive from the negative examples. Among all decision surfaces, SVM learning finds the decision surface S_i, which separates the positive from the negative examples with the maximum margin. The maximum margin is defined by the maximum possible translation of S_i, such that the separation property remains. In Figure 6.2, the two lines parallel to S_i represent the maximum possible translation of S_i, as they just touch the positive and negative training examples closest to S_i - any more translation and a positive example would fall into the set of negative examples or vice versa. The positive and negative examples which are closest to the decision surface and define its position are highlighted by small boxes around them in Figure 6.2. These defining examples are called *support vectors*.

If the examples are divided by a linear function, as in Figure 6.2, one speaks of a *linear kernel* SVM. Kernel functions with higher degrees d_{svm} are possible, but experimental results showed that

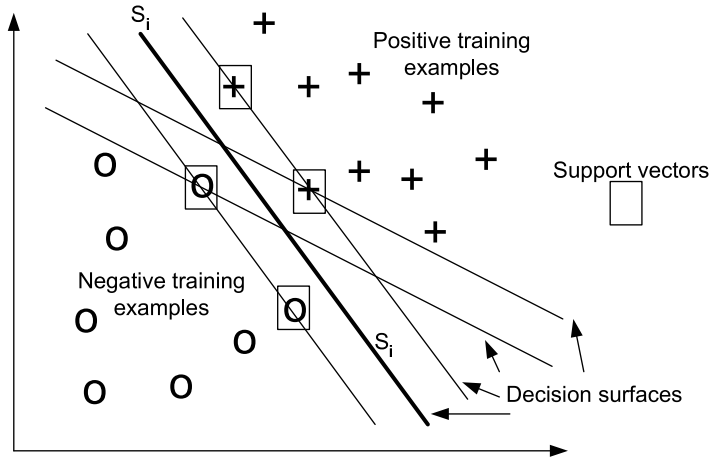


FIGURE 6.2: Support Vector Machine learning: finding the best hyperplane (adapted from [Sebastiani 2002]).

linear kernel SVMs perform better [Etzold 2003]. If the training examples are not separable using the chosen kernel, a cost c_{svm} can be assigned to each falsely classified training example to still find an optimal separation. More details on SVM text classification can be found in [Joachims 1998] and [Yang and Liu 1999].

SVMs have three important advantages for text classification [Joachims 1998]:

- feature selection may not be necessary, as SVMs scale well with large feature spaces. Moreover, SVMs are not prone to overfitting to the training examples.
- no effort is necessary for parameter tuning, as there is a theoretically motivated choice for SVM parameters, which has been shown to deliver best results.
- The superior performance of SVM based classifiers has been demonstrated in [Joachims 1997], [Joachims 1998] and [Yang and Liu 1999].

A disadvantage of SVMs for text classification is their training speed. Due to the complexity of the algorithm, SVMs in their basic form could not compete with other approaches presented here in terms of runtime. However, it has been shown that through optimizations, SVMs can compete with simpler classifiers such as Rocchio [Dumais et al. 1998]. An SVM classifier is more complex to implement than the Naive Bayes approach. In the CAIMAN implementation, we use the optimized SVM implementation presented in [Chang and Lin 2001].

6.3.3.3 Rocchio Centroid classifier construction for context aware classification

The Rocchio Centroid [Sebastiani 2002] can be seen as a category representative vector, the purpose of which is to represent a category with all of its documents in one vector. The Rocchio Centroid classification technique is not used as a classifier in CAIMAN. Instead, we use the Rocchio Centroid as a category representative vector for **context aware classification**. Context aware classification is a classification approach, that takes information about the source category of a document into account for classification. Context aware classification has first been designed for classification in [Groh 2001; Lacher and Groh 2001] and is described in more detail in Section 6.5.3.1.

The Rocchio Centroid as category representative vector is constructed as follows. Let ϱ_i represent the Rocchio centroid vector for category c_i, which holds the set of documents $d_j \in c_i$. The individual components ϱ_i^k of the Rocchio Centroid vector for each term feature t_k are then calculated as:

$$\varrho_i^k = \frac{\beta}{||\{d_j \in c_i\}||} \sum_{\{d_j \in c_i\}} w_{kj} - \frac{\gamma}{||\{d_j \notin c_i\}||} \sum_{\{d_j \notin c_i\}} w_{kj}. \tag{6.19}$$

where w_{kj} is the weight of the feature t_k in document d_j and β and γ are weight parameters. The intuition behind the calculation is, that the first term represents a weighted average vector (*centroid*) of all documents $d_j \in c_i$ with weight β. As the Rocchio centroid is usually used for classification, a second term has been introduced in Equation 6.19, which is the weighted average of all documents $d_j \notin c_i$ with weight γ. The second term is introduced to improve discrimination between categories for classification. Experimental results in [Joachims 1997] showed that for classification $\beta = 16$ and $\gamma = 4$ are a good choice.

6.3.4 Multi-category classification

As all of the presented classification techniques come from an Information Retrieval background, they are not especially suitable for multi-category classification. In their basic form, the classifiers decide whether or not a document belongs to a certain category - not to which category among a number of categories. However, with some variations, they can easily be adapted for classification with multiple destination categories.

Naive Bayes classification can be adapted to multi-category classification by a simple modification. For each category, the classifier estimates the probability $P(c_i|\overrightarrow{d_j})$, as shown in Equation 6.13. The Naive Bayes classifier is adapted to multi-category classification by simply choosing the c_i, for which $P(c_i|\overrightarrow{d_j})$ is maximal among all c_i, as shown in Equation 6.17.

Vector-based classification techniques like SVM, another approach is required. Two different approaches have been referred to in [Chang and Lin 2001]: the *one-against-one* approach and the *one-against-all* approach. In the *one-against-one* approach, for k categories, $k \cdot (k-1)/2$ classifiers are constructed, one for each pair of categories from the original catalog. After having trained the classifiers, the new and unseen documents are given as input to each of the trained classifiers. Each positive classification decision of a classifier counts as a vote for the respective category. For the final classification decision, the category with the maximum number of votes is picked. The application of the *one-against-one* approach to SVM is described in more detail in [Chang and Lin 2001]. In the alternative *one-against-all* approach, k classifiers are trained, one for each category c_i. The documents in category c_i are used as positive training examples, all other documents in the catalog as negative examples. Again, the single document classification decisions are counted as votes in favor or against the respective category. In [Chang and Lin 2001], experiments that show the superior performance of the *one-against-one* approach are referenced. The *one-against-one* approach has been implemented in the SVM implementation presented in [Chang and Lin 2001], which we use for the CAIMAN matching implementation.

6.3.5 Information retrieval performance measures

Evaluation of text classification techniques is typically performed experimentally rather than analytically. The reason for the experimental evaluation lies in the subjective nature of the text classification:

to which category a document should belong is a decision that different users will take with different outcomes [Sebastiani 2002]. Evaluation measures for text classification consider *effectiveness*, i.e. whether a classification is right or not, instead of runtime efficiency. All measures presented here rely on the advance availability of information about the actual correct classification of documents.

Precision and Recall are the most popular measures for both Information Retrieval as well as text classification evaluation. They are defined in the following way in [Sebastiani 2002]:

- **Precision** with respect to category c_i is defined as $P(\,\Phi(d_x, c_i) = T \,|\, \hat{\Phi}(d_x, c_i) = T\,)$.
- **Recall** with respect to category c_i is defined as $P(\,\hat{\Phi}(d_x, c_i) = T \,|\, \Phi(d_x, c_i) = T\,)$.

These two probabilities can be estimated using *contingency values*, which are defined in the following way:

- $TP_i = |\{d_j|\hat{\Phi}(d_x, c_i) = T \wedge \Phi(d_x, c_i) = T\}|$ (true positive classifications).
- $FP_i = |\{d_j|\hat{\Phi}(d_x, c_i) = T \wedge \Phi(d_x, c_i) = F\}|$ (false positive classifications).
- $FN_i = |\{d_j|\hat{\Phi}(d_x, c_i) = F \wedge \Phi(d_x, c_i) = T\}|$ (false negative classifications).
- $TN_i = |\{d_j|\hat{\Phi}(d_x, c_i) = F \wedge \Phi(d_x, c_i) = F\}|$ (true negative classifications).

The different contingency values can be summed up by simply comparing the prediction of the classifier $\hat{\Phi}$ to what is known to be the correct classification decision Φ. The estimates $\hat{\rho}_i$ of *recall* and $\hat{\pi}_i$ of *precision* for category c_i are then defined as:

$$\hat{\pi}_i = \frac{TP_i}{TP_i + FP_i}, \quad \hat{\rho}_i = \frac{TP_i}{TP_i + FN_i}. \tag{6.20}$$

To calculate the values of $\hat{\rho}$ and $\hat{\pi}$ for a complete catalog, the values for single categories have to be averaged. Two different averaging techniques with different characteristics have been described in [Sebastiani 2002]: **micro-averaging** and **macro-averaging**. In *micro-averaging*, the individual contingency values are first summed up over all categories and thereafter, $\hat{\rho}$ and $\hat{\pi}$ are calculated analogously to Equation 6.20. In *macro-averaging*, the values of $\hat{\rho}$ and $\hat{\pi}$ are calculated as the average of the $\hat{\rho}_i$ and $\hat{\pi}_i$ over all categories c_i. These two averaging techniques may deliver substantially different results for the same classification experiment, especially if the test catalog contains categories with large variations in category sizes. Macro-averaging emphasizes the performance of a classifier on categories with few test documents, since small categories have the same weight as large categories in the averaged value. Micro-averaging values instead are dominated by the performance of a classifier on large categories. Neither one of the two measures can be considered superior to the other, they just have different characteristics. In fact, in order to present a complete and balanced picture of the performance of a classifier, it is advisable to show the values of both averaging techniques. We are going to present the results of our experiments in Section 8 using both averaging techniques.

Accuracy is another performance measure, which is frequently used in IR literature. There are several different mathematical definitions for measures termed *accuracy*. One definition is given in [Sebastiani 2002] as $\hat{A} = \frac{TP+TN}{TP+TN+FP+FN}$. In our opinion, this measure only makes sense in an IR setting, in which a classifier decides whether a document belongs to a category or not. The more documents and categories are involved, the larger TN becomes in comparison to the other

terms involved. For large catalogs, the accuracy measure is likely to deliver values close to 1, almost independent of the actual classifier performance. It has been shown in [Yang and Liu 1999] that with the above accuracy measure, a classifier that rejects all documents for all categories (*trivial rejector*) tends to outperform all non-trivial classifiers. A different definition for *accuracy* that much resembles our definition of *precision* is used in [Agrawal and Srikant 2001][5].

F-Measure. As can be seen from Equation 6.20, there exists a trade-off between recall and precision. Therefore, one of the two measures by itself will not provide a complete picture of classifier performance. The F_β function [van Rijsbergen 1979] (also called F-measure) is a measure that combines the measures precision and recall:

$$F_\beta = \frac{(\beta^2 + 1) \cdot \pi \cdot \rho}{\beta^2 \cdot \pi + \rho} \tag{6.21}$$

where β can be interpreted as the relative importance of π and ρ respectively. For $\beta = 1$, equal importance is attributed to π and ρ, which is why we used the F_1 measure in our experiments in Chapter 8.

6.4 Overview of the CAIMAN catalog matching approach

We have compared document granular catalog mediation to category granular catalog mediation in Chapter 5. Category granular approaches have a number of advantages over document granular approaches in our scenario. Existing text classification techniques can be used for document granular mediation. However, none of the existing matching techniques for conceptual structures is directly applicable in a category granular mediation approach for our scenario (see Section 6.2).

We introduce the novel CAIMAN catalog matching approach, which we use for category granular mediation in our scenario. Using the CAIMAN catalog matching approach for catalog mediation has the inherent advantages of a category granular mediation approach (see Section 5.3.2). The CAIMAN matching approach also fulfills the application requirements of the knowledge exchange services (see Section 5.1.2):

1. Our matching approach allows to convey all kinds of knowledge mentioned in the application requirements.

2. The matching quality and thus the mediation quality of our approach is high (see Chapter 8).

3. The matching approach can cope with changing catalog contents efficiently.

4. Our matching approach scales well with large catalogs (see Section 8).

The CAIMAN matching approach uses the document contents and the generic catalog structure to calculate catalog matchings. Name labels are not used for the calculations, because they are too subjective in our scenario and would impair the matching quality.

[5]The definition of *accuracy* in [Agrawal and Srikant 2001] and *micro-averaged precision* here are incidental, if all documents from the test catalog are assigned to a category during classification.

The calculation of the catalog matching M as defined in Section 6.1.1 is performed in the CAIMAN approach in three consecutive phases:

- In the **document categorization phase**, the user catalog is indexed and classifiers are trained for the user catalog. Using the trained classifiers, the classification result for a subset of documents from each category in the community catalog is calculated. This phase serves to establish an initial connection between categories by document categorization.
- In the **category correlation phase**, the document categorization results from the categorization phase are used as input for the computation of initial values of the similarity function σ. The initial values of σ are calculated for all category pairs in the cross product of the user categories and the community categories $\mathcal{C}^U \times \mathcal{C}^C$. The initial similarities are calculated by a voting approach.
- The **structural matching phase** consists of two different sub-phases:
  - The **similarity improvement phase** takes the initial category similarities as input to calculate improved similarities between user and community categories. The improved similarities are calculated by exploiting the graph structure of both catalogs using graph matching techniques.
  - The **match candidate filtering phase** applies the service specific filter function $\mathcal{F}^K$ to filter out the final category match pairs.

An overview of the three phases and the respective steps in each phase can be seen in Figure 6.3.

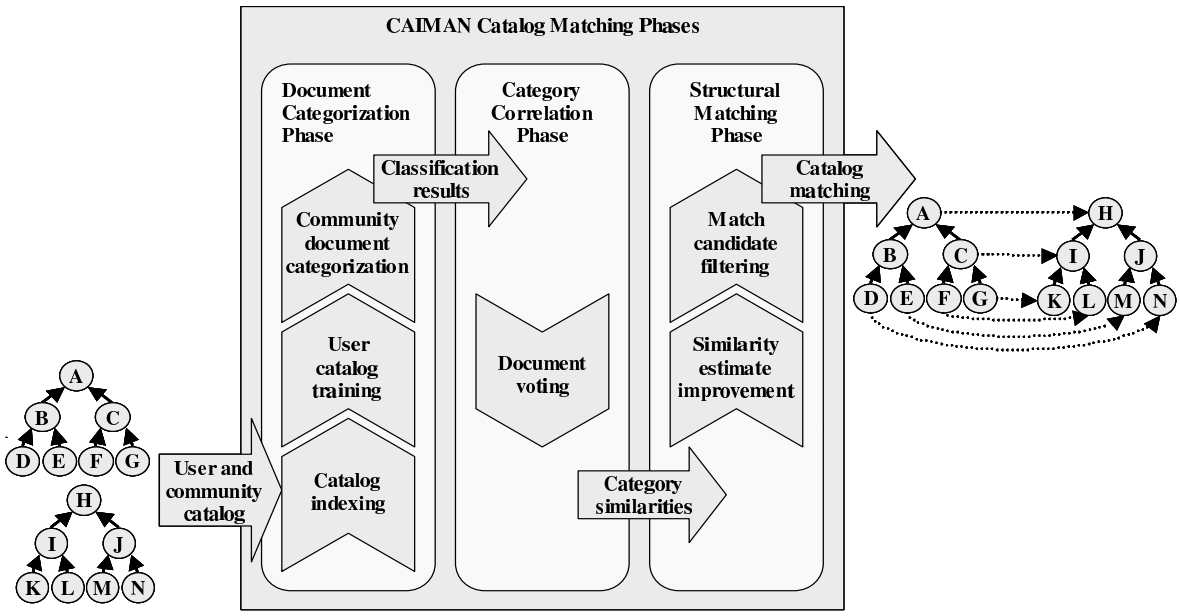


FIGURE 6.3: Phases of the CAIMAN catalog matching approach

Our matching approach as described here takes two catalogs as input for the calculations and outputs a final catalog matching. One of the catalogs is always the user catalog, the other is the catalog, for which a matching with the user catalog should be calculated - by default the community catalog.

Both catalogs are structured as presented in Section 2.3.3.1 and contain the respective documents or references to documents. We assume that no preparatory calculations have been made, i.e. the calculation sequence described here resembles a situation, in which the two catalogs have just been chosen by the user for mediation.

6.5 Document categorization phase

The categorization phase consists of three consecutive steps: user catalog indexing, user catalog classifier training and community document categorization. The indexing step brings the documents from the user catalog into a format that allows the machine learning techniques presented in Section 6.3 to be applied. Then, classifiers for the user catalog are trained with documents from the user catalog. A subset of the documents from the community catalog are categorized with respect to the user catalog using the trained classifiers. The classification results $\hat{\phi}(d_i, c_j)$ for a set of community documents $d_i \in \mathcal{D}^C$ and user categories $c_j \in \mathcal{C}^U$ are calculated using the trained classifiers. In the following, we are going to describe the three consecutive steps of the classification phase in more detail.

6.5.1 Catalog indexing

The machine learning techniques applied in the CAIMAN matching approach require documents to be indexed as feature vectors. The indexing procedure consists of the following steps:

- **Plain text conversion:** the documents are converted from their respective source formats to plain text documents, e.g. from Postscript[6] to ASCII representation.
- **Stop word removal:** terms, which occur in almost all documents and thus do not have discriminative power, are removed from the documents (see Section 6.3). Typical stop words are for example “and”, “or” and “that”.
- **Stemming:** terms are reduced to their morphological root. Stemming improves the classification performance. In our mediation approach, we apply a Porter stemmer (see Section 6.3). “Querying’, for example, would be reduced to “query” in the stemming step.
- **Feature vector construction:** each term, which occurs in the whole document corpus, is considered one dimension in a feature space. Each document can now be represented as a vector in the feature space (see Section 6.3), with word occurrence counts as the vector coefficients.
- **Feature weighting:** the features of the document vector are weighted according to their importance and discriminative quality for the document. We use a *tfidf* weighting scheme (see Section 6.3).
- **Feature vector dimensionality reduction:** As a large feature space has negative influence on runtime and space requirements complexity, it is desirable to reduce the dimensionality of the feature space. The CAIMAN matching approach uses the information gain measure (see Section 6.3) to pick the most discriminative features and remove the less significant features. This way, only the features which contribute most to the discrimination between classes, are kept and the used term set size is effectively reduced.

[6] Trademark of Adobe

The above indexing sequence works well for long documents, like for example scientific papers, which are stored as unstructured plain text. However, for catalogs with hypertext documents, such as HTML documents, special indexing techniques are required (see Section 6.3.1.1). However, our focus in this work is on unstructured documents.

6.5.2 User catalog training

In the catalog training step, an automatic text classifier is trained on the categories of the user catalog. The trained classifiers can then be used to calculate the classification results for a subset of documents from the community catalog and categories in the user catalog. These classification results serve as an initial connection between the categories.

The CAIMAN matching approach can use various approaches for classifier training, among which Naive Bayes, Support Vector Machines, k-Nearest-Neighbor, Rocchio/Cosine Similarity techniques (see [Sebastiani 2002] for the latter two) have been implemented in the CAIMAN mediation prototype (see Section 8.3). In our experimental results, however, we only present experimental evaluations for Naive Bayes and SVM as those two best represent the extremes in the trade-off of simplicity versus matching quality (see Section 6.3.3.2).

As neither Naive Bayes nor SVM classifier training can deal with hierarchical catalogs, catalogs have to be flattened for classifier training. In order to allow the distinction of categories with the same name label in the hierarchical catalog, the path from the catalog root to the category is used as category label in the flattened catalog (see Figure 6.4).

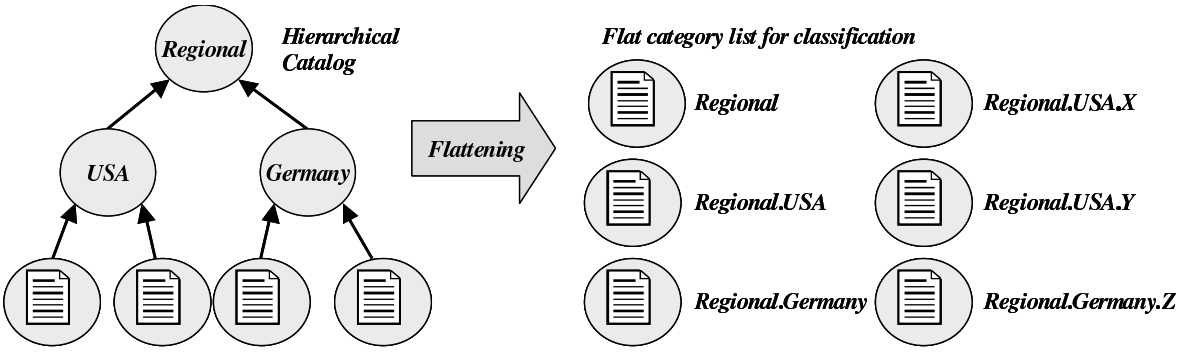


FIGURE 6.4: Flattening of hierarchical catalogs for classification

As the final catalog matching of the CAIMAN matching approach is symmetric, the classifiers could technically as well be trained on the community catalog(s). However, we do not assume any of the involved community/global catalog servers to be cooperative enough to offer document classification for large numbers of users. Moreover, instead of just one classifier for the user catalog, classifiers for each community catalog would have to be trained (see also Section 5.3.2).

6.5.3 Community document categorization

Preliminary categorization results. After the classifiers for categories in the user catalog have been trained, the classification results $\hat{\phi}(d_i, c_j)$ for documents from the community catalog can be calculated with the trained classifiers. We use the multi-category classifier adaptations presented in Section 6.3.4 in the CAIMAN approach. The classification results are seen as preliminary suggestions

for the assignment of community documents to user categories. The classification result is of a purely preliminary nature, because the final assignment will be determined by the final category match. Instead, the preliminary document assignments are used as the basis for the calculation of initial category similarities. For this purpose, each preliminary assignment of a document from category $c_i \in \mathcal{C}^c$ of the community catalog to a category $c_j \in \mathcal{C}^u$ in the user catalog is counted as a vote for the similarity between c_i and c_j (see Section 6.6).

Document subset categorization. We have seen in Section 5.3.1 that for a document granular mediation approach, all documents from the mediated catalogs have to be transferred to the user's workstation for classification. The same document transfer would actually be necessary for our category granular approach. However, we avoid this problem by simply only using a subset $\mathcal{D}'_i \subseteq c_i \in \mathcal{C}^C$ of all documents in each community category c_i for classification. This way, only the document subset $\mathcal{D}'_i$ has to be transferred to the user.

Since we are going to use the preliminary classification results to establish the initial category similarities, it is possible that the categorization of the document subset will lead to different initial similarity results than the categorization of all documents. However, we claim that a document subset sufficiently represents the whole category for classification for the following reasons: firstly, document classification itself is deterministic and is based on term statistics of the classified document and the trained categories. Thus, in the process of classifying a subset of documents from a category, only the choice of which documents to classify is random. Hence, the precise question that we need to answer is: can a subset of randomly picked documents of a category represent the category sufficiently well with respect to the overall term statistics of the category ? We can answer this question positively, using experimental evidence shown in Chapter 8. There, we will show that training a classifier on a subset of documents of a category is sufficient. As classifier training is also just based on term statistics of the documents, we can say that the experimental results show that a subset of documents in a category can represent the complete set of categories with regard to term statistics.

Even though we can say that only a document subset of each category has to be transferred in our category granular approach, we cannot say much about the fraction or even absolute number of documents to be transferred. Moreover, even if the decrease is small, a reduction of the classified documents will always reduce the matching quality. As different users will have different preferences concerning the document transfer / matching quality trade-off, we claim that choosing a fixed fraction of documents does not make sense. We propose a user-adjustable parameter for this purpose.

6.5.3.1 Context Aware Classification

In order to further improve classification quality, we use the approach of *context aware classification*, which has been proposed and demonstrated to bring a classification improvement for a Rocchio classifier in [Groh 2001] and [Lacher and Groh 2001]. We use the context aware classification approach with an SVM classifier, because the SVM classifier performs better than the Rocchio classifier.

The intuition behind the context aware classification approach is, that documents that are in the same source category as a document d_j should influence the classification decision for d_j, because they provide a valuable context for d_j.

For context aware classification, a community category representative vector is calculated for each community category. The context aware classification technique is applicable for all classification approaches that use document vectors for classification. The idea is to combine the category representative vector with each individual document vector as a linear combination of the two vectors

before classification:

$$\overrightarrow{d'_j} = \alpha \cdot \overrightarrow{c_i} + (1 - \alpha) \cdot \overrightarrow{d_j} \tag{6.22}$$

where d_j is the document vector, c_j is the category representative vector of the source category of d_j, and α is the weight of the category representative vector in the linear combination. The new document vectors $\overrightarrow{d'_j}$ are now used for classification.

We apply the context aware classification technique for SVM classification, which also uses document vectors for classification. We use the Rocchio centroid (see Section 6.3.3.3) as the category representative vector.

Note that the classification decision is still taken independently for each individual document. The classification decision is merely influenced by the linear combination. Thus, context aware classification can also be used for document granular mediation.

Having calculated the classification function $\hat{\phi}(d_i, c_j)$ for the subset $\mathcal{D}'_i$ of documents we are using in each category, we can now use the results as input to the category correlation phase.

6.6 Category correlation phase

In the category correlation phase, the initial category similarity values $\sigma_i(m_i)$ for all category pairs $m_i \in \mathcal{C}^u \times \mathcal{C}^c$ are calculated. The calculation is based on the classification results for the documents from the community catalog. Each document $d_y \in c_y$ with $c_y \subseteq \mathcal{C}^C$ that is preliminarily categorized into $c_x \in \mathcal{C}^U$ is counted as one vote for the similarity of c_x and c_y. The category similarity phase considers the community and user catalogs as flat lists of categories without catalog structure so far. For each category match pair $m_i = (c_x, c_y)$ with $c_x \in \mathcal{C}^U$ and $c_y \in \mathcal{C}^C$, the initial similarity value $\sigma_i(c_x, c_y)$ is calculated as follows:

$$\hat{V}(d_y, c_x) := \begin{cases} 1 & \text{if } \hat{\Phi}(d_y, c_x) = T \\ 0 & \text{otherwise} \end{cases} \tag{6.23}$$

$$\sigma_i(c_x, c_y) = \frac{\sum_{d_k \in c_y} \hat{V}(d_k, c_x)}{|c_y|} \tag{6.24}$$

where $|c_y|$ signifies the number of documents in category c_y.

For the Naive Bayes classifier approach, using the calculated probability values $P(C = c_j|d_i)$ instead of the function $\hat{\Phi}$ in Equation 6.24 would intuitively give a more accurate result: instead of a yes/no vote, a gradual agreement with values from $[0, 1]$ would be used. However, as mentioned in Section 6.3.3.1 the estimations in the Naive Bayes approach do not include the exact calculation of all terms. Since the CAIMAN prototype uses an external implementation of the Naive Bayes approach, we have little influence on the estimation procedures and it is safer to use the decision function $\hat{\Phi}'$ instead of $P(C = c_j|d_i)$.

The resulting category similarities σ_i can now be used in two ways. For catalogs without structure, i.e. which consist of a flat list of categories, the category $c_y \in \mathcal{C}^C$, for which $\sigma(c_x, c_y)$ is maximal among all $c_x \in \mathcal{C}^U$, can be picked as the final match for c_x. In this case, the matching process is finished. For catalogs with hierarchical structure, the initial similarity value pairs can now be fed into the structural matching phase for further refinement of the similarity values and final selection of the category matches by filtering.

6.7 Structural matching phase

The final phase in the CAIMAN catalog matching approach is the structural matching phase. The intuition behind the structural matching phase is that the calculation of category similarities we have presented so far is error-prone and ambiguous by nature. The initial similarity calculation in Section 6.6 may deliver erroneous results or several equally valid match proposals. The accuracy of the initial similarity values is dependent on the limited accuracy of the described text classification techniques. Thus, what is required, is some second source for information about category similarity.

In CAIMAN, no category labels are used for category matching, as we claim that category labels tend to be misleading and would often hurt the category matching accuracy more than improve it. However, the graph structure of catalogs expresses generic, catalog independent relationships between categories, which can be used for improvement of the match accuracy. Intuitively, errors or ambiguities in the similarity calculation for two categories can be amended, if the similarity of their local catalog graph neighborhoods is considered. Our aim is to exploit the catalog graph structures for improving the calculated similarity values. Hence, the category similarity values σ_i calculated in the category similarity phase are seen as a first approximation of the final similarity values σ, which are calculated in the structural matching phase.

Obviously, the more is known about the meaning of relationships between categories in the catalog graph, the more they can contribute to the correctness of the similarity calculation process. Unfortunately, unlike in formal ontologies (see [Noy and McGuiness 2001]), the categories and relations in document catalogs tend to have unclear semantics, as fuzzy relation meanings are more convenient for the user during definition as well as during usage. Moreover, even if there were formal semantics underlying the catalog graph, a user without educational background in Knowledge Representation is unlikely to describe the semantics of her catalog in a sufficiently formal way to allow methods of machine deduction to operate on them, as often proposed in ontology matching approaches (see Section 6.2). Figure 6.5 shows an example part of a typical document catalog, in which all relations are named *sub-category*, but their real formal meaning varies.

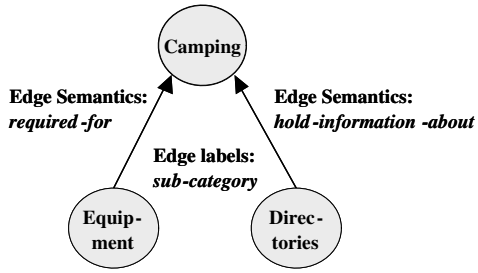


FIGURE 6.5: Ambiguous formal meaning of category relations with same label

The ambiguities of labels and meanings within a catalog graph are unavoidable. In addition, there are the differences in meanings between different catalogs. Thus, for our purposes we can only use information that is inherent and uniform in all catalogs. Due to the hierarchical structure of document catalogs, the notions of *parent* and *sibling* in a catalog graph are unambiguous. We use those tree-inherent notions for improvement of the category similarity values in the structural matching phase.

In order to be able to exploit the catalog graph structures, the catalogs first have to be prepared with the help of the user. These steps have partly already been described in Section 6:

- The user chooses one perspective in both matching candidate catalogs.

- The user identifies those edges in the match candidate catalogs, which can be subsumed under the *parent* notion (i.e. removing edges, which cross different perspectives in the catalog).

The result of these two steps are two catalogs, in which the *parent* of a node is unambiguously defined. The CAIMAN matching approach includes two different techniques to exploit the catalog graph structure for category matching. We are first going to present a simple structural approach to matching improvement that makes direct use of the *parent* relations in the catalog. Additionally, we are going to show how the *Similarity Flooding* [Melnik et al. 2002] algorithm, an elaborate graph matching algorithm, can be used to improve category matches in CAIMAN. Experimental evaluations for both approaches will be presented in Chapter 8.

6.7.1 Similarity estimate improvement through similarity inheritance

The intuition behind the simple structural matching algorithm is shown in an example in Figure 6.6. In the example, we aim to find a matching category in the community catalog for the category $c_G \in \mathcal{C}^u$ in the user catalog. After the calculation of the initial similarities, $\sigma_i(c_G, c_N)$ is maximal for all categories $c \in \mathcal{C}^c$. Let us assume that c_L would be the correct match, but the similarity $\sigma_i(c_G, c_N)$ is slightly higher than $\sigma_i(c_G, c_L)$. This can happen for example, if c_G, c_L and c_N all contain documents about cooking, but c_C and c_I contain documents about camping, whereas c_J contains documents about indoor hobbies in general.

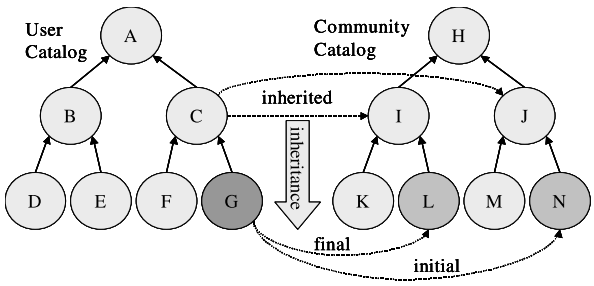


FIGURE 6.6: Intuition behind the simple similarity inheritance algorithm

Our simple similarity inheritance algorithm aims to improve the accuracy of the category matches by propagating similarities from the catalog root down to the leaves by *similarity inheritance*. The similarity inheritance algorithm adjusts the similarities of all initial category match pairs. We calculate the final similarity of a category match pair $m = (c_x, c_y)$, with $c_x \in \mathcal{C}^u$ and $c_y \in \mathcal{C}^c$, given its initial similarity, in the following way:

$$\sigma_{final}(c_x, c_y) = \alpha_i \cdot \sigma_{initial}(c_x, c_y) + (1 - \alpha_i) \cdot \sigma_{inherited}(c_x, c_y) \quad (6.25)$$

$$\sigma_{inherited}(c_y, c_y) = \sigma_{final}(parent(c_x), parent(c_y)) \quad (6.26)$$

where α_i is a parameter, which adjusts the weight of the inherited similarity in contrast to the initial similarity. In our example in Figure 6.6, the similarity $\sigma(c_C, c_I)$ is higher than $\sigma(c_C, c_J)$. These two similarities are inherited down the tree, such that the final category match pair with the highest similarity is (c_G, c_L). The similarity inheritance algorithm has no restrictions in terms of catalog structure of the matched catalogs. Differences in the catalog structures between two matched catalogs do not hinder the calculations. The final similarity is calculated for all category match pairs in the catalog matching.

The simple similarity inheritance approach considers the catalog structures of the match candidate catalogs in a limited way. The only graph features that influence the similarity of two categories are the respective paths from the match candidate categories to the catalog root category. This root path arguably covers the most important neighborhood information for a category, but other neighbors of a category are potentially important for the match decision as well. Taking into account a larger graph neighborhood requires a full-fledged graph matching algorithm. This issue is tackled in the next section, in which we explain how the versatile graph matching algorithm *Similarity Flooding* [Melnik et al. 2002] can be applied in the CAIMAN catalog matching approach.

6.7.2 Similarity estimate improvement through Similarity Flooding

The simple similarity inheritance algorithm described above considers only parts of the graph structures of two match candidate catalogs to improve the match. Moreover, its output is most likely not a globally optimal match for all categories in the catalogs. To amend these two shortcomings of the simple structural matching approach, we apply a versatile graph matching technique, which has been presented in [Melnik et al. 2002], for structural catalog matching. We are going to show how this technique, called *Similarity Flooding* can be employed in the context of the CAIMAN catalog matching technique.

The Similarity Flooding algorithm divides the matching process into two phases:

1. Improvement of the initial similarity values
2. Filtering of the resulting category match pairs.

Input to the Similarity Flooding approach is a cross product $\mathcal{C}^u \times \mathcal{C}^c$ of the set of categories in the user catalog $c_x \in \mathcal{C}^u$ and the set of categories in the community catalog $c_y \in \mathcal{C}^c$ with respective initial similarity values $\sigma_{initial}(c_x, c_y)$. The initial similarity values $\sigma_i(c_x, c_y)$ for elements (c_x, c_y) in the cross product are the output of the category similarity phase. The improvement phase attempts to improve the similarity values by propagating similarities in the full catalog graph structure on both sides. The filtering phase then filters out one or several match pairs, depending on the requirements of the respective CAIMAN knowledge exchange service (see Section 4) and/or considering global optimality and other constraints.

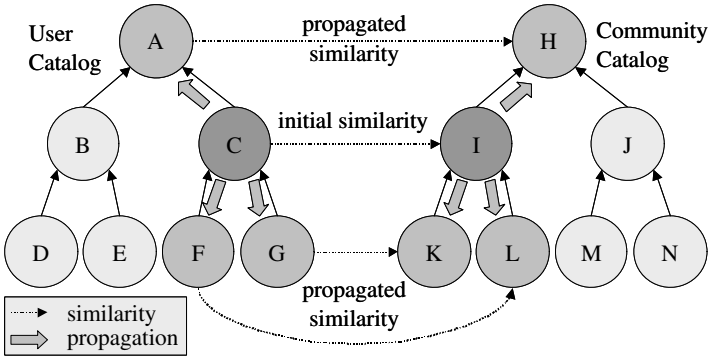


FIGURE 6.7: Intuition behind the Similarity Flooding algorithm presented in [Melnik et al. 2002]

Figure 6.7 shows that categories c_C and c_I exhibit an initial similarity of $\sigma_{initial}(c_C, c_I)$. The intuition behind the Similarity Flooding algorithm is, that if c_C and c_I are similar, probably graph neighbors of c_C are similar to graph neighbors of c_I. Thus, the similarities of categories are propagated

in the graphs by the Similarity Flooding algorithm to the category pairs (c_A, c_H) or (c_F, c_L) for example. Abstracting from the example in Figure 6.7, the Similarity Flooding algorithm propagates all initial similarities in the graphs to neighboring category pairs. How this propagation is achieved and what neighboring category pairs means exactly, will now be examined in more detail.

Improvement of Similarity Values. In order to explain how the Similarity Flooding algorithm can be applied in the CAIMAN category matching approach, it is first necessary to explain some more details about the Similarity Flooding algorithm. Figure 6.8 describe the different steps of the Similarity Flooding algorithm by an adapted example from [Melnik et al. 2002].

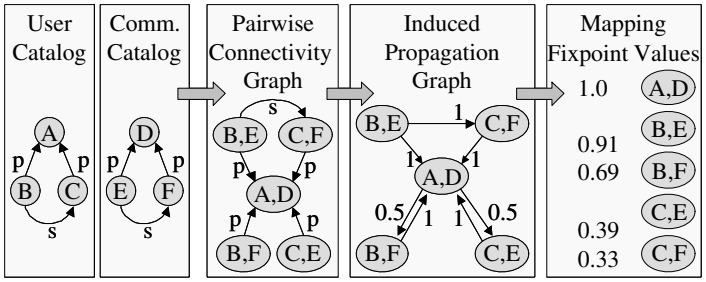


FIGURE 6.8: Similarity Flooding example pass (adapted from [Melnik et al. 2002])

Pairwise connectivity graph. On the left hand side of Figure 6.8, two small example catalogs can be seen. The example catalogs have edges labelled p for *parent* and s for *sibling*. In the first step, a so-called pairwise connectivity graph (PCG) is created. The PCG consists of the following elements: $((c_x, c_y), e, (c'_x, c'_y)) \in PCG(\mathcal{C}^u, \mathcal{C}^c) \iff (c_x, e, c'_x) \in \mathcal{C}^u \; and \; (c_y, e, c'_y) \in \mathcal{C}^c$. Each node in the PCG is called a map pair $m \in \mathcal{C}^u \times \mathcal{C}^c$. Map pairs are connected by edges, which indicate that two map pairs are *neighboring* map pairs, and their similarity should influence each other. As can be seen from the formal definition of the PCG elements, two map pairs are considered neighboring, if and only if the involved original categories are connected with an edge, which has the same label in both the user and the community catalog.

Induced propagation graph. The next step is the creation of a so-called induced propagation graph (IPG). The induced propagation graph represents the directions and weights of the propagation of similarities among the map pairs. The IPG is created from the PCG in the following way: for each directed edge in the PCG, the IPG includes the original edge as well as an additional edge connecting the same map pairs with opposite direction. Each edge is attributed with a propagation coefficient in $[0, 1]$, which signifies how well the similarity of a map pair is propagated along the respective edge. One possibility to calculate the propagation weights is to evenly distribute a weight of 1 among all outgoing edges of a map pair, which have the same label. The calculation of the weights is explained in more detail in [Melnik et al. 2002].

The propagation of similarities in the graph is the next step in the Similarity Flooding approach. The Similarity Flooding algorithm iteratively calculates a series of $\sigma^i(c_x, c_y)$ values for $i = 0 \cdots n$ for all map pairs in the induced propagation graph. The iteration terminates, when the Euclidean length of the residual vector $\sigma^n - \sigma^{n-1}$ becomes less than a given constant ϵ, i.e. if none of the σ values changes substantially from one iteration to another. In each iteration step, the similarity

value $\sigma^i(c_x, c_y)$ of a map pair (c_x, c_y) is calculated as the weighted sum of the map pair's previous similarity value $\sigma^{i-1}(c_x, c_y)$ and the previous similarity values $\sigma^{i-1}(c_u, c_v)$ of all neighboring map pairs in the IPG. More details on the fixpoint computation and different weighting schemes can be found in [Melnik et al. 2002]. After the iteration terminates, the algorithm outputs the final similarity values for all map pairs in the PCG, as can be seen on the left hand side of Figure 6.8[7].

The Similarity Flooding calculation in CAIMAN requires two important inputs: the initial map pair similarities and the graph structures of the catalogs to be matched. The initial map pair similarities, which have been calculated in the category similarity phase, are used as σ^0 values in the similarity flooding iterations. The input catalog graph structures originally have edges labelled p for *parent* only. However, if only the p edges are used, from a Similarity Flooding perspective, a sibling category and a grand-parent category would be indistinguishable: both are two p edges away. In order to increase the influence of siblings in the Similarity Flooding process, we introduced extra s (for *sibling*) edges in the catalog graphs. Sibling edges are created between each two categories on the same catalog tree level, resulting in a total of $\frac{n \cdot (n-1)}{2}$ extra edges per tree level with n category nodes. As can be seen in Figure 6.8 in the IPG, the extra sibling edges lead to a direct similarity propagation edge between two siblings. The fixpoint calculation of the improved similarity values is shown to converge in [Melnik et al. 2002].

We have presented two structural matching approaches in this section. So far, we have calculated an improved similarity measure $\sigma(c_x, c_y)$ for all category pairs $(c_x, c_y) \in \mathcal{C}^U \times \mathcal{C}^C$. The final step in our matching approach is now to filter out the right category according to the respective knowledge exchange service.

6.7.3 Match candidate filtering

Depending on the CAIMAN application service, for which the catalog matching is later used, a set of final catalog matching results have to be filtered out from the cross product of matching candidates. The criteria for the matching filters in CAIMAN, some of which have been presented in [Melnik et al. 2002] are:

1. Cardinality constraints on the number of matching partners for one category.
2. Minimum match pair similarity constraints.
3. The Stable Marriage constraint, which enforces that the match pairs are picked in a globally optimal way.

Let us look at those different filter criteria in the CAIMAN matching approach in more detail. We are first going to describe the basic filter mechanisms we apply and thereafter define four different match filter scenarios, which are tailored to the requirements of the CAIMAN knowledge exchange services (see Chapter 4). The result of the previous steps in the CAIMAN matching approach and thus the input to the filtering step is a cross product of map pairs $M = \{m_i | m_i \in \mathcal{C}^u \times \mathcal{C}^c\}$ with the respective similarity values $\sigma(m_i)$, which we call *matching*.

Minimum similarity. The simplest way to constrain the match cardinality is to introduce a similarity threshold σ_{min}, such that for similarities $\sigma(m_i) < \sigma_{min}$ the respective match pair m_i is excluded from the matching. Using the minimum similarity as a parameter is a means of balancing recall and precision of a service.

[7] The values shown are not actual simulation results but arbitrary numbers for illustration purposes.

Cardinality constraints. Among the possibilities to reduce the number of map pair candidates, which are mentioned in [Melnik et al. 2002], are matching cardinality constraints. Cardinality constraints basically provide lower and upper bounds on the number of matching partner categories any given category is allowed to have in the final matching. Let us consider an example matching between a user catalog and a community catalog. A cardinality constraint of $[0,1]-[0,n]$, would denote that every category from the user catalog may have an arbitrary number of match partners in the community catalog. On the other hand, not every category in the user catalog must necessarily have a partner category in the community catalog and vice versa. However, a category in the community catalog may also have at most one match partner in the user catalog. Cardinality constraints can be used to make sure that each document is uniquely categorized.

Stable marriage constraint. Even if the cardinality constraints are such that each category from each of the two catalogs can only have exactly one matching category, the matching is still not precisely defined. Consider the example we have presented in Figure 6.8. If a $[1,1]-[1,1]$ constraint is applied, the matching possibilities are either $M_1 = \{(A,D),(B,E),(C,F)\}$ or $M_2 = \{(A,D),(B,F),(C,E)\}$. If B gets to pick its partner first, the result is M_1 and if C gets to pick its partner first, the result is M_2. One of the two possible matchings can be picked by enforcing the stable marriage constraint [Melnik et al. 2002].

Definition 6.8 (Stable Marriage Matching) *A stable marriage matching is defined as a complete matching M with the property that there are no two map pairs (c_x, c_y) and (c'_x, c'_y), such that x prefers y' to y and y' prefers x to x'.*

Going back to the above example, only M_1 fulfills the stable marriage constraint and would thus be picked as the final matching.

Having explained how the different filter criteria work, we can now define the filter function $\mathcal{F}^K$ for each knowledge exchange service K by means of different scenarios. Based on the filter constraints proposed in [Melnik et al. 2002] and presented above, we define the following filter scenarios for application with the CAIMAN knowledge exchange services:

- **Publication Filter**: This filter is used for the *information publication* service described in Chapter 4. In the *information publication* service, documents from the user catalog are published to a partner catalog. The catalogs, which we consider in this work, adhere to the constraint, that each document can only be assigned to one category. However, documents from several user categories may well be published into the same destination category. In this case, there are several user categories that match with one community category, as shown in Figure 6.9. Thus, we set the filter constraint to $[0,n]-[0,1]$. Additionally, to balance recall and precision of the service, we apply a minimum similarity constraint. Thus, the filter function $\mathcal{F}^P(M)$ has a value of T, if the following holds: 1) in the matching M, each category $c_i \in \mathcal{C}^U$ has at most one matching partner $c_j \in \mathcal{C}^C$, and 2) $\forall m_i \in M : \sigma(m_i) \geq \sigma_{min}$. A schematic overview of the *publication filter* can be seen in Figure 6.9.

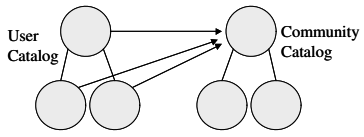


FIGURE 6.9: Filter cardinalities for the publication service

- **Related information retrieval filter**: This filter is used for the *related information retrieval* service, as described in Chapter 4. In the *related retrieval* service, documents from the community catalog are retrieved and presented in respective categories in the user catalog. Again, each retrieved document can only be assigned to one category. However, for a category in the user catalog, several categories in the community catalog may be relevant. We apply a $[0,1]-[0,n]$ filter and a minimum similarity constraint as described above with the publication filter. Thus, the filter function $\mathcal{F}^R(M)$ has a value of T, if the following holds: 1) in the matching M, each category $c_j \in \mathcal{C}^C$ has at most one matching partner $c_i \in \mathcal{C}^U$, and 2) $\forall m_i \in M : \sigma(m_i) \geq \sigma_{min}$. A schematic overview of the *retrieval filter* can be seen in Figure 6.10.

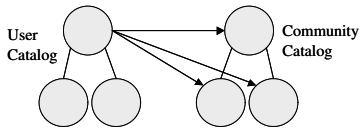


FIGURE 6.10: Filter cardinalities for the related information retrieval service

- **Category discovery filter**: This filter is used for the *category discovery* service, as described in Chapter 4. In the *category discovery* service, categories in the community catalog that are related to a category in the user catalog are discovered and presented to the user. There are no documents assigned in the category discovery service, thus there is no constraint on the cardinality of matches on each side. We apply a $[0,n]-[0,n]$ filter with a minimum similarity constraint as described in the publication filter description. Thus, the filter function $\mathcal{F}^D(M)$ has a value of T, if $\forall m_i \in M : \sigma(m_i) \geq \sigma_{min}$. A schematic overview of the *category discovery filter* can be seen in Figure 6.11.

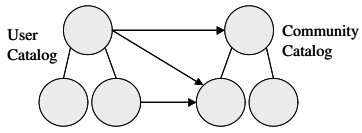


FIGURE 6.11: Filter cardinalities for the category discovery service

- **Stable Marriage Filter**: This filter can be used for all three CAIMAN knowledge exchange services. There is a one-to-one match between the categories, thus the filter has a $[1,1]-[1,1]$ cardinality. Additionally, the *stable marriage* constraint is enforced. Thus, the filter function $\mathcal{F}^S(M)$ has a value of T, if the following holds: 1) in the matching M, each category $c_j \in \mathcal{C}^C$ has exactly one matching partner $c_i \in \mathcal{C}^U$ and vice versa, 2) the stable marriage constraint holds as defined in definition 6.8. A schematic overview of the *stable marriage filter* can be seen in Figure 6.12.

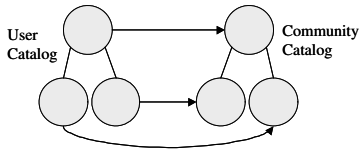


FIGURE 6.12: Filter cardinalities for the stable marriage filter

In the last two sections, we have explained how the similarity flooding graph matching algorithm and additional match filtering can be applied in the CAIMAN matching approach. Matching quality evaluation results for different applications, a small catalog matching example among others, have been presented in [Melnik et al. 2002]. The experimental results show, that the similarity flooding algorithm can significantly improve the average accuracy of matches. However, the accuracy is not high enough to allow for a totally automatic matching - in most cases, interaction with the user is required. In [Melnik et al. 2002], the workload for the user to create a matching from scratch has been compared to the workload for the user, if the matching results of a Similarity Flooding are given to the user for correction or completion. It has been found that the average workload could be reduced significantly. In Chapter 8, we are going to present experimental results, which show that Similarity Flooding in combination with match filtering can bring significant improvement in the CAIMAN matching approach.

6.8 Summary

In this chapter, we have presented the novel CAIMAN approach to document catalog matching. The presented matching technique is the basis for semi-automatic category granular mediation of document catalogs, as required by the CAIMAN knowledge exchange services (see Section 4). The CAIMAN matching technique is especially suitable for our community scenario, as it satisfies the application requirements of the knowledge exchange services (see Section 5.1.2).

We have presented the theoretical background from the fields of Information Retrieval and Machine Learning, which is the necessary basis for the description of the CAIMAN matching approach. The CAIMAN approach employs text classification techniques, which have been described along with suitable performance measures in Section 6.3.

Our new catalog matching approach consists of a sequence of three phases: the **document categorization phase**, the **category correlation phase** and the **structural matching** phase. In the categorization phase, documents from each category of the community catalog are temporarily assigned to categories in the user catalog using techniques for automated text categorization.

The classification results are fed into the *category correlation phase*. In the category correlation phase, the initial similarity values for match pairs of categories from the user and the community catalog are calculated, based on the document classification results.

The initial similarity values are in turn fed into the *structural matching phase*. In the structural matching phase, match pair similarities are propagated among category pairs according to the graph structure of the catalogs. The final match pairs are filtered out in the match candidate filtering step. A match filter enforces service-specific matching constraints. The outcome of the filtering step is a final catalog matching, which is specific to a certain knowledge exchange service.

Chapter 7

Querying Heterogeneous Document Catalogs with CAIMAN

7.1 Introduction

This chapter deals with the catalog querying approach in the CAIMAN framework and thus details aspects of the mediation principle from Chapter 5 . The catalog querying approach allows the knowledge exchange services to query the different catalogs in our scenario for their structure and their contents. An example structure query would be "all sub-categories of category c_i" and an example content query would be "all documents in category c_i". The focus of this chapter is on an approach for joint querying of heterogeneous catalogs on a data model level, and especially on the conversion of different data models with wrappers. We also briefly discuss an approach for query mapping that serves as a simple proof of concept for our general querying approach. Our querying approach concerns functionalities of the query processing and the mapping component of the CAIMAN framework. The query processing component also uses the matching results that have been calculated by the matching component.

This chapter is organized as follows. First, we are going to present some necessary theoretical background about the KR formalisms RDF[1] [Lassila and Swick 1999] and Topic Maps [Pepper and Moore 2001] in Section 7.2. Both RDF and Topic Maps play an important role in this chapter.

In Section 7.3, we present the core principles that we applied in the CAIMAN querying approach. RDF is used as the common data format, to which all other formats are converted. All query processing can then performed by an existing RDF query processor.

We present some related querying and data model conversion approaches, especially for Topic Maps and RDF, in Section 7.4.

As a proof of concept of our querying approach, we show in Section 7.5 how joint querying of RDF based catalogs and Topic Maps based catalogs can be realized. To achieve this interoperability, we show how Topic Maps data can be modelled with RDF without loss of information. RDF and Topic Maps are currently the two most widely used data models for the representation of document catalogs on the Web.

Finally, we are going to present a concrete example for joint querying of the RDF based *Open Directory*[2] web catalog and the Topic Map based CIA World Factbook[3] in Section 7.6. We show how

[1] Resource Description Format

[2] See http://www.dmoz.org/

[3] See http://www.cia.gov/cia/publications/factbook/

the catalog data can be converted and queried using our approach.

Specification of the querying problem

We differentiate between two different types of catalogs available for personal information management as well as on the Web today and in the near future:

- **Informal catalogs.** In the first catalog type, which we call informal catalogs, the explicit catalog structure is not directly accessible for querying. Examples for *informal catalogs* are folder hierarchies in a file system or bookmarks folders for a browser and web catalogs like *YAHOO* and *Altavista*. All of these catalogs require some kind of preprocessing to make the catalog structure explicit and available for querying, such as in a database. In this work, we assume that the explicit structure of all informal catalogs has been made available either through preprocessing or through specialized *wrappers*. Preprocessing or wrapping of informal catalogs is not the focus of this work, more details can for example be found in [Raghavan and Garcia-Molina 2001a].

- **Semi-formal catalogs** represent the second type of catalog that can be found on the web today. Following efforts of turning the Web into a large knowledge base - a so-called *Semantic Web* [Berners-Lee et al. 2001] - document catalogs are represented in a semi-formal way on the Web. Knowledge Representation (KR) formalisms, that have been adapted for use on the Web, have recently been used to explicitly describe the structure and contents of document catalogs.

Our focus in this chapter are semi-formal catalogs with explicitly represented structure as well as informal catalogs, the structure of which has been made accessible through preprocessing.

The query component of the CAIMAN infrastructure needs to be able to query the different user, community and global catalogs (see Figure 7.1). There can be a number of technical differences between the various catalogs, some of which are not within the scope of this work. For our purposes, we assume, that all catalogs are physically accessible via the Internet. Moreover, we assume that low-level interoperability issues are resolved through XML[4] representation of catalog data.

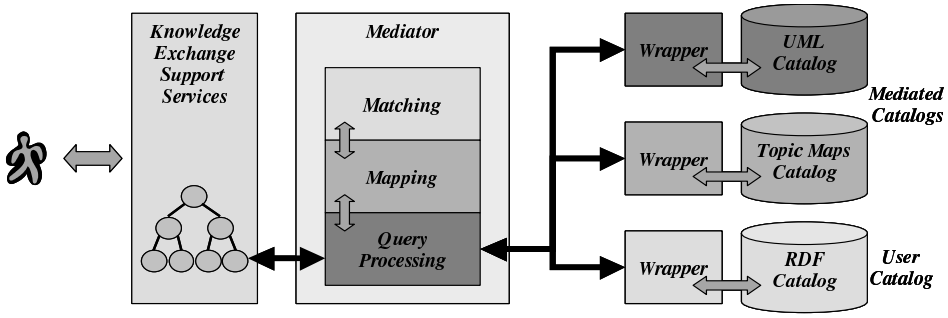


FIGURE 7.1: Problem scenario: Querying heterogeneous catalogs

The different catalog data sources in Figure 7.1 can in principle be jointly queried on a syntax level with a query language like XQuery[5] for XML data. However, querying becomes easier, if queries can be posed on a higher semantic level and issues of syntax become transparent. Another

[4] eXtensible Markup Language

[5] See http://www.w3.org/TR/xquery/

advantage of higher level queries is, that we need not differentiate between different syntaxes of a data format. Ideally, a query infrastructure in our scenario would for example allow a simple query for "all documents that are related to category c_i in the user catalog" and deliver results from all mediated catalogs, independent of their data model and syntactic representation. The differences between the data sources have to be overcome in some way. Describing the necessary transformations in a declarative way rather than in a procedural way is commonly seen as more transparent, more flexible and thus easier for the user.

Our foremost goal for this chapter is to simplify joint querying of catalogs by allowing queries with more semantics than pure syntax queries. Additionally, although not the primary focus of this chapter, we want to provide a coarse exemplary concept how queries can be mapped to overcome the remaining differences between catalog data sources. Preferably, the query mappings should be specified in a declarative fashion.

The main difference of our goals here to the goals of other integration efforts like for example TSIMMIS [Garcia-Molina et al. 1995] is that we do not aim to describe a full-featured mediation infrastructure. The aspect of query mapping is only briefly described to prove that the rest of our concepts are viable. Query mapping in general is a broad topic, which has been discussed extensively in [Halevy 2001], for example. On the one hand, our approach is also targeted specifically at document catalogs and is thus focused on a narrower, more specific aspect than TSIMMIS. On the other hand, some aspects of our solutions here can be seen as an application of certain TSIMMIS concepts. Another difference is that TSIMMIS also provides concepts for an automatic generation of interoperation solutions from formal problem definitions. We are going to present some more differences to existing approaches in Section 7.4.

7.2 Theoretical background

We base the presentation of related work on a layered model of the future *Semantic Web*, presented in [Berners-Lee 2000]:

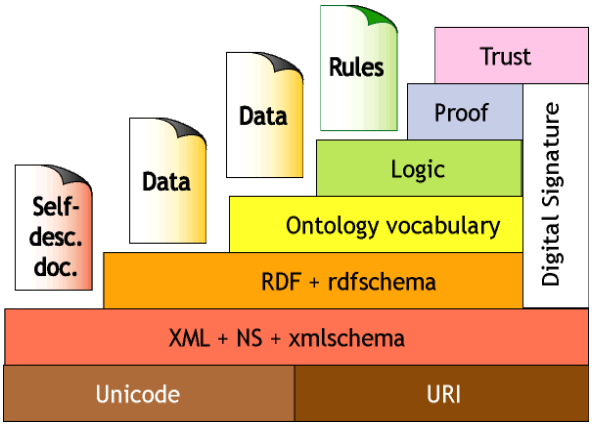


FIGURE 7.2: Conceptual overview of the layers of a future *Semantic Web* [Berners-Lee 2000]

Figure 7.2 shows the basic building blocks of a Semantic Web infrastructure. A detailed description of the vision of the Semantic Web can be found in [Berners-Lee 2000; Berners-Lee et al. 2001]. For our purposes here, it suffices to understand the Semantic Web as a giant knowledge base of semi-structured data. For this chapter, we further assume some basic working knowledge of the standards shown in the bottom two layers of the stack in Figure 7.2. The catalog query problem and its solution

are on layer three of the stack in Figure 7.2. Layer three is concerned with data model issues like how object identity is established or how binary relations are represented. Everything above layer three, which concerns the semantics of catalog representations, is not considered in this chapter. In this chapter, we present how a generic approach to data interoperability can be applied to the catalog query problem. We further present a proof of concept for our catalog query concept by showing how RDF and Topic Maps data can be jointly queried.

7.2.1 RDF and related W3C standards

RDF is a data model for semi-structured data. The Resource Description Framework Model and Syntax Specification [Lassila and Swick 1999], which became a World Wide Web Consortium (W3C) Recommendation in February 1999, defines the RDF data model and a basic serialization syntax. The RDF Data model is essentially a directed, labelled graph: it consists of entities, identified by unique identifiers, and binary relationships between those entities. In RDF, a binary relationship between two specific entities is represented by a statement (or *triple*). An RDF statement can be represented in a graph as two nodes and a directed edge between the nodes. The node with the outgoing edge is called *subject* of the statement, the edge is called *predicate* and the node with the incoming edge is called *object* of the statement. The RDF data model distinguishes between resources, which have URI identifiers, and literals, which are just strings. The subject of a statement is always a resource. The predicate of a statement is a member of a subset of the set of resources, called *properties*. The object of a statement can be either a resource or a literal.

RDF has several possible syntaxes for serialization, among which there is one basic and one abbreviated syntax defined in [Lassila and Swick 1999]. RDF can have an XML syntax as well.

The most basic use of RDF is to establish object identity and represent binary relations between objects. Using additional vocabulary from schema languages like RDFS[6] or DAML+OIL[7], RDF can be used to define a category system that is required for a catalog (see [Staab et al. 2000c] for more information on what a schema language is).

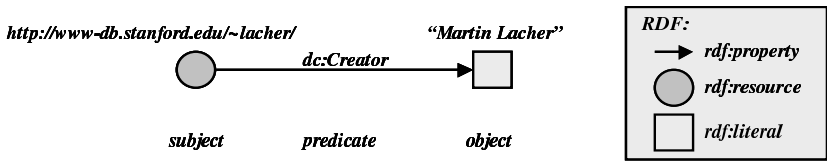


FIGURE 7.3: Example statement in the RDF data model

7.2.2 Topic Maps

Topic Maps were originally designed to serve as complex, multi-dimensional indexes for document collections. However, just like RDF, Topic Maps can also simply be seen as a data model. Topic Maps [Biezunski et al. 1999] have been standardized by the ISO in 1999. A Topic Map is defined as a collection of Topic Map documents, which adhere to a certain SGML[8] syntax that is defined in the standard document. The SGML Syntax of those documents is described along with an informative conceptual model for memory representation of Topic Maps. Topic Maps can be used as a format for

[6] Resource Description Format Schema
[7] DARPA Agent Markup Language + Ontology Inference Language
[8] Standard Generalized Markup Language

the representation of multi-dimensional subject-based indices for document collections. Topic Maps can also be used as a format for interoperable knowledge representation.

The original ISO standard specified an SGML syntax for the exchange of Topic Maps. To make Topic Maps applicable on the Web, the XML Topic Maps standard (XTM) has been drafted [Pepper and Moore 2001]. XTM defines an XML syntax for Topic Maps and gives a specific, albeit slightly simplified, data model of a Topic Map. Both the SGML syntax and the XML syntax incorporate syntax shortcuts for complex data model constructs. The XML syntax presented in [Pepper and Moore 2001] has been included into [Biezunski et al. 1999] after publication.

Several data model representations have been proposed for Topic Maps. We will adhere to the graph representation described in [Biezunski and Newcomb 2001]. This graph representation knows four different kinds of edges and three different kinds of nodes. The different nodes can each only have edges of a certain type attached. Additionally, each node can have several *subject identity points*. Subject identity points can serve as unique identifiers for the nodes.

The semantics of the graph nodes defined in the data model is that of subjects, defined as anything that can be referred to in human discourse. These subjects are divided into *topics*, *associations* and *scopes*, corresponding to the three different node types. Topics can have a number of characteristics, which can be bound to them by means of associations. The processing models described in [Biezunski and Newcomb 2000] and [Biezunski and Newcomb 2001] state some semantic constraints on the graph, which have to be enforced in order to produce a *consistent* Topic Map. Basically, these constraints ensure that no duplicate topics occur in a consistent Topic Map.

7.3 General approach for querying heterogeneous document catalogs

7.3.1 RDF as lingua franca for catalog data

Our goal is to make different catalog data sources queryable with the same query infrastructure. Generally, interoperability among n different data sources can be achieved in two ways:

- with a query infrastructure that converts queries and data between all n different formalisms and thus requires $n \cdot (n-1)/2$ converters, or
- with a query infrastructure that converts all queries and data to one *lingua franca* data model and thus requires only $n-1$ converters.

All model-based data, and thus also document catalog data, can be represented as *semi-structured data*. The concept of semi-structured data has been identified in the database community [Hammer et al. 1997; Suciu 1998] as a means for data integration [Garcia-Molina et al. 1995], [Papakonstantinou et al. 1995] and transformation [Abiteboul et al. 1997]. RDF can be seen as a simple formalism to represent any kind of semi-structured data and is thus suitable as the lingua franca for all data in our scenario. The use of RDF for the representation of semi-structured data implies a very general perspective on RDF: RDF can be seen as a simple way to express object identity and binary relations between objects - independent of what these objects mean.

Thus, one cornerstone of our joint query component is the conversion of all other data models to RDF. An RDF query infrastructure can then be used to jointly query all catalogs.

7.3.2 Modelling heterogeneous catalog representations with RDF

There are two principal approaches for the conversion of data models, which can be characterized as *translating* versus *modelling* [Moore 2001].

The translation approach attempts to translate atomic building blocks of one data model into atomic building blocks of another data model. After the conversion, the nature of the original data model is completely transparent. Consider, for example, the conversion of a relation between two categories in a catalog from one data model into another. How the relation had been represented before the conversion is not visible anymore, once the conversion has been performed. The advantage of the translation approach is that no details about the different source data models have to be known. However, the translation may incur loss, because the modelling primitives of two formalisms will not have the exact same semantics. Loss also makes an inverse mapping impossible. Especially, if the application domain of the data is not known in advance, loss is not acceptable.

The modelling approach attempts to model the semantics of one data model with the building blocks of another data model. With the modelling approach, the nature of the original data model is conserved. Consider again our example with the relation between two categories in a catalog. How this relation had been represented before the conversion remains visible, the representation is just modelled with another data model. We think that the modelling approach is preferable, because in contrast to the translation approach, the modelling does not incur any loss in the transformation from one data model to another.

7.3.2.1 Layered approach to data interoperability

Interoperation of different data formats is a complex problem. The layered model of data interoperability in [Melnik and Decker 2000] breaks up the problem of data model interoperation into a stack of layers, which have a high level of independence from each other. This approach resembles the ISO/OSI protocol stack for network interoperation. The approach presented in [Melnik and Decker 2000] concerns the bottom three to four layers in Figure 7.2. The different layers presented in [Melnik and Decker 2000] are bottom up: the syntax layer, the object layer and the semantic layer. Each of those layers actually has sub-layers, but we do not require such a detailed perspective on the layers here. The syntax layer is concerned with a serialization syntax for persistent storage and transportation of data. The object layer is concerned with how to assign identity to objects or how binary relations are represented. The semantic layer is concerned with the interpretation of the objects and their relationships. Figure 7.4 shows an overview of the issues that are relevant on each layer of the layered interoperability model.

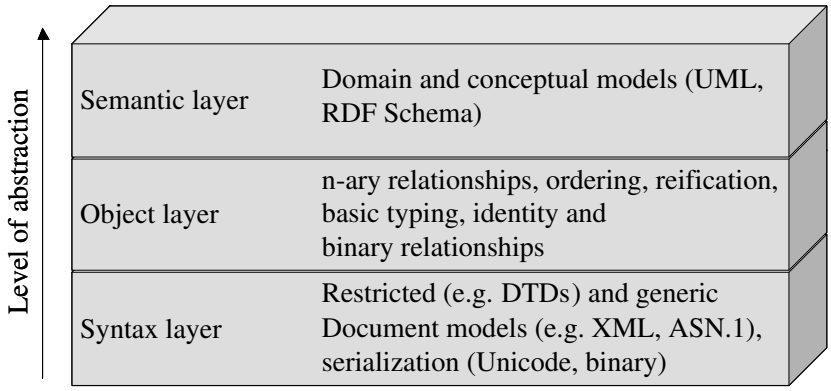


FIGURE 7.4: The layered interoperability model [Melnik and Decker 2000].

The important essence of our approach is that we perform a graph transformation on the object

layer, which can be performed quasi-independently from the other layers. This independence is possible, because any semi-structured data model [Suciu 1998] can be represented as a directed graph, which is also the data model of RDF. Thus, any kind of semi-structured data model can be represented by RDF on the object layer. How the RDF graph is interpreted on a higher level may vary again for different data models. In this work we are not going to consider the issue of mapping those higher level semantics. We are only going to look at RDF as the common denominator for data representation and query purposes. By using the modelling approach, the Topic Map semantics on a higher level is conserved in the mapping and only the representation on the object layer is mapped to RDF.

7.3.3 Joint querying of document catalogs using the TRIPLE query language

We have chosen RDF as the common representation for all catalog structure data in CAIMAN. Thus, we require a suitable RDF query language and processing engine. The RDF query processor has to be able to query different distributed RDF data sources that have various higher-level semantics. Moreover, the query mappings based on the catalog matching results have to be performed.

The TRIPLE query language. We chose the *TRIPLE* RDF query infrastructure [Sintek and Decker 2002] for catalog querying in CAIMAN. TRIPLE is a query engine and rule language that can query RDF data sources by making use of PROLOG and Description Logic reasoning engines. The advantage of *TRIPLE* over other query languages is, that it can query heterogeneous data sources with different semantics. The semantics of the data sources can be encoded in *TRIPLE rules*, which are organized in so-called *models* for each individual data source. We are going to present, how the TRIPLE rules and engine can be used to perform the required query mapping and processing in our catalog scenario.

Query mapping and processing. For joint querying of the various catalog sources as depicted in Figure 7.1, queries over the user catalog have to be mapped to queries over the mediated catalogs. Then, the mapped queries can be processed over the respective catalogs and the query results can be integrated by the query processing engine.

For the mapping of queries, categories and relations between categories have to be mapped. The queries we consider here are simple queries, for example “all sub-categories for category A” or “all documents that belong to category A”. These simple queries are sufficient for our purposes. The mapping of categories is straightforward and uses the matching results from the catalog matching process. A user category in a query is mapped to all matching categories in one mediated catalog. This mapping can be encoded in TRIPLE rules within the TRIPLE model for the respective mediated catalog. The rules can be generated automatically, based on the catalog matchings. As an example for a category mapping, consider a query for all documents in a category labelled *software* in the user catalog. The matching categories in the community catalog are labelled *databases*, *office software* and *web publishing*. Thus, a set of TRIPLE rules are generated in the TRIPLE model for the community catalog, which state that the categories *databases*, *office software* and *web publishing* in the community catalog have to be queried for documents. An example for category mapping rules can be found in Figure C.2 in Appendix C.

Since our approach for joint querying includes the low-level modelling of different catalog data models as RDF, the higher-level semantics of the different data models remain different. For example, a *sub-category* relation is represented by a single predicate in the RDF model of the Open Directory[9]

[9] See http://www.dmoz.org/

web catalog. In the RDF-based Topic Map of the CIA Factbook catalog[10], a *sub-category* relation is represented by several relations, which together represent a *class-instance* relation. Thus, the simple *sub-category* relation needs to be mapped to the more complex *class-instance* relation. Again, this mapping can be defined using TRIPLE rules in a TRIPLE model for the respective catalog source.

To keep the user workload for this mapping as low as possible, we divide the required rules into two parts: one general part for the respective data model that is used in the catalog source and one that is specific for a certain catalog. In the example above, we would have one CIA Factbook specific set of rules that maps a *sub-category* relation to a *class-instance* relation. Moreover, we would have one general Topic Map set of rules that maps a *class-instance* relation to the set of relations that generally represent *class-instance* in Topic Maps. The advantage of this approach is that the general part of the rules can be defined in advance by an administrator and the user does not have to be concerned with that. Thus, if a new catalog is added for mediation, the user only needs to supply the information which relation in the new catalog should be interpreted as the *sub-category* relation. The same procedure that we have described for *sub-category* relations also applies for *contains* relations between categories and their contained documents. An example for relation mapping rules can be found in Figure C.2 in Appendix C.

7.3.4 Realization of the query component

We have presented our general query approach. Figure 7.5 shows how we envision the query component to be realized and how it interacts with the various catalog sources.

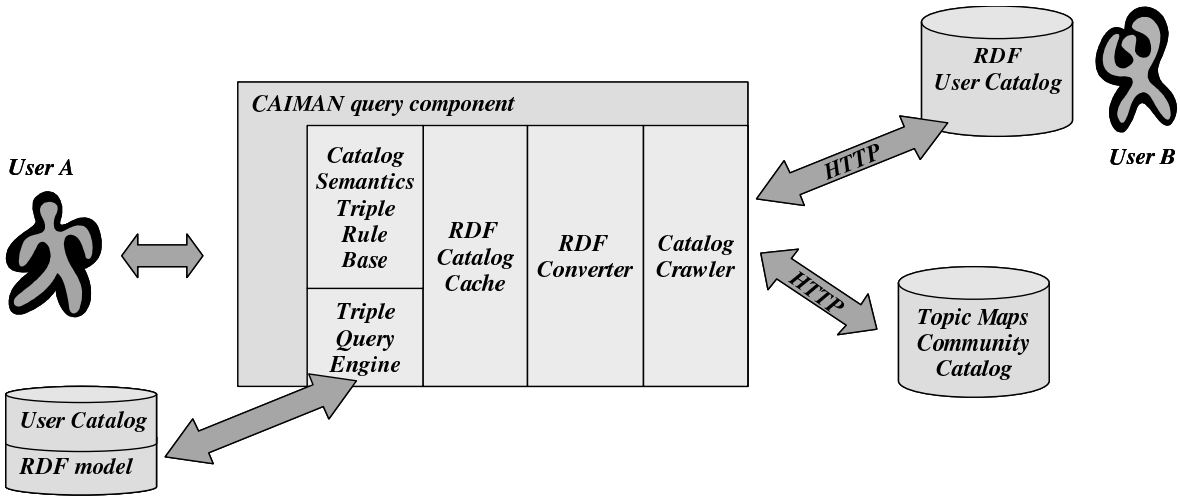


FIGURE 7.5: Conceptual overview of joint querying of different catalogs

It can be seen in Figure 7.5, that the query component can query user and community catalogs, which are represented with different data models. User A is the center of our considerations, the catalogs of user B and the community catalog are mediated with user A's user catalog. The CAIMAN query component runs locally on user A's workstation. The catalogs of user B and the community catalog are accessible via the Internet, using HTTP. We chose HTTP, because it is a widespread protocol, but any other application level transport protocol would be suitable as well.

[10] See http://www.cia.gov/cia/publications/factbook/

The components that are responsible for query mapping and processing are the TRIPLE query engine with its associated rule base. The TRIPLE engine processes queries on data that is cached in the *RDF Catalog Cache*.

The *RDF Catalog Cache* is required for two reasons: 1) we cannot assume that all mediated catalogs are willing to process all incoming queries from all community members, and 2) the TRIPLE query engine does not have the capability to query distributed data sources yet. Thus, catalog structure data, but no documents, are cached locally after they have been retrieved by the *Catalog Crawler*.

The *Catalog Crawler* connects to the catalog data sources, once the user has added a new catalog for mediation. We assume that all catalog structure data can be downloaded in bulk via Internet, as for example for the Open Directory catalog[11].

The *RDF converter* converts the downloaded catalog data from different data models into RDF. We show in detail how the conversion of Topic Maps to RDF can be performed in Section 7.5.

7.4 Related work

In [Borghoff and Schlichter 1996; Borghoff et al. 1996], a general framework for integration of heterogeneous data sources is presented. The architecture consists of wrappers and mediators to integrate the different heterogeneous information sources. The main difference to other integration frameworks such as TSIMMIS [Garcia-Molina et al. 1995] is that a constraint language is used for representation of queries and answers to queries in the framework. Query representation in a constraint language allows for effective decomposition of queries into sub-queries, which can then be directed to the right data sources. The framework presented in [Borghoff and Schlichter 1996; Borghoff et al. 1996] focuses on the decomposition of joint queries to heterogeneous data sources and reassembly of the query results. We focus on the translation of the semantics of the different data sources to RDF and subsequent joint querying.

In [Bowers and Delcambre 2000], a general approach to integration of heterogeneous model-based information has been presented. It is shown that in principle all model based information can be represented by an RDF based meta-model. It is shown that this also includes Topic Maps. However, the authors do not go into details about this specific mapping.

In [Moore 2001], two general approaches for the integration of Topic Maps and RDF have been proposed. The first approach shows how Topic Maps can be modelled with RDF vocabulary and vice versa. The second approach shows how a semantic mapping between the two standards can be performed. An inherent disadvantage of semantic mappings is that the mapping is lossy and thus the mapping cannot be bijective.

Pure syntax transformations have been proposed[12], but this approach disregards the need for a processing model to generate the Topic Map graph from the serialized syntax. Pure syntax transformations may lead to inconsistent data models in this case.

7.5 Joint querying of Topic Maps and RDF catalogs

As a proof of concept for our general approach for querying heterogeneous document catalogs, we demonstrate our data model conversion approach for the conversion of Topic Maps to RDF. The only sub-component within the CAIMAN query component that is specific for Topic Maps in this case, is the *RDF Converter*. The other components can be reused for other source data models.

[11] See http://rdf.dmoz.org/

[12] See http://lists.w3.org/Archives/Public/www-rdf-interest/2001Mar/0062.html

Our integration approach is to model a graph representation of a Topic Map with the means that an RDF graph gives us. In this approach, all information from the source model is preserved and just represented in another format. Thus, this transformation can also be seen as a syntax transformation on the level of a graph syntax. We picked the modelling approach because it is loss-free. We describe the conversion with respect to the layers of the layered model in [Melnik and Decker 2000].

7.5.1 Syntax layer

Our query approach is targeted for RDF on a data model level, i.e. the object layer in the layered interoperability model, and is thus independent of different syntaxes. Consequently, the conversion of data models has to be performed on the object layer as well and the different syntaxes, in which the different catalog data are represented, are irrelevant for the conversion. The underlying assumption is that the catalog crawler component is able to deserialize the crawled catalog data in a correct way. The correct way to deserialize an XML Topic Map has been described in [Biezunski and Newcomb 2001] and implemented in [Ahmed 2001].

7.5.2 Object layer

The representation of Topic Maps as RDF is a graph transformation between the data model graphs on the object layer of both models.

A graph model for RDF is defined in [Lassila and Swick 1999] RDF defines a graph model for RDF. The Topic Maps standard does not enforce a certain object layer representation for a Topic Map. Instead, a processing model has been proposed, which describes how to deserialize an abbreviated Topic Map syntax into a consistent graph-based data structure [Biezunski and Newcomb 2001].

Our goal is to map the Topic Map graph representation onto an RDF graph representation without any loss of information. We do this by mapping each element of the Topic Map graph described in [Biezunski and Newcomb 2001] to a corresponding construct in RDF. Figure 7.6 shows the RDF schema definition, which defines the RDF vocabulary that is necessary for our mapping (see [**?**] for more details on RDF schema).

```
<rdf:RDF xmlns:rdf="http://www.w3c.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3c.org/2000/01/rdf-schema#"
  xmlns:tms="http://www11.in.tum.de/rdftmmapping/tm-schema#"
  xmlns="http://www11.in.tum.de/rdftmmapping/tm-schema#">
  <rdfs:Class ID="t"/>
  <rdfs:Class ID="a"/>
  <rdfs:Class ID="s"/>
  <rdf:Property ID="associationMember"/>
  <rdf:Property ID="associationScope"/>
  <rdf:Property ID="associationTemplate"/>
  <rdf:Property ID="scopeComponent"/>
  <rdf:Property ID="roleLabel"/>
  <rdf:Property ID="scr"/>
  <rdf:Property ID="sir"/>
</rdf:RDF>
```

FIGURE 7.6: The RDF schema for an RDF-based Topic Map

Topic Maps graph model. As a prerequisite for the mapping, we require that the Topic Map graph is consistent, i.e. there are no redundant elements in the Topic Map graph [Biezunski and Newcomb 2001]. A Topic Map graph $tm = (N, A, S)$ consists of a set of nodes N. Every node in N is assigned one of the three types a (association node), t (topic node) and s (scope node). A is a set of edges , which have the different types *associationMember*, *associationScope*, *associationTemplate* and *scopeComponent*. S is a set of resources, which indicate or constitute a subject. The *associationMember* edge has a t-node (i.e., a node with type t) attached as a role label. Each node has at most one subject constituting resource and any number of subject indicating resources attached to it. The connection with these resources is not part of the graph [Biezunski and Newcomb 2001], but taken care of in the implementation domain. However, for a mapping without loss of information, we need to consider those resources as well. Figure 7.7 shows an example for an association represented by a Topic Map graph.

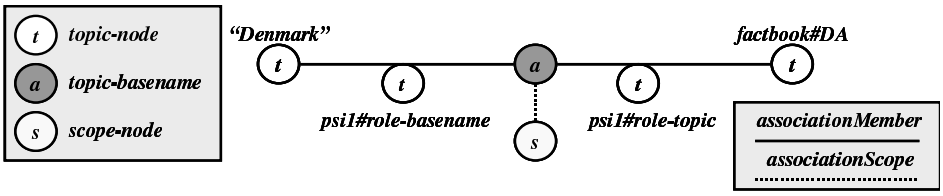


FIGURE 7.7: Example association represented in a Topic Map graph.

Graph mapping overview. An RDF Model graph $r = (R, L, ST)$ consists of a set of resources R, a set of literals L and a set of statements ST. Our mapping m maps the set of all consistent Topic Maps TM to the set of all RDF Models R. The set of nodes N is mapped to the set of resources R in RDF. The set of edges A is mapped to the set of statements ST in RDF. The set of subject indicating/constituting resources S is also mapped to the set of resources R in RDF. We map the Topic Map graph to an RDF graph by first mapping the graph nodes and then mapping the edges.

Graph node mapping. Each node in the Topic Map graph is mapped to a resource in the RDF model. The ID of the RDF resource is the ID of one of the subject identity points of the Topic Map node. If there is no subject identity point for the node, an ID is generated. For the rest of the subject identity points, statements are generated, which connect the subject identity points to its node in the RDF graph. An RDF statement is generated, which identifies the type of node that has been mapped. The Topic Map graph model knows three different kinds of nodes. We make use of the name space capability of RDF to define the three types of nodes available in Topic Maps. The node types are defined as shown in Figure 7.6. An exemplary mapping of a Topic Map node to an RDF graph is shown in Figure 7.8.

After all the nodes have been mapped, we map the edges in the Topic Map graph to statements in the RDF graph. For each edge between two nodes n_1 and n_2 we generate an RDF statement. The property of the statement corresponds to the edge type in the Topic Map graph. The corresponding properties are defined in the schema in Figure 7.6. Although edges in the Topic Map graph are not explicitly directed, they have an implicit direction given by the node types at each edge end. Thus, in that respect, the RDF graph is not more constrained than the Topic Map graph. If the mapped edge is an *associationMember* edge, it has a role label in the Topic Map graph. To represent the role label in the RDF graph, we reify the RDF statement signifying this edge and bind the role label node to this

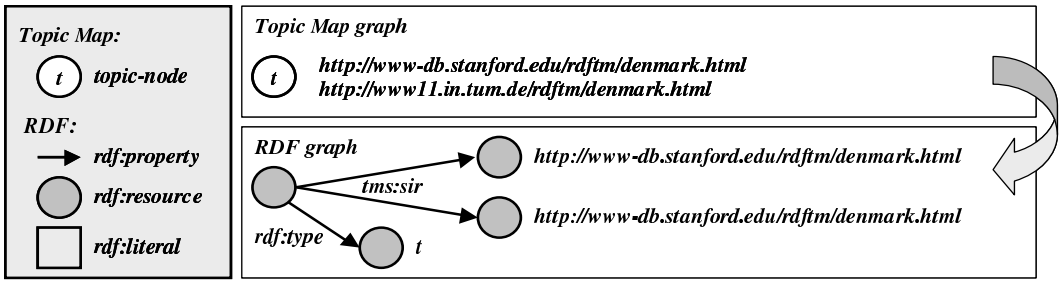


FIGURE 7.8: Exemplary mapping of a Topic Map node to an RDF graph

statement with the *roleLabel* property defined in Figure 7.6. The mapping of an *associationMember* edge between two nodes is shown in Figure 7.9.

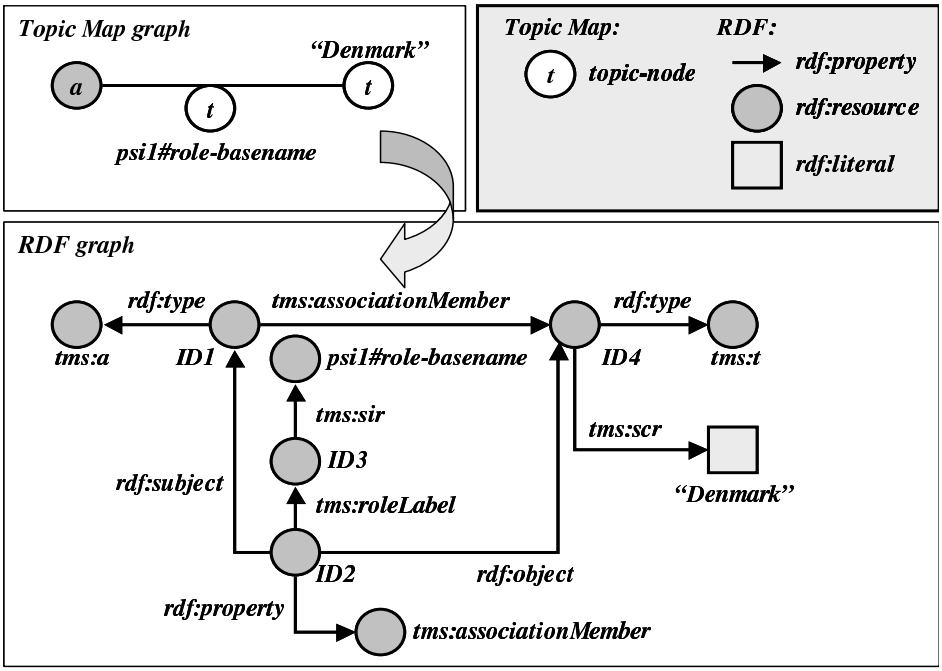


FIGURE 7.9: Exemplary mapping of a Topic Map associationMember edge to an RDF graph

7.5.3 Semantic layer

On the one hand, all higher-level semantics of the original Topic Map source data are conserved in our mapping of the graph model. For example, an *associationMember* edge in the original source data remains an *associationMember* edge in the converted data, it is just modelled with RDF. Thus, the semantic layer is not influenced by our data model conversion.

7.6 Query example for RDF and Topic Maps

Having presented the general query approach as well as the concrete conversion of Topic Maps to RDF, we are now going to present a real world application example. Our example shows how two

catalogs, one represented in RDF and one in Topic Maps, can be queried using the approach we propose. The first catalog is the Open Directory Catalog, a large web catalog the structure of which has been formalized as RDF[13]. The second catalog is the CIA World Factbook, a collection of resources about countries of the world[14], which is available as a Topic Map catalog[15].

The example consists of three parts. The first part describes the data sources in detail with respect to the graph model of the data. We pick a small part of the catalog definition of each of the two catalogs to illustrate the graph representations. The second part shows how the Topic Map example graph is converted to an RDF graph. The third part explains for each of the two catalog sources how they can be queried using the querying approach depicted in Figure 7.5. We choose a graph representation of the data sources. Queries thus represent path expressions over the data graphs[16].

7.6.1 Graph models of the data sources

For both catalogs, we chose the category about the country *Denmark* as the example category by which we explain the catalog characteristics. The characteristics of interest include the path to the catalog root as well as how *sub-category* and *related-category* relations are represented in the respective catalogs. In the Open Directory (DMOZ) catalog, the path from the catalog root to the category *Denmark* is *Root/Regional/Europe/Denmark*. The DMOZ catalog structure consists of categories, *sub-category* relations, which are represented by the *narrow* property in RDF and *related* relations, which are represented by the *related* property in RDF. The documents in each category are referenced by *link* properties. An RDF/XML serialization of the piece of the catalog structure that describes the *Denmark* category can be seen in Figure D.2 in Appendix D. The category contents, i.e. the documents in categories, are separated from the structure in the DMOZ data sources and can be seen in Figure D.1 in Appendix D. The RDF/XML data is downloaded by the *catalog crawler* component and cached locally by the *RDF catalog cache component* of the CAIMAN query component (see Figure 7.5). Here, we only show the cached graph model, with which the query process can later be explained. The graph resulting from processing the RDF/XML file from Figures D.1 and D.2 in Appendix D can be seen in Figure 7.10.

The unquoted strings in Figure 7.10 represent the IDs of RDF resources, the strings in quotation marks represent literals. The prefix *r:* represents the RDF standard XML-namespace[17], the prefix *d:* represents the Dublin Core XML-namespace[18]. As can be seen in Figure 7.10, catalog categories are represented by resources of type *Topic*. The categories carry some extra metadata in addition to their ID, such as a plain text *Title*. The sub-category relation is represented by the *narrow* property, related categories are referenced by a *related* property. Documents contained in a category are referenced by a *link* property.

In the CIA World Factbook catalog, there is not a clear tree structure of the categories. However, there are topic classes, of which the rest of the categories are instances. We interpret the classes as the catalog perspectives and *class-instance* relationships as *sub-category* relations. Related categories are referenced in many ways. For example, a country can be related to another country through a common natural resource that both countries have. However, if such complex relationships would

[13] See http://www.dmoz.org/rdf

[14] See http://www.cia.gov/cia/publications/factbook/

[15] See http://www.ontopia.net/omnigator/models/topicmap_complete.jsp?tm=factbook.hytm

[16] Since semi-structured data can always be represented by a graph, a query over semi-structured data can be represented by a path expression.

[17] See http://www.w3.org/1999/02/22-rdf-syntax-ns#

[18] See http://purl.org/dc/elements/1.0/

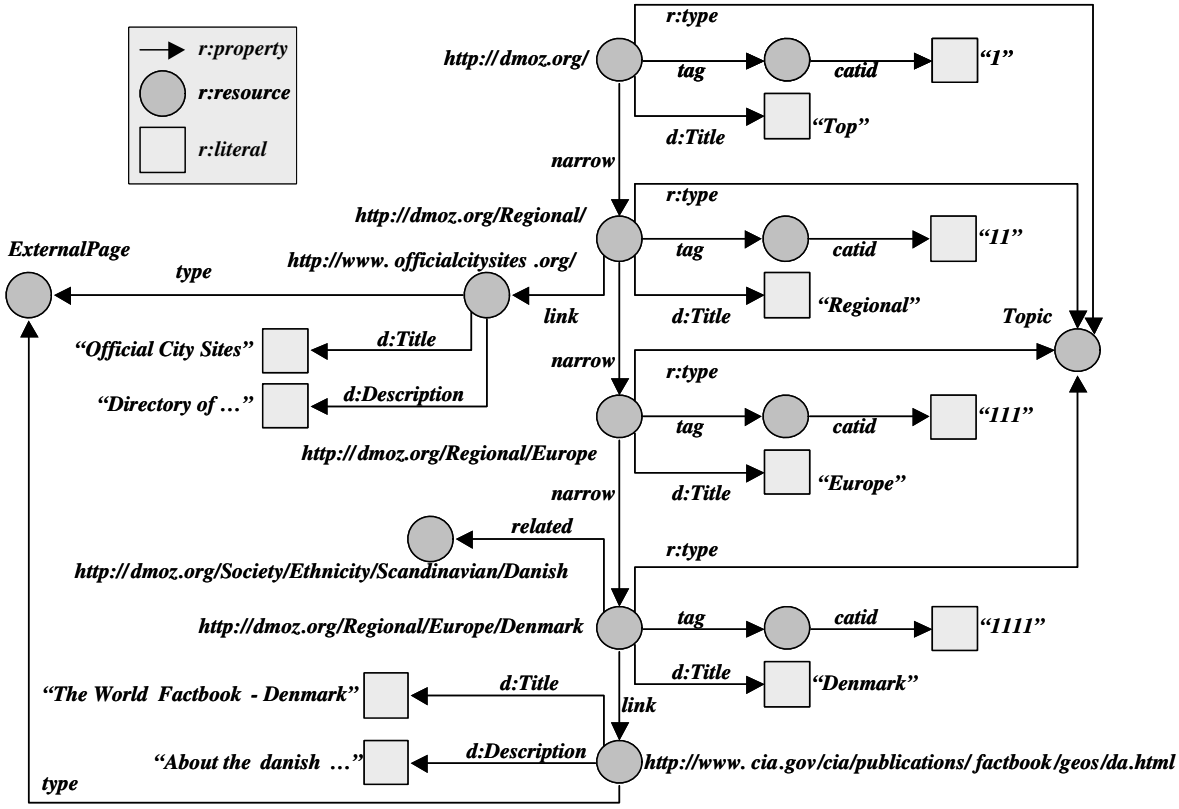


FIGURE 7.10: RDF graph part from the Open Directory Catalog

be interpreted as *related* relations among categories, eventually every category would be related to all others. Thus, for this catalog, we neglect *related* relations. Assuming the *country* perspective, the path from the catalog root to the category *Denmark* is *Root/Country/Denmark* in this case. A XTM/XML serialization of the piece of the catalog structure that describes the *Denmark* category can be seen in Figure D.2 in Appendix D. The XTM/XML data is downloaded by the *catalog crawler* component, processed and fed into the *RDF converter* component of the CAIMAN query component (see Figure 7.5). Processing of the XTM data according to the processing model presented in [Biezunski and Newcomb 2001] results in the Topic Map graph shown in Figure 7.11.

The unquoted strings in Figure 7.11 represent the *Subject Indicating Resources* of topics, strings in quotation marks represent *Subject Constituting Resources* of topics. The prefix *psi1#* represents the Topic Maps *Published Subject Indicator* resource, in which some general notions like *class* or *instance* as well as association templates are defined[19]. The prefix *factbook#* represents the Factbook XTM document base URL. For clarity reasons, we left out the complete graph representation of association types and introduced typed, differently colored a-nodes instead. As can be seen in Figure 7.11, the catalog categories for the subjects Denmark (*factbook#DA*) and Country (*factbook#country*) are represented by t-nodes. Topic Map role labels are represented by t-nodes, which in Figure 7.11 are attached to the *associationMember* edges. The *sub-category* relation is represented by an a-node of type class-instance. The scopes, within which associations hold, are represented by s-nodes. The graph in Figure 7.11 in Topic Map terms represents the facts that the topic *factbook#DA* is an instance of the

[19] See http://www.topicmaps.org/xtm/1.0/psi1.xtm

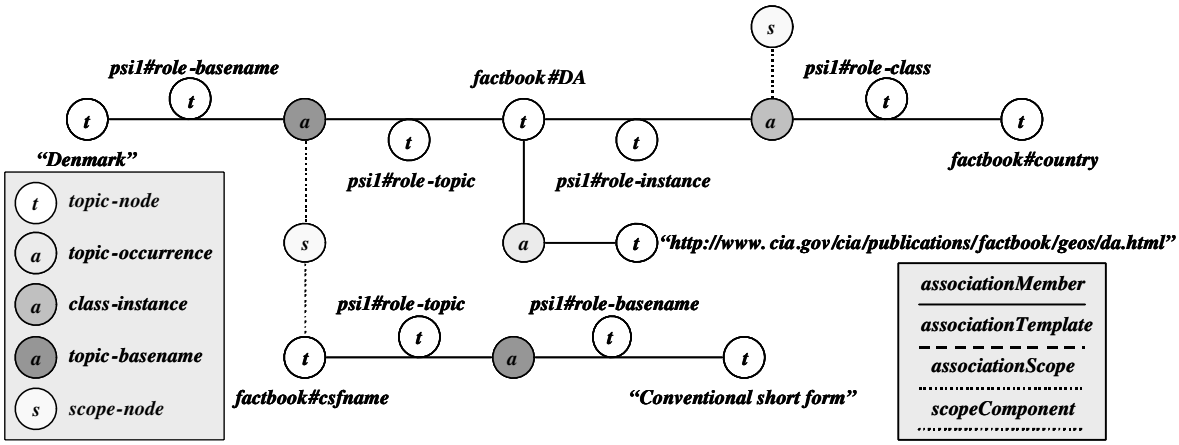


FIGURE 7.11: Topic Map graph part from the CIA World Factbook

topic *factbook#country*. Further, *factbook#DA* has *"Denmark"* as one of its base names, which is a *"Conventional short form"* of the Topic name. The only document, that is assigned to *factbook#DA* is *"http://www.cia.gov/cia/publications/factbook/geos/da.html"*.

7.6.2 Mapping the Topic Map source graph to RDF

After the Topic Map data has been read in and processed by the *catalog crawler* component, the resulting graph model (see Figure 7.11 is fed into the *RDF converter* component (see Figure 7.5). The *RDF converter* component models the Topic Map with an RDF graph without any loss of information. The transformation is performed as shown in Figure 7.8 for Topic Map nodes and as shown in Figure 7.9 for Topic Map associations. The transformation result can be seen in Figure 7.12.

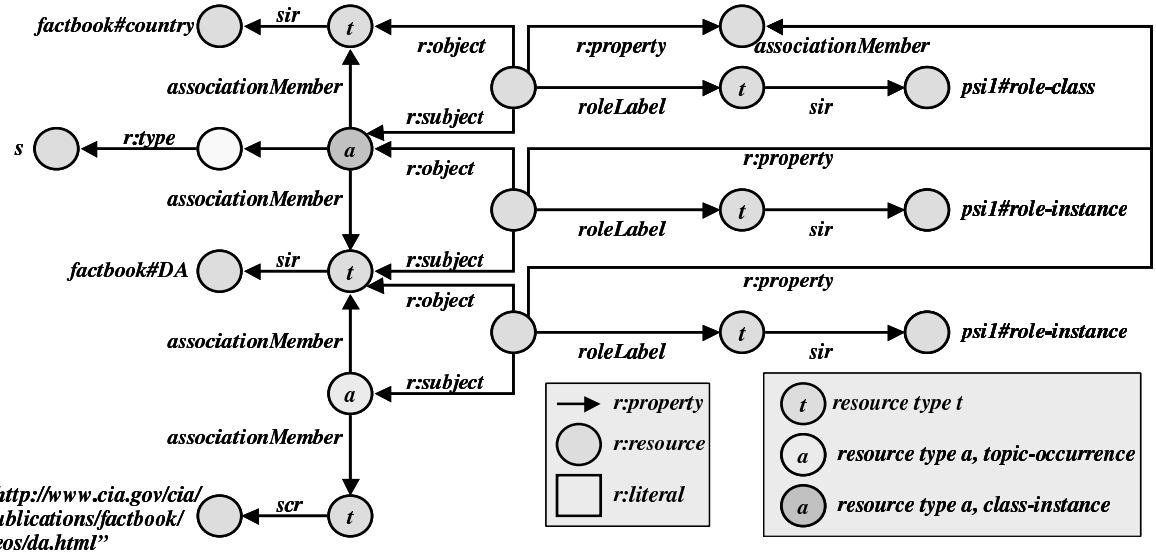


FIGURE 7.12: The generated RDF Topic Map graph.

In Figure 7.12, the plain strings represent RDF *resource IDs* and the strings in "" represent literals.

The prefix *r:* represents the RDF standard XML-namespace[20]. In Figure 7.12, some features of the RDF graph have been abbreviated for the sake of clarity. The *r:type* property for a-nodes and t-nodes is replaced by nodes that are labelled *t* or *a* respectively. Moreover, not all information from Figure 7.11 has been transformed and included in Figure 7.12. We also cut the assignment of a *base name* to the *Denmark* subject along with the scope of this assignment. The *sir* and *scr* properties represent *subject indicating resources* and *subject constituting resources* respectively. Nodes that have no ID in the graph, are assigned a generated ID. The edge labels of the Topic Map graph are modelled in the following way: the RDF statement that represents the *associationMember* edge of the Topic Map source, is reified and its reification has a *roleLabel* property, which points to the respective role label. After the transformation, the generated RDF graph is cached in the *RDF catalog cache* component (see Figure 7.5).

7.6.3 Joint querying of the DMOZ and the Factbook catalog

After the catalog data has been converted to RDF and has been cached in the *RDF cache*, the data can be queried using the *TRIPLE query engine* and the rule base, which holds the query mapping rules.

Query mapping rules. The user has to supply some information that is required for the generation of query mapping rules. First, the user has to identify the type of relation that resembles the *sub-category* relation between two categories in the original catalog source data. Second, the user has to identify, with which type of relation, documents are assigned to a category. We have seen in the example graphs above that the *narrow* relation carries the *sub-category* semantics in the DMOZ catalog and documents are assigned to a category with the *link* relation. In the CIA Factbook, the *class-instance* relation represents the *sub-category* relation and documents are assigned to a category with the *topic-occurrence* relation.

Based on the mapping information for relations and the matching information for categories, query mapping rules can be generated. TRIPLE allows the specification of such query transformation rules. We have created some simple examples for query mapping rules with TRIPLE for the query example here, which are presented in Appendix C. More details on transformation rules in TRIPLE can be found in [Sintek and Decker 2001] and details on query mapping in a database context can be found in [Halevy 2001].

Query processing. After the catalog matchings have been calculated and the respective mapping rules have been generated, we can query the catalogs in an integrated fashion. We assume that the user has a category labelled *Scandinavia* in her catalog and invokes the related information retrieval service on this category. The query for all documents related to the *Scandinavia* category posed by the related retrieval service looks like this in TRIPLE syntax (see [Sintek and Decker 2001]):

```
FORALL D <- Scandinavia[contains->D]@USER
```

where `D` is a variable for the respective documents and `contains` is the relation that relates a document to a category in the user catalog. In our example, the matching categories of the *Scandinavia* category in both other catalogs are labelled *Denmark*. Together with the user-supplied information about relation mappings, this is encoded in TRIPLE mapping rules for the respective source. Subsequently, the same query can be posed over the DMOZ and Factbook catalog simply by adding the respective *source expression* to the query:

[20] See http://www.w3.org/1999/02/22-rdf-syntax-ns#

```
FORALL D <- Scandinavia[contains->D] OR
            Scandinavia[contains->D]@DMOZ OR
            Scandinavia[contains->D]@CIAFACT
```

The source expressions `@DMOZ` and `@CIAFACT` reference the respective TRIPLE model for a catalog, in which both the catalog data and the mapping rules are encoded.

For the DMOZ catalog, for example, the `contains` relation will be mapped to a `link` relation and the `Scandinavia` category will be mapped to the `Denmark` category (see Figure 7.10). More details of the mapping rules for this example can be found in Appendix C.

7.7 Summary

In this chapter we presented an approach that allows heterogeneous catalogs to be queried with the same RDF query infrastructure. Our main contributions in this chapter have been the following:

1. We proposed RDF as the lingua franca for catalog representation to allow for joint querying of catalogs in our scenario, using the TRIPLE RDF query engine [Sintek and Decker 2001]. We have shown how the different data models of document catalogs can be modelled with RDF, following a layered approach to data interoperability from [Melnik and Decker 2000].

2. As a proof of concept for the modelling solution, we showed how document catalog data represented as Topic Maps can be modelled with RDF by a loss-free mapping. RDF and Topic Maps are the two most popular standards for knowledge representation on the Web today.

3. We introduced a coarse software architecture for the query component and its sub-components in CAIMAN.

As a supplement to our querying approach, we introduced an exemplary concept for performing the necessary query mapping in our querying approach. The query mapping concept merely serves as a proof of concept for our querying approach and is not defined in great detail. The query mapping concept is also based on the TRIPLE [Sintek and Decker 2002] RDF query engine [Sintek and Decker 2001]. We showed how TRIPLE rules can be used for a declarative specification of some of the necessary query mappings.

Finally, we presented an example, how the Topic Map based CIA World Factbook can be modelled with RDF. We showed how the converted World Factbook can be jointly queried with the RDF based Open Directory web catalog, assuming that the TRIPLE system takes care of all necessary query mapping.

Our querying approach has the following advantages:

- Catalogs based on different data models can be queried with the same infrastructure.
- In a scenario with n different data models, only $n - 1$ converters are required.
- The data model conversions are simple and straightforward graph transformations.
- If a query engine like TRIPLE is used, the mapping rules can be defined declaratively for all data models, which is a great simplification over a procedural definition. Moreover, with TRIPLE, the data model semantics can be defined independently of specific catalogs. Thus, the only information that a user has to supply if a new catalog is added for mediation, are the labels (names) of data model relations identifying sub-categories and contained documents.

Chapter 8

CAIMAN Knowledge Exchange Service Performance Evaluation

8.1 Introduction

In Chapter 6, we have introduced a novel approach for matching document catalogs as the basis for category granular catalog mediation in the CAIMAN framework. In this chapter, we are going to present experimental evidence as a proof of concept for our document catalog matching approach. Our goals for this chapter are threefold:

- Provide experimental evidence that the CAIMAN catalog matching approach is a suitable basis for the knowledge exchange services presented in Chapter 4 with respect to the application requirements.
- Show that a category granular mediation approach based on the CAIMAN catalog matching approach can compete with a state-of-the-art document granular approach, in terms of the knowledge exchange service quality.
- Show that the CAIMAN catalog matching approach performs well enough to serve as the basis for the semi-automatic knowledge exchange services.

We have conducted experiments that compare category granular mediation based on the CAIMAN matching approach to state-of-the-art document granular techniques. Our performance measure resembles the quality of service of the *related retrieval* service. The experimental results of the *related information retrieval* service can be seen as a proof of concept for the *information publication* service as well, as the two are symmetric. The matching performance is evaluated automatically on two different document corpora, which contain a user catalog and a community catalog each.

The experimental setup is described in detail in Section 8.2.3. We briefly explain the implementation of the CAIMAN mediation prototype in Section 8.3. In Section 8.4, we show the results of our performance evaluation experiments.

8.2 Experimental setup

We evaluate the quality of the CAIMAN matching approach in terms of the quality of service that can be delivered by the knowledge exchange services (see Chapter 4). There are two principle ways to

evaluate the quality of the CAIMAN matching approach: through automatic evaluation or through a user study. As the knowledge exchange services have been designed to assist users in certain tasks, it is ultimately the user, who can judge whether the services are helpful or not. However, for a user study, a full functioning prototype for all evaluated services is required. The evaluation of a prototype is subject to many influences, which are not directly related to the quality of service provided by the prototype. Among these external influences are the prototype interaction design, integration into individual work processes, and runtime performance, which can all influence the users' perceived quality of service. Thus, a stand-alone user study would make it difficult to single out the ultimate cause for poor performance. In contrast, an automatic evaluation can supply important information about the necessary conditions for good performance. Therefore, we consider an automatic evaluation without user involvement a necessary basis for further evaluation steps. As the implementation of a full working prototype of the CAIMAN system is not the focus of this work, we perform an automatic evaluation. For our evaluation, we focus on the *related retrieval* service. The results can be transferred to the *Publication* service, as the two services are symmetric. The *category discovery* browsing service cannot be evaluated automatically, only the user can say whether two categories with different names are related or not.

8.2.1 Automatic evaluation

Automatic evaluation of the CAIMAN service performance requires some advance information what correctness means for a certain service. To be able to evaluate the correctness of the *related retrieval* service (see Chapter 4), we need to know in advance for each of the documents to be retrieved, to which category in the destination catalog they belong. For our performance evaluation here, we achieve this by splitting an original catalog into two catalogs with the same structure, but different documents in each of the respective categories. One of the resulting catalogs then takes on the role of the user catalog, the other catalog takes on the role of a community catalog. For each document from the community catalog, it is known in advance to which category in the user catalogs it should belong. What we evaluate is the quality of the *related retrieval* service, retrieving documents from the community catalog into the user catalog. An overview of our evaluation approach can be seen in Figure 8.1.

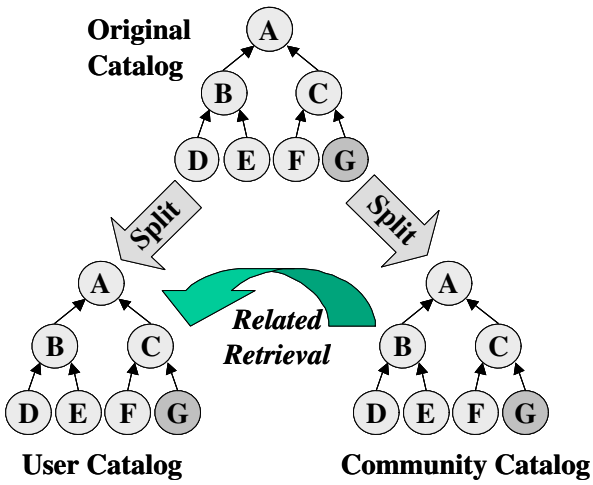


FIGURE 8.1: Generation of the user and community catalog for automatic evaluation of the *related information retrieval* service

To evaluate the quality of the *related retrieval* service, we compare standard text categorization approaches to the CAIMAN approach in different variations. For each of the documents from the community catalog, we estimate to which of the categories in the user catalog it should belong. The estimate is correct, if the source category has the same name label as the destination category. Our performance measure is based on the fraction of correct estimates for documents among all estimates.

The above mentioned catalog splits divide the original catalog into the user catalog, which holds 80% of all documents and the community catalog, which holds 20% of all documents. These percentages hold for each single category, such that the relative category sizes of the original catalog are conserved. The $80 : 20$ size relation is due to the fact that we perform *5-fold cross validation* (see Section 6.3) in order to increase the statistical significance of our results. Cross-validation is performed for all catalogs except the *Reuters 21578* collection, since this catalog is evaluated with a given catalog split, which is popular in the IR literature (see Section 8.2.3.1). We also show in our experimental results that the $80 : 20$ catalog split relation incurs no unfair performance advantage over a $50 : 50$ split relation.

8.2.2 Performance measure

We chose the F_1 measure (F_{β_F}-measure with $\beta_F = 1$, see Chapter 6.3.5) as our performance measure for the following reasons:

- The F_{β_F} measure has been acknowledged as a measure that takes into account most aspects of categorization performance, in contrast to unbalanced measures, such as accuracy.
- The F_1-measure assigns equal importance to π (precision) and ρ (recall). This balance resembles a tight quality requirement for an average user.

Furthermore, we evaluate both the macro-averaged F_1^M-measure and the micro-averaged F_1^μ-measure. The F_1^μ-measure emphasizes the approach performance on categories in the user catalog with a large number of training documents. In contrast, the F_1^M-measure emphasizes especially the performance on categories with few training documents. These two different averaging techniques allow us to give a very detailed performance account for catalogs with varying category sizes, i.e. numbers of documents per category.

We have stated as our goal for this chapter, that we aim to show that the CAIMAN catalog matching approach performs well enough to be the basis for semi-automatic knowledge exchange services presented in Chapter 4. We consider the *related retrieval* service to perform well enough, if the user has an advantage from using it over not using it. Simply put, we consider a service to perform well enough if the F_1 measure is over 50%. The intuition behind this is, that the user has to invest less effort correcting the errors of the *related retrieval* service than it would take to retrieve the correct documents herself. Actually, this is a very cautious assumption, as retrieving related documents most likely takes a lot more effort than rejecting proposed documents. Thus, with an F_1 measure of over 50%, we can be sure that the service brings an advantage for the user. Moreover, if the user prefers a higher precision or a higher recall, one can be traded off against the other in order to suit the user's requirements.

8.2.3 Experiments

We evaluated our performance measure in various experiments in order to give a very detailed impression of the performance under varying circumstances. As the CAIMAN matching performance

depends on more than 20 parameters, a few parameters have been picked to show their influence on the service performance. For the rest of the parameters, we have adopted optimal values published in related works, or run experiments to determine near-optimal values, which are briefly presented in [Etzold 2003]. Our experiments are grouped by the catalogs, with which they have been performed. For each catalog, we performed the following experiments:

- F_1 performance dependent on the reduced term set size $\mathcal{T}'$ (see Chapter 6). $\mathcal{T}'$ has a great influence on the F_1 performance of the *related retrieval* service and there is a tradeoff between $\mathcal{T}'$ and runtime complexity. In our experiments, we reduce the original term set size $\mathcal{T}$ to $\mathcal{T}'$ by picking the terms, which provide the maximum information gain (see Section 6.3). This experiment serves to show the overall relative F_1 performance for a given catalog, as well as to determine $\mathcal{T}'$ values for the rest of the experiments.

- F_1 performance dependent on the maximum classifier training set size κ. κ also has a very strong influence on the F_1 performance and trades off with runtime complexity. This experiment serves to show the relative scalability of the various approaches with user catalog category sizes.

- F_1 performance dependent on the relative category vector weight for context aware categorization, α. This experiment serves to show, that our proposed context aware categorization technique can further improve the service performance.

8.2.3.1 Evaluated catalogs

In order to get a detailed picture of the relative performance of the CAIMAN approach, we chose two different test catalogs, which differ in the following key characteristics:

- Average number of documents per category, as well as its variance (also called **skewedness**).

- Total number of categories as well as the catalog tree height.

- Average amount of text data in kBytes available per category.

We evaluate the performance of the CAIMAN approach on catalogs, which contain documents in English only. For each catalog, we pick one categorization criterion and demand that each unique document belongs to exactly one category in a catalog. The two catalogs we picked are:

- The **Reuters 21578 collection** of news feeds, which is a popular IR benchmark[1]. The collection consists of 21578 short plain text documents categorized into a flat list of categories. We used a pre-determined split of the original catalog into User and Community catalog: the modApté split, modified as described in [Joachims 1998]. In this split, each document belongs to exactly the first category in the list of assigned categories. Moreover, empty categories in the User as well as the Community catalog are pruned, along with their sibling categories in the opposite catalog. Both the user catalog and the community catalog are very skewed, with categories holding close to 3000 documents as well as categories with as little as 2 documents (see [Yang and Liu 1999]). The user catalog holds a large average number of documents per category. The single documents are very small, however. The exact figures can be seen in Table 8.1.

[1] See http://www.research.att.com/ lewis/reuters21578.html

- The **Researchindex**[2] collection of computer science research papers. We downloaded 20 documents from each of the categories of the Researchindex catalog. Since the documents from the Researchindex catalog are in Postscript or PDF[3], they had to be converted to plain text for our purposes. This catalog is the same that has already been used for evaluations in [Groh 2001]. We use the cross validation split described in Section 8.2.1. The total number of documents, the number of documents per category and the skewedness are much smaller than in the Reuters collection. The Researchindex catalog has twice as many categories as the Reuters catalog and a tree structure with a maximum depth of two. The Researchindex catalog holds the maximum amount of text data per category, up to 10 times as much as the Reuters catalog. The exact figures can be seen in Table 8.1.

Catalog	D	C	D/C	S [MB]	S/D [kB]	S/C [kB]	Skew	TL
Reuters Total	10763	62	174	8.5	0.79	136.7	High	1
Reuters User	7752	62	125	6.2	0.8	100.1	High	1
Reuters Community	3011	62	49	2.3	0.75	36.6	High	1
Researchindex	2417	123	20	156	68	1270	Low	2

D = Total Number of documents, C = Number of Categories, S = Total Catalog size in MBytes, Skew = Catalog Skewedness level, TL = Maximum number of tree levels

TABLE 8.1: Overview of the catalogs used for evaluation

8.2.3.2 Evaluated Approaches

In our experiments, we evaluate the performance of state of the art IR techniques in comparison with the CAIMAN approach in different variations. To test the plain IR techniques, the respective classifiers are trained on the user catalog. Thereafter, all documents from the community catalog are categorized into the user catalog and the result is checked for correctness. To test the CAIMAN approach (see Chapter 6), the first step is to train a classifier on the user catalog as well. Next, a tentative categorization of the documents from the community catalog into the user catalog is performed. The result of the tentative categorization is used to calculate an initial similarity value between categories in the user catalog and categories in the community catalog. This is followed by an optional phase of structural matching to refine the similarity values. Using filters, the final category match between categories in the user catalog and categories in the community catalog is determined. Finally, all documents from the community catalog are categorized into the user catalog according to the established category match and the result is checked for correctness.

In our experiments, we compare the following four approaches:

- **D**: Document granular approach. Each document from the community catalog is categorized independently into the user catalog. This resembles state-of-the-art text categorization techniques.

- **C1 (CAIMAN 1)**: Category granular approach without structural matching. An initial category match between the categories in the user catalog and the categories in the community catalog

[2] See http://www.researchindex.com/

[3] Postscript and PDF are registered trademarks of Adobe.

is established. All documents from the community catalog are categorized into the user catalog in a category granular fashion.

- **C2 (CAIMAN 2)**: Category granular approach with similarity inheritance structural matching. The outcome of the structural match is filtered with *Retrieval* or *Stable Marriage* filters. All documents from the community catalog are categorized into the user catalog in a category granular fashion.

- **C3 (CAIMAN 3)**: Category granular approach with similarity flooding structural matching. The outcome of the structural match is filtered with the *Retrieval* ($[0,1]-[0,n]$) filter in C3 and with the *Stable Marriage* ($[1,1]-[1,1]$) filter in **C3***. All documents from the community catalog are categorized into the user catalog in a category granular fashion.

For a fair comparison of the approaches, the same categorization technique is used in all four approaches within one experiment. In the experiments with the *Reuters 21578* catalog, approaches C2 and C3 are not tested, due to the lacking catalog graph structure of the catalog.

8.3 Implementation

We have implemented a prototype of the CAIMAN mediation component, in order to be able to perform mediation experiments with real life document catalogs. The prototype holds all catalog matching functionalities of the CAIMAN matching approach described in Section 6.4. However, no catalog querying functionalities, as described in Chapter 7, have been included in the current prototype. The prototype of the mediation component is supported by functionalities for performing experiments for quality of service evaluation of the CAIMAN knowledge exchange services. Overall, the prototype has been designed for batch evaluation experiments and cannot directly be used in a CAIMAN application (see Chapter 4). We give a brief overview of the prototype implementation here, a more detailed description can be found in Appendix B.

Using the implemented prototype, experiments can be defined in a configuration file and be run in batch mode. The prototype offers the following functionalities:

- **Catalog preparation functionalities**: crawling, plain text conversion from various formats, preparation of pre-defined catalog splits and cross-validation catalogs

- **Catalog indexing functionalities**: indexing of hierarchical and flat catalogs, stemming, stopword removal, term set size reduction, word-document-matrix conversions

- **Catalog classifier training and categorization functionalities**: Naive Bayes approach, Support Vector Machines, k-Nearest-Neighbor, Rocchio vector cosine measure

- **Catalog structural matching**: Category similarity estimate by document voting, structural matching with similarity inheritance algorithm, structural matching with similarity flooding algorithm, category match filters

- **Evaluation measures**: Precision, Recall, F-measure (all macro-averaged or micro-averaged), Accuracy

- **Experiments batch processing**: configuration driven batch processing of experiments, results table output, graph output

The mediation component prototype uses both own implementations of techniques presented in Sections 6.3 and 6.4 as well as external, freely available implementations of certain components. An overview of the implementation for our experiments in this chapter can be seen in Figure 8.2.

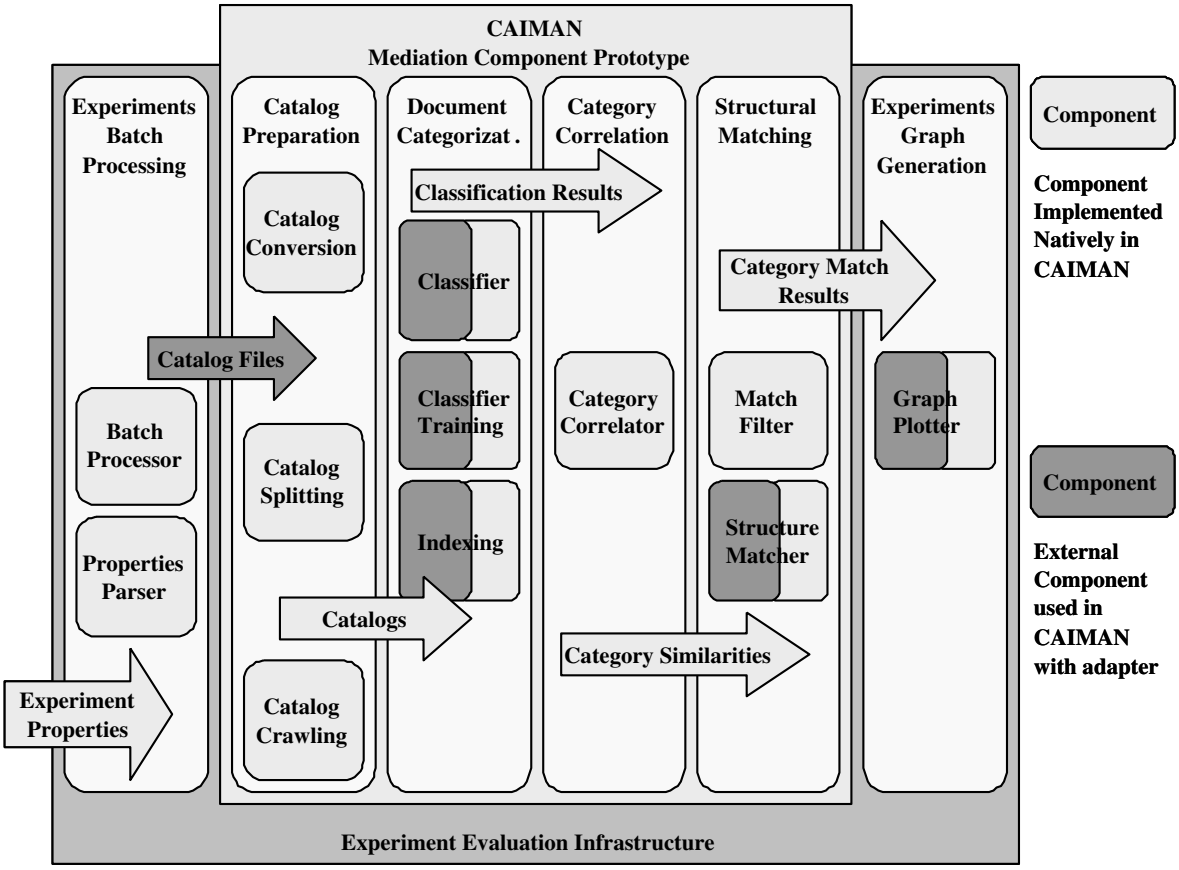


FIGURE 8.2: Overview of the data flow in the mediation component prototype implementation

Figure 8.2 shows the different functional components as quadratic blocks. Components, which have been implemented from scratch in the CAIMAN mediation prototype have a brighter shade than external components, which are used from within CAIMAN through an adapter. The different components are grouped into consecutive phases of the experiment process, which are represented as vertical bars in Figure 8.2. One run through a complete experiment can be seen as a pass through all phases from left to right in Figure 8.2. The arrows in Figure 8.2 represent data flow between the different phases. Within one phase, data flows from the incoming arrow towards the outgoing arrow, passing through the components in the respective phase on the way. Components, which are not between an incoming and an outgoing arrow, like the *Catalog Crawling* component, are independent of the experiments batch process.

The CAIMAN mediation prototype has been implemented in JAVA. However, the external libraries used allow the experiments to be run only under a Linux operating system. In detail, the following external components have been used for the realization of the CAIMAN mediation prototype:

- **Catalog preparation functionalities**: The online document catalogs have been crawled using

the WebSphinx toolkit[4], and a unix-shell-script, which relies on the *GNU wget* toolkit[5]. The conversion of Postscript and PDF[6] documents to plain text has been performed with the *GNU pstotext* toolkit[7].

- **Catalog indexing functionalities**: Most of these functionalities are provided by the *Rainbow* text categorization toolkit[8]. The Rainbow toolkit is used in CAIMAN by executing a native binary with the respective parameters. Input to and output of the Rainbow binaries are created and converted by several CAIMAN-internal adapters. The text categorization techniques implemented in Rainbow only allow categorization of documents into a flat list of categories. Therefore, CAIMAN converts catalog tree structures into flat category lists and back, as described in Section 6.4.

- **Catalog classifier training and categorization functionalities**: The Naive Bayes categorization technique is provided by the Rainbow toolkit. The Support Vector Machines technique has been implemented in an optimized fashion in the *LibSVM* toolkit[9]. Just like with the Rainbow toolkit, the LibSVM binaries are executed by CAIMAN-internal adapters. The k-nearest-neighbor as well as the Rocchio classification technique have been implemented natively in CAIMAN. However, the latter two techniques have been implemented for testing purposes only and no performance results are shown for these techniques in this work.

- **Catalog structural matching**: The initial category similarity estimate as well as the similarity inheritance structural matching have been implemented in CAIMAN. For the similarity flooding structural matching, we rely on the freely available JAVA implementation[10] from the author of the original similarity flooding publication [Melnik et al. 2002]. The match filters including the stable marriage filter have been implemented natively in CAIMAN.

- **Evaluation measures**: The calculation of performance measures has been implemented natively in CAIMAN for all available measures.

- **Experiments batch processing**: The batch processing of experiments and configuration has been implemented natively in CAIMAN. *Gnuplot*[11] is called from CAIMAN for output of the experiments results as graphs.

The CAIMAN prototype is the result of a combination of several student project implementations of specific prototype sub-components. For more details on the building blocks of the CAIMAN prototype, the interested reader is referred to the description in Appendix B as well as to the original descriptions in [Groh 2001], [Kolodizki 2002] and [Etzold 2003].

8.4 Performance evaluation

The experimental results presented in this section have been calculated with the CAIMAN prototype implementation described in the previous section. The graphs show the performance of the CAIMAN

[4] See http://www-2.cs.cmu.edu/ rcm/websphinx/
[5] See http://www.gnu.org/software/wget/wget.html
[6] Postscripts and PDF are registered trademarks of Adobe.
[7] See http://research.compaq.com/SRC/virtualpaper/pstotext.html
[8] See http://www-2.cs.cmu.edu/ mccallum/bow/
[9] See http://www.csie.ntu.edu.tw/ cjlin/libsvm/
[10] See http://www-db.stanford.edu/ melnik/mm/sfa/
[11] See http://www.gnuplot.info/

matching approach with regard to the *related retrieval* service, described in Section 8.2. Moreover, the graphs show the performance of a document granular mediation approach, using state of the art information retrieval techniques. We used the F_β performance measure in different experiments on the three catalogs described in Section 8.2.3.1. We show results for two different classification approaches: the Naive Bayes [Lewis 1998] approach and the Support Vector Machine [Joachims 1998] approach. We picked the Naive Bayes approach, because it is among the simplest text categorization approaches with good runtime complexity characteristics. However, a Naive Bayes classifier usually requires more training documents than an SVM classifier to perform equally well. SVM have been shown to outperform other classifiers in [Joachims 1998], [Yang and Liu 1999] and [Sebastiani 2002]. On the other hand, SVM have a higher runtime complexity than a Naive Bayes classifier. We consider the two approaches representatives of two extremes on the complexity-performance trade-off curve.

If not mentioned otherwise in the experiments, we use the following default parameters in our experiments:

- **Naive Bayes**: removal of stop-words, uniform category priors, multinomial event model (see Section 6.3)
- **Support Vector Machines**: C-SVC (see [Chang and Lin 2001]), linear kernel, $c_{SVM} = 1$, $\nu = 0.5$, all other parameters remain in the default setting described in [Chang and Lin 2001].
- **Similarity Inheritance**: $s_{min} = 0.02$, $\alpha_i = 0.3$.
- **Similarity Flooding**: $s_{min} = 0.02$
- **Match Filters**: $[0,1] - [0,n]$ or $[1,1] - [1,1]$ with stable marriage condition

The chosen parameter values for Naive Bayes and Support Vector Machines have been shown in [Etzold 2003] and [Goller et al. 2000] to either perform best or only incur a negligible disadvantage over parameters which would incur a major runtime complexity disadvantage. The parameters for the two structural matching approaches have been shown in [Kolodizki 2002] and [Melnik et al. 2002] to perform best for the respective approach.

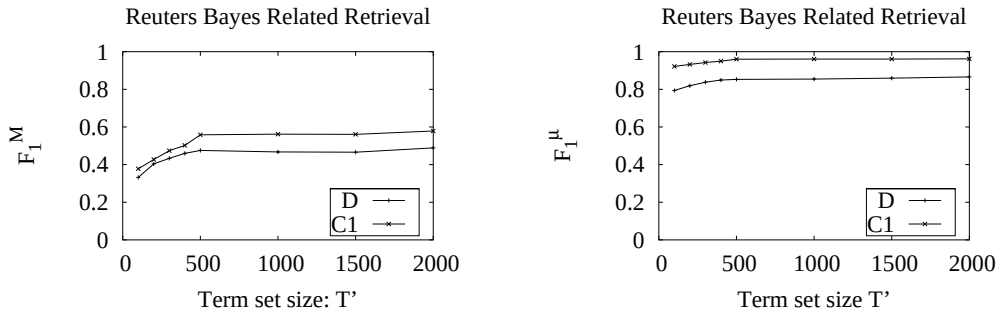
8.4.1 Reuters 21578 catalog

We use the popular *modApté* split as described in [Joachims 1997] as User and Community catalogs. All empty documents in either one of the catalogs have been ignored in the experiments. We are first going to show experimental results for the Naive Bayes classifier.

8.4.1.1 Naive Bayes classifier

Figure 8.3 shows the F_1 performance of the *related retrieval* service dependent on the reduced term set size T'.

It can be seen in Figure 8.3 that the category-centric approach C1 outperforms the document granular approach D. The exact numbers show that C1 performs up to 16% better than D with the macro-averaged measure and up to 20% better than D with the micro-averaged measure. There is no structural matching performed for the Reuters catalog. Thus, the performance advantage of C1 is due to the fact that document voting corrects classification errors, as long as the correct classifications constitute the majority of votes. Moreover, Figure 8.3 shows that a T' can be found, such that $F_1 > 0.5$ in the macro-averaged as well as the micro-averaged case. Thus, C1 performs well enough for the



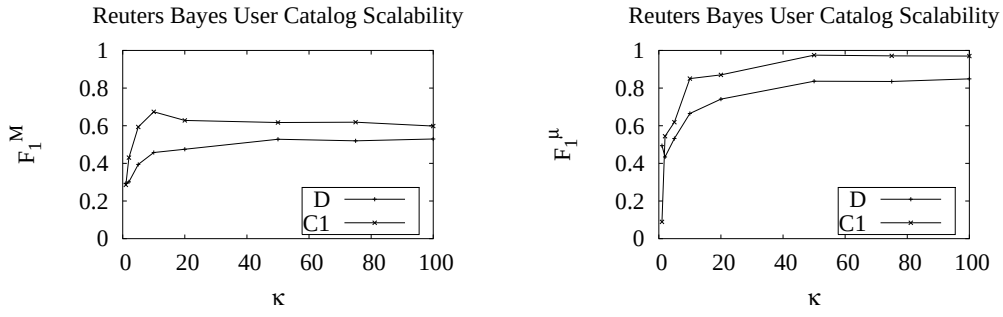
(a) Macro-averaged F_1 measure (b) Micro-averaged F_1 Measure

$$\mathcal{T}' \in [100, 2000]$$

FIGURE 8.3: *Related retrieval* service performance on Reuters catalog with modified modApté split, Naive Bayes classifier with distribution determined category priors, macro-/micro-averaged F_1-Measure: D vs. C1

knowledge exchange services presented in Chapter 4. Figure 8.3 shows that both approaches perform better with increasing T'. However, it can be seen that the performance saturates at $T' \approx 500$, which allows to improve runtime performance over higher values of T'. The performance difference between Figure 8.3(a) and Figure 8.3(b) shows that both approaches do not perform as well on categories with few training documents as on categories with a large number of training documents.

In the next experiment, we are going to examine the scalability of the approaches C1 and D with the maximum number of documents in each category of the user catalog, that are used for classifier training. Figure 8.4 shows the dependency of the F_1 performance on κ, the maximum number of training documents per user category.



(a) Macro-averaged F_1 measure (b) Micro-averaged F_1 Measure

$$\kappa \in [1, 100],\ T' = 2000$$

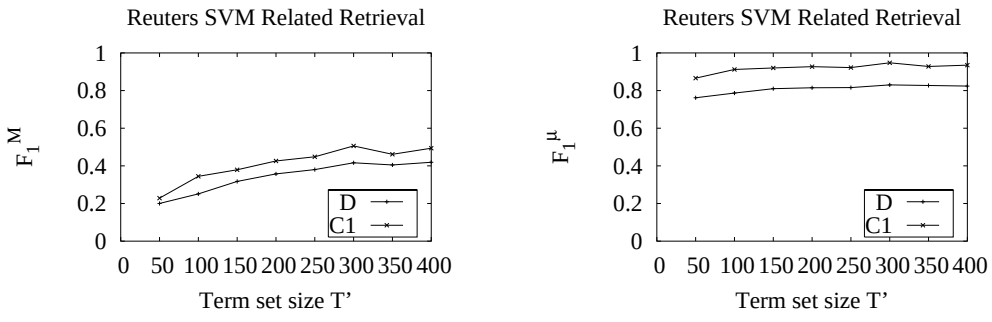
FIGURE 8.4: *Related retrieval* service user catalog scalability on Reuters catalog with modified modApté split, Naive Bayes classifier with uniform category priors, macro-/micro-averaged F_1-Measure: D vs. C1

It can be seen in Figure 8.4 that for both the macro- and micro-averaged case, the performance is low for $\kappa < 10$. This is due to the fact that the trained classifier cannot distinguish categories well enough, if there are not enough positive training examples characterizing each category. It can also be seen in Figure 8.4(a) that for the macro-averaged F_1-measure the peak performance is reached at $\kappa = 10$, i.e. a maximum of 10 documents per category in the user catalog. In the macro-averaged

measure, categories with few training examples have equal weight as categories with a large number of training examples. With increasing κ, the generality (i.e. the number of training documents) of the larger categories increases, whereas the generality of small categories remains constant. With increasing generality of the larger categories, more and more documents will be classified into the larger categories. This leads to a performance increase for the larger categories, but an even stronger relative performance decrease for the smaller categories. The combined effect is a performance decrease with increasing κ above the optimum. This effect does not occur in the micro-averaged case in Figure 8.4(b), where the performance increases monotonically. In conclusion, it can be stated that a maximum number of $\kappa = 50$ documents per category is sufficient to exploit the full performance potential on the Reuters catalog. A number of $\kappa = 10$ documents per category is sufficient for good enough performance for the knowledge exchange services presented in Chapter 4.

8.4.1.2 SVM classifier

Figure 8.5 shows the dependency of the F_1 performance on the reduced term set size T' using an SVM classifier.



(a) Macro-averaged F_1 Measure (b) Micro-averaged F_1 Measure

$\mathcal{T}' \in [100, 2000]$, $c_{SVM} = 1$, $d_{SVM} = 1$

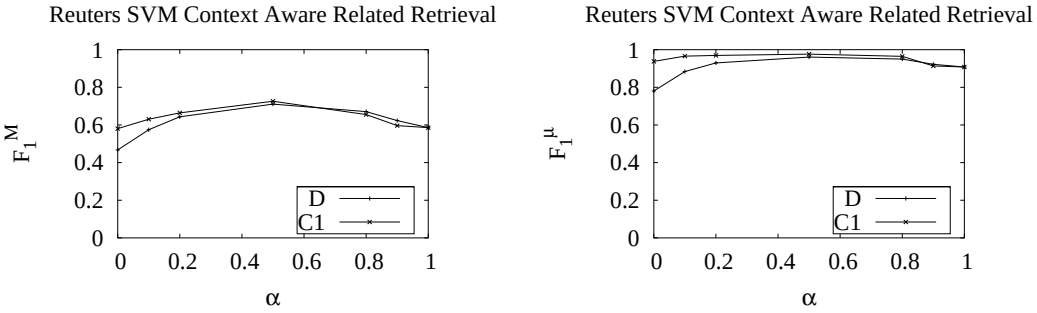
FIGURE 8.5: *Related retrieval* service performance on Reuters catalog with modified modApté split, SVM classifier, macro-/micro-averaged F_1-Measure: D vs. C1

Figure 8.5 shows that C1 outperforms D also for SVM classification. However, the absolute performance is not as good as with the Naive Bayes classifier (see Figure 8.3)[12]. As can be seen in Figure 8.5(a), for small categories, the performance is just above 50% for $T' = 300$. For larger categories, the SVM classifier can compete with the Naive Bayes classifier. Overall, the SVM performance is sufficient for the knowledge exchange services presented in Chapter 4, but leaves some room for improvement.

The performance of the *related retrieval* service using an SVM classifier can be improved using context aware classification, as is shown in Figure 8.6.

Figure 8.6 shows that using context aware classification in both C1 and D leads to a significant performance improvement compared to using regular SVM classification. In the macro-averaged case (Figure 8.6(a)), the performance can be improved by up to 43%. In the micro-averaged case, the performance improvement is a maximum of 3%. In both cases, the maximum improvement can be achieved with $\alpha = 0.5$, i.e. both the document vector as well as the category vector have equal weight.

[12]We ran the experiments with only up to $T' = 400$, because earlier experiments showed, that larger values for T' do not lead to a performance improvement.



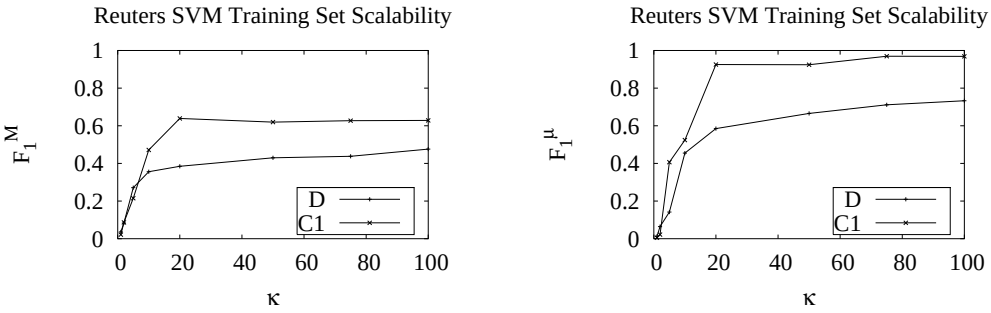
(a) Macro-Averaged F_1 Measure (b) Micro-Averaged F_1 Measure

$\alpha \in [0,1]$, $\beta_R = 4$, $\gamma_R = 0$, $\mathcal{T}' = 300$

FIGURE 8.6: *Related retrieval* service performance on Reuters catalog with modified modApté split, context aware SVM classifier, macro-/micro-averaged F_1-Measure: D vs. C1

The context aware SVM approach also outperforms the Naive Bayes approach by 53% for F_1^M and by 3% for F_1^μ. The performance difference of C1 and D with a context aware SVM classifier is small and both provide an excellent basis for the CAIMAN knowledge exchange services.

The last experiment for the Reuters catalog examines the scalability of the SVM approach with the user catalog category size. The results can be seen in Figure 8.7.



(a) Macro-averaged F_1 Measure (b) Micro-averaged F_1 Measure

$\kappa \in [1,100]$, $\mathcal{T}' = 400$, $c_{SVM} = 1$, $d_{SVM} = 1$

FIGURE 8.7: *Related retrieval* service user catalog scalability on Reuters catalog with modified modApté split, SVM classifier, macro-/micro-averaged F_1-Measure: D vs. C1

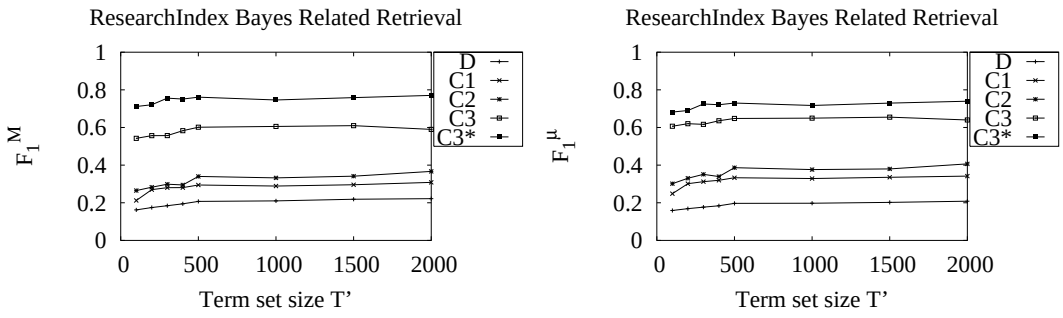
The results in Figure 8.7 for the SVM approach are very similar to the results for the Naive Bayes approach. The performance is low for small values of κ and saturates for values of $\kappa > 20$. Compared to the Naive Bayes case, the threshold here is higher for small categories, which can be seen in Figure 8.7(a), and lower for large categories, which can be seen in Figure 8.7(b). These results mean that in a catalog where the difference between the average category size and the actual number of documents used is large, using the SVM classifier leads to better performance. On the other hand, if the difference is small, Naive Bayes is the better choice. The reason for this is that the SVM classifier is less subject to overfitting than the Naive Bayes classifier [Sebastiani 2002].

8.4.2 Researchindex catalog

We use 2-fold cross validation to generate the User and Community catalog, i.e. the original catalog is split in half and each of the two halves is used as the user catalog first and as Community catalog in a second run. The results here represent the average of the two runs. We begin with results for the Naive Bayes classifier.

8.4.2.1 Naive Bayes classifier

Figure 8.8 shows the performance of the *related retrieval* service, dependent on the reduced term set size T', for different approaches.



(a) Macro-averaged F_1 measure (b) Micro-averaged F_1 Measure

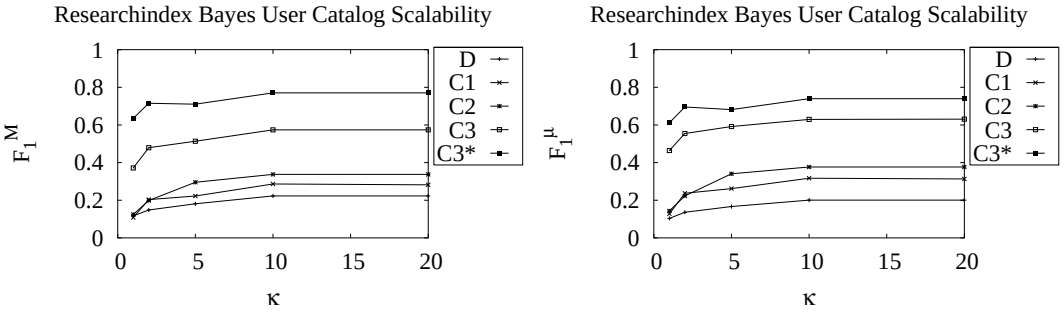
$$\mathcal{T}' \in [100, 2000]$$

FIGURE 8.8: *Related retrieval* service performance on Researchindex catalog with 2-fold cross-validation, Naive Bayes classifier, macro-/micro-averaged F_1-Measure: D vs. C1 vs. C2 vs. C3 vs. C3*

In addition to the four standard approaches defined in Section 8.2.3.2, Figure 8.8 shows the additional approach C3*, which differs from C3 only by using a $[1,1]-[1,1]$ match filter cardinality with stable marriage condition. It can be seen in Figure 8.8 that all category granular approaches C outperform the document granular approach D by far. The exact numbers show that **C3* is up to 439% better than D** using the macro-averaged measure, and up to 429% better than D using the micro-averaged measure. The performance difference between the macro-averaged and the micro-averaged case is negligible, because the categories all contain the same number of documents. Just like with the Reuters catalog, the superiority of C1 over D is due to the error correcting effect of document voting. The additional performance gain in C2 results from the positive influence of the similarity inheritance structural matching. However, the best performance among all approaches is delivered by C3 and C3*, which both use similarity flooding structural matching. In C3*, the match filter together with the stable marriage condition avoid that categories with high generality in the user catalog are found as a match for several categories in the community catalog. The performance of all approaches increases slightly with increasing T' and saturates at $T' \approx 500$. The overall performance of all approaches is not as good as for the Reuters catalog. However, the performance of C3 and C3* is largely sufficient for the CAIMAN knowledge exchange services.

The experiment depicted in Figure 8.9 shows the performance dependency on the number of documents in the user catalog used for classifier training.

Figure 8.9 shows that although the performance increases slightly with κ, the performance influence of κ is much smaller than with the Reuters catalog. The reason for the lower sensitivity is that



(a) Macro-averaged F_1 measure (b) Micro-averaged F_1 Measure

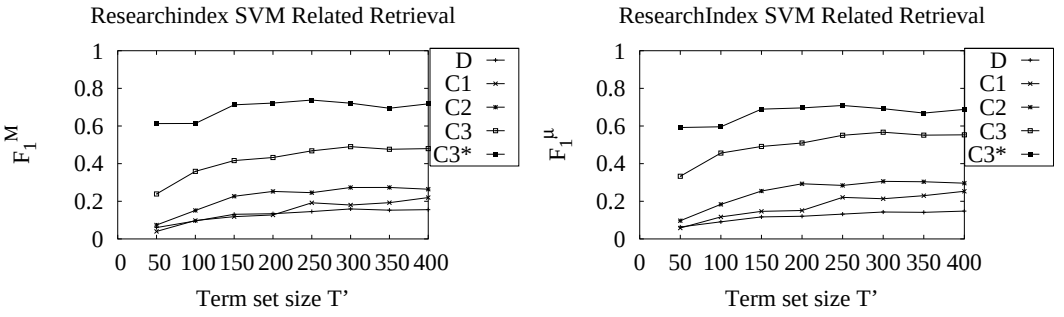
$\kappa \in [1, 100]$, $\mathcal{T}' = 2000$ $\alpha_i = 0.3$, $\mu = 0.02$, $[0, 1] - [0, n]$ Filter, uniform category priors, stemming, stop word removal

FIGURE 8.9: *Related retrieval* service user catalog scalability on Researchindex catalog with 2-fold cross validation, Naive Bayes classifier, macro-/micro-averaged F_1-Measure: D vs. C1 vs. C2 vs. C3 vs. C3*

documents in the Researchindex catalog are much larger in size than for the Reuters catalog. Thus, one Researchindex catalog already supplies as much training data as several Reuters documents. In conclusion, we can say that in a catalog with large documents, as few as two documents per category in the user catalog do not constitute a threat to the service performance.

8.4.2.2 SVM classifier

The results in Figure 8.10 shows the service performance using an SVM classifier.



(a) Macro-averaged F_1 Measure (b) Micro-averaged F_1 Measure

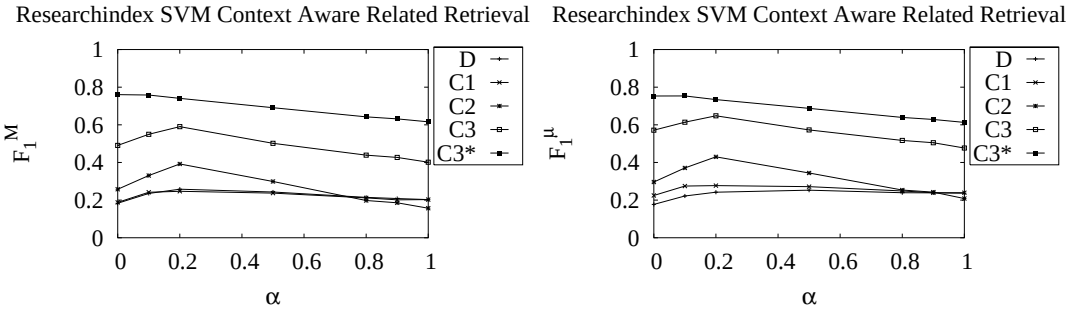
$\mathcal{T}' \in [100, 2000]$

FIGURE 8.10: *Related retrieval* service performance on Researchindex catalog with 2-fold cross validation, SVM classifier, macro-/micro-averaged F_1-Measure: D vs. C1 vs. C2 vs. C3 vs. C3*

In Figure 8.10, the performances of D and C1 are close to each other. However, C2 and especially C3 and C3* outperform D by far. The maximum **performance advantage of C3* over D is 1013%**, at $T' = 100$. Looking at the absolute performance, only C3* is suitable as the basis for the CAIMAN knowledge exchange services. Moreover, using the Naive Bayes classifier outperforms the SVM classifier by up to 16% (see Figure 8.8).

Whether or not the performance of the SVM classifier can be improved through context aware

classification, can be seen in Figure 8.11.



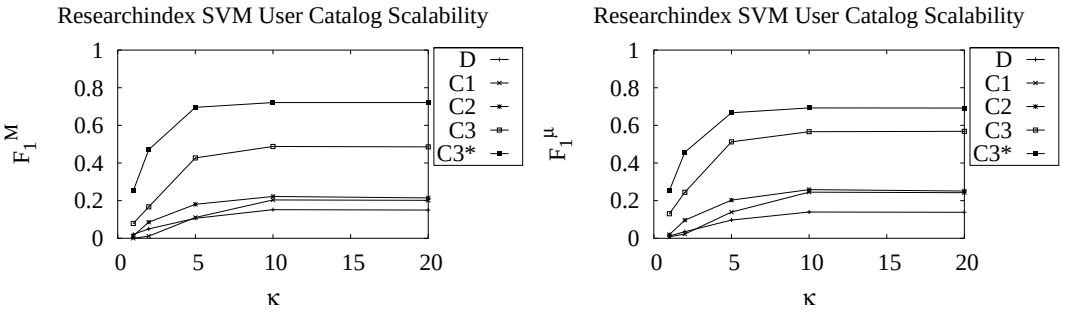
(a) Macro-averaged F_1 Measure (b) Micro-averaged F_1 Measure

$\mathcal{T}' = 300$

FIGURE 8.11: *Related retrieval* service performance on Researchindex catalog with 2-fold cross validation, context aware SVM classifier, macro-/micro-averaged F_1-Measure: D vs. C1 vs. C2 vs. C3 vs. C3*

As can be seen in Figure 8.11, the performance of the SVM based service can be improved by context aware classification for all approaches except C3*, for which the performance even deteriorates. The best performance for all approaches except C3* is achieved for $\alpha = 0.2$. D profits most from context aware classification, showing a 61% improvement. C1 still gains 36% , C2 43%, and C3 20% performance over the comparable non-context aware approach (see Figure 8.10). Compared to the Naive Bayes based performance (see Figure 8.8), context aware SVM performance is better for D and C2, worse for C1 and equally good for C3 and C3*. The performance of C3 and C3* is largely sufficient for the CAIMAN knowledge exchange services. D and C1 perform equally well and are outperformed by C2. **The only approaches, which perform largely well enough for the CAIMAN knowledge exchange services are C3 and C3***.

Finally, we are going to examine the scalability of the SVM-based service performance with the category size of the user catalog. The results can be seen in Figure 8.12.



(a) Macro-averaged F_1 Measure (b) Micro-averaged F_1 Measure

$\kappa \in [1, 20]$, $\mathcal{T}' = 400$, $c_{SVM} = 1$, $d_{SVM} = 1$, $\alpha_i = 0.3$, $\mu = 0.02$

FIGURE 8.12: *Related retrieval* service user catalog scalability on Researchindex catalog with 2-fold cross validation, SVM classifier, macro-/micro-averaged F_1-Measure: D vs. C1 vs. C2 vs. C3 vs. C3*

As is shown in Figure 8.12, the SVM-based service exhibits a sharp performance decrease for

small κ. The Naive Bayes based service outperforms the SVM based service in terms of scalability (see also Figure 8.9). Just like with the Reuters catalog, the relative performance of Naive Bayes compared to SVM depends on the difference between the average category size and κ. If the difference is small, the Naive Bayes based approach scales better, whereas if the difference is large, the SVM based approach scales better. For small differences, the SVM based approach generalizes too quickly.

8.5 Summary

In this chapter, we showed that the CAIMAN catalog matching approach is a suitable basis for the knowledge exchange services presented in Chapter 4: it allows for high-quality mediation and is scalable with large catalogs. We also proved that with the CAIMAN catalog matching approach, a better knowledge exchange service performance can be achieved than with state-of-the-art document granular approaches. Moreover, the CAIMAN catalog matching approach performed well enough for the semi-automatic knowledge exchange services in all cases.

We compared the CAIMAN matching performance to document granular approaches in terms of quality of service that can be delivered for the *Related Information Retrieval* service. We chose automatic evaluation for our comparison, because the outcome of a user study is determined by many factors that are difficult to control and track. For the automatic evaluation, the original catalog is split into a user and a community catalog. The experiments are performed with several cross validation runs. We chose the F_1-measure as performance measure because it resembles a tight quality requirement for an average user.

We examined the matching quality influence of the reduced term set size $\mathcal{T}'$, the maximum classifier training set size κ and the relative category vector weight for context aware classification, α. For all other parameters, we have adopted optimal values published in related work, or run special experiments to determine near-optimal values.

The experiments have been conducted on two catalogs with different characteristics: the **Reuters 21578 collection** of categorized news feeds and the **Researchindex** collection of computer science research papers.

We have compared four different approaches in their quality of service:

- **D**: Document granular approach / text categorization techniques.
- **C1 (CAIMAN 1)**: Category granular approach without structural matching.
- **C2 (CAIMAN 2)**: Category granular approach with similarity inheritance structural matching.
- **C3 (CAIMAN 3)**: Category granular approach with Similarity Flooding structural matching and *Retrieval* match filter.
- **C3***: Like C3, but with *Stable Marriage* match filter.

We have implemented a prototype of the CAIMAN mediation component to run our experiments. The prototype provides all necessary **classifier training and categorization functionalities**, **structural matching** algorithms, **evaluation measures calculations** as well as **experiments batch processing**. Although more classification techniques are available in the prototype, we performed experiments only for Naive Bayes classification and Support Vector Machine classification, because these two approaches represent extremes in the complexity-performance trade-off, as previously published results show.

- Our experiments showed the following results concerning the **relative performance of the approach D and the approaches C1-C3***:
  - Except for a few irrelevant outliers, all C-approaches outperform the D-approach in all experiments.
  - The maximum performance gain occurs on the Researchindex catalog using an SVM classifier: C3* outperforms D by **1013%**, i.e. C3* performs 10 times better than D.
  - Generally the performance of the different approaches can be characterized as D $\prec$ C1 $\prec$ C2 $\prec$ C3 $\prec$ C3*, where $\prec$ indicates better quality of service.
  - The quality of one of the C-approaches is always sufficient for the knowledge exchange services, whereas the quality of the D-approach is not always sufficient.
- Regarding the **different catalogs**, we found that the service quality of all approaches is better for the simpler Reuters catalog than for the Researchindex catalog. The Reuters catalog has a larger number of documents per category.
- The **parameters** we examined had the following influence:
  - The service quality increases with T', but $T' = 500$ seems largely enough to assure near maximum performance in all cases.
  - The quality can become very low for $\kappa < 10$ and increases with increasing κ. A maximum value of $\kappa = 50$ ensured maximum quality in all cases. If the documents in a category are sufficiently large, κ has a smaller influence on the quality.
  - Context aware classification generally increased the service quality with a maximum improvement of 61%. The quality was optimal for $\alpha = 0.5$ for the Reuters catalog and $\alpha = 0.2$ for the Researchindex catalog.
- Regarding the **different classification techniques**, we found the following:
  - In general, the relative performance of the techniques can be characterized as SVM $\prec$ Naive Bayes $\prec$ SVM Context Aware.
  - The Naive Bayes approach does not lose as much of its performance with smaller training categories as the SVM approach, i.e. Naive Bayes scales better with the training set.
  - An exception to the above is that if a very small fraction of the original document set of a category is used for training, SVM outperforms Naive Bayes.

Part IV

Conclusion

Chapter 9

Conclusion

In this chapter, we summarize the most important results of this work. We focus on the unique contributions of this work relative to existing publications. Moreover, based on the achieved results, we are going to identify areas for future research.

9.1 Summary of contributions

In this work, we presented the concept for the CAIMAN framework, which can support the exchange of knowledge in communities of interest via document catalogs. The knowledge exchange is supported by a semi-automatic document catalog mediation infrastructure for heterogeneous document catalogs. We have implemented a prototype of the catalog matching component to provide experimental evidence that our mediation approach is feasible and can support knowledge exchange better than existing approaches.

Application scenario

In order to be able to specify knowledge exchange services that are especially targeted for communities, we presented a community scenario. Our scenario defines the environment for which CAIMAN is targeted:

- A virtual community of interest, the members of which see knowledge exchange as one of the community purposes.
- Users organize their document collections in *personal catalogs*, communities as a whole share a common *community catalog* and additionally, there are *global catalogs*, which are maintained by community-independent catalog providers.
- The different catalog scopes and categorization schemes of the involved catalogs mirror the domain of interest and the domain perspective of the catalog creator.

In order to be able to design a framework for knowledge exchange especially for our scenario, we presented an account of characteristics of communities from existing works. In our scenario, knowledge can only exist in individuals, everything that can be stored in catalogs is information. Consequently, we define knowledge exchange in our scenario as a *virtual knowledge exchange*, which is achieved by exchanging information in a way that promotes the knowledge exchange process.

Requirements for knowledge exchange support in communities

The functionalities of many existing systems for supporting knowledge exchange in communities are based on intuitive heuristics, which fail to take into account the special requirements of the community environment. The features of the CAIMAN framework are based on a systematic analysis of the knowledge exchange process as well as special requirements for communityware design.

We split the knowledge exchange process into four consecutive phases, based on a common process model from the field of Knowledge Management (KM): 1) *knowledge distribution*, 2) *knowledge awareness*, 3) *knowledge creation* and 4) *knowledge application*. For each of these phases, support requirements have been published, which we transferred to our scenario. Additionally, we considered some psychological characteristics of human information processing to complement the KM requirements. Finally, we also identified general requirements for community support applications, based on requirements for groupware.

We analyzed different ways of organizing knowledge exchange with the document catalogs that are available in our scenario. We showed that using all catalogs and relying on a semi-automatic mediation infrastructure is the most advantageous solution. However, for using all catalogs, a catalog mediation infrastructure is required that provides for high-quality catalog mediation and access to heterogeneous document catalogs.

Application level design of the CAIMAN framework

Based on the identified requirements, we have designed knowledge exchange supports services for the CAIMAN framework. The CAIMAN services support the exchange of knowledge between community members via document catalogs. CAIMAN is the first framework to date for this purpose, for which the functionalities have been systematically designed based on requirements of the knowledge exchange process. CAIMAN also avoids the shortcomings of most related systems that we have reviewed. CAIMAN supports knowledge exchange for the knowledge types as defined in the application scenario for each of the phases of the knowledge exchange process:

- **Knowledge types:** Document content knowledge, document topic knowledge, catalog domain structure knowledge, subjective document relations knowledge.
- **Knowledge exchange phases:** Knowledge distribution, knowledge awareness, knowledge creation, knowledge application

The CAIMAN framework offers three services to support the above exchange phases for the different knowledge types:

- The **information publication** service allows the user to publish documents from her personal catalog to other catalogs without additional categorization effort. This service supports **knowledge distribution**.
- The **related information retrieval** service retrieves related documents from mediated catalogs for a given category in the personal user catalog. This service supports **knowledge awareness, creation and application**.
- The **category discovery** service allows the user to discover new relevant categories in mediated catalogs by browsing through her own personal catalog. This service supports **knowledge awareness, creation and application**.

We showed that the CAIMAN services require a catalog mediation infrastructure. The mediation infrastructure has to support the exchange of all knowledge types, provide for high quality and dynamic mediation and scale well with large document catalogs.

CAIMAN document catalog mediation principle

We have introduced the concept of a mediator and showed that the mediator needs to be able to perform *catalog matching*, *query mapping* and *query processing* tasks to provide the virtual catalog integration that is required in the mediation process. Additional *wrapper* components are required for the conversion of data formats. Mediation in CAIMAN has to adhere to the application requirements of the knowledge exchange services. As an additional application requirement, we explained that name labels of the catalog structure cannot be used to calculate the catalog integration solution, because their subjective nature would impede a high-quality mediation.

We have differentiated between **document granular mediation**, where virtual catalog integration is performed for each document individually, and **category granular mediation**, where virtual catalog integration is performed for all documents within the same category. Document granular catalog mediation can be realized by state-of-the-art text classification techniques. Category granular catalog mediation requires a catalog matching approach, and among existing related approaches, there is none that can be directly applied to our catalog scenario. We have compared the two general mediation approaches with respect to the knowledge exchange services and showed that a category granular approach has more advantages in our scenario. In particular, with category granular mediation, the matching calculations are less costly, fewer documents have to be transferred over the network, less user workload is incurred in the semi-automatic mediation process, all knowledge types can be exchanged and cold start problems can be completely avoided.

We have introduced the coarse concept of the CAIMAN category granular mediation approach, which fulfills the requirements of the exchange services. The CAIMAN mediator concept comprises a catalog matching component and query processing component that includes wrapper functionalities. Finally, we described the semi-automatic mediation process in CAIMAN.

The CAIMAN approach to document catalog matching

Existing matching techniques for conceptual structures are not directly applicable to our catalog scenario. Thus, we have presented the novel CAIMAN document catalog matching approach, which satisfies the application requirements of the knowledge exchange services. Our matching approach is the basis for semi-automatic category granular mediation of document catalogs.

The CAIMAN matching approach employs text classification techniques, which we have described along with suitable performance measures. Our matching approach consists of a sequence of three phases:

- In the **classification phase**, documents from each category of the source catalog are temporarily categorized in categories in the destination catalog using techniques for automated text classification.

- The **category similarity phase** calculates initial similarity values for all pairs of categories from the source and destination catalog based on the classification results from the classification phase.

- In the **structural matching phase**, the initial similarities form the category similarity phase are propagated between category pairs according to the graph structure of the catalogs. The final match pairs are filtered out with specialized match category filters.

Using the CAIMAN catalog matching approach for mediation has the following advantages over existing document granular approaches:

- Better flexibility and scalability with large and changing catalogs.
- Knowledge about subjective relations betweens documents can be exchanged with the knowledge exchange services.
- Exchange of knowledge about domain structure is possible with the knowledge exchange services.

Querying heterogeneous document catalogs with CAIMAN

W have presented an approach that allows heterogeneous catalogs to be queried with the same query infrastructure on a data model level. Our main contributions have been the following:

1. RDF as the lingua franca for catalog data allows for joint querying. We have shown how different document catalog data models can be modelled with RDF, following a layered approach to data interoperability from [Melnik and Decker 2000].
2. As a proof of concept, we introduced an approach for loss-free modelling of Topic Maps data as RDF data.

Additionally, we proposed a coarse exemplary concept for joint querying of catalogs that allows to perform the necessary query mapping and is applicable to different data models. The mediation concept is based on the TRIPLE [Sintek and Decker 2002] RDF query infrastructure. We showed how TRIPLE rules can be used for a declarative specification of some of the necessary query mappings. We introduced a coarse software architecture for the querying component in CAIMAN.

Our querying approach has the following advantages:

- Catalogs based on different data models can be queried with the same infrastructure.
- In a scenario with n different data models, only $n - 1$ converters are required.
- The data model conversions are simple and straightforward graph transformations.
- With TRIPLE, the mapping rules can be defined declaratively for all data models. Moreover, using data model specific mapping rules, the mapping information that has to be supplied by the user can be reduced to a minimum.

CAIMAN knowledge exchange service performance evaluation

We showed that the CAIMAN catalog matching approach is a suitable basis for the CAIMAN knowledge exchange services, as it delivers high-quality mediation results and scales well with large catalogs. Moreover, with the CAIMAN catalog matching approach, a better knowledge exchange service quality can be achieved than with state-of-the-art document granular approaches. Our experiments proved that the CAIMAN catalog matching approach performs well enough for the knowledge exchange services in all cases.

We performed an automatic comparison of the CAIMAN matching approach to document granular approaches in terms of quality of service that can be delivered for the *Related Information Retrieval* service. We examined the matching quality influence of some important parameters. For all other parameters, we have determined optimal values published in related works. The experiments have been conducted on two catalogs with different characteristics: the **Reuters 21578 collection** of categorized news feeds and the **Researchindex** collection of computer science research papers. We performed experiments using Naive Bayes text classification and Support Vector Machine classification in both document and category granular approaches. For our experiments, we have implemented a prototype of the CAIMAN mediation component.

We have compared five different approaches with respect to their quality of service: one document granular approach with different classification techniques and four category granular approaches with different structural matching algorithms and match filters. Our experiments showed the following results concerning the **relative performance of the document granular and the category granular approaches**:

- The category granular CAIMAN approach gives better service quality than the document granular approach.
- The quality of the category granular approach is always sufficient for the knowledge exchange services, whereas the quality of the document granular approach is not always sufficient.
- The CAIMAN approach scales well with large catalogs.
- The maximum performance gain occurs on the Researchindex catalog using an SVM classifier: The category granular approach with similarity flooding structural matching and stable marriage match filter performs 10 times better than the state-of-the-art document granular approach.
- The Similarity Flooding structural matching performs better than the similarity inheritance structural matching and for both the performance is better than without structural matching.
- The stable marriage filter performs better than the retrieval filter.

Regarding the different catalogs, we found that the service quality is better for the simpler Reuters catalog, which has a larger number of documents per category.

Main research contributions of this work

We see the major contributions of this work in three different areas:

- **Application level:** We have collected requirements for an application that supports knowledge exchange in communities via document catalogs. We have systematically designed application services according to these requirements. The features of existing comparable systems have not been designed in such a systematic manner and consequently all lack features for certain aspects of the knowledge exchange.
- **Matching of document catalogs:** We have proposed a novel catalog matching approach. Our approach is a complex combination of text classification techniques and graph matching techniques. Used for catalog mediation, our new approach outperformed existing mediation approaches by far. Besides the superior performance, our approach is scalable with large catalogs. Moreover it is also applicable in related fields like schema matching and ontology matching.

- **Querying of heterogeneous catalogs:** We have introduced a coarse concept for joint querying of heterogeneous document catalogs. Our approach requires the conversion of all catalog data to a common format. We showed, how this loss-free, straightforward conversion can be performed in general, and in detail for the popular formalisms RDF and Topic Maps.

9.2 Future work

The issue of supporting knowledge exchange via document catalogs has been covered extensively in this work. We are now going to discuss some useful extensions of the results of this work.

9.2.1 Application framework

Requirements. We have motivated the requirements for the CAIMAN framework from abstract principles from the field of Knowledge Management, principles of human information processing and general requirements for groupware applications.

We can see a potential benefit in an empirical evaluation of the CAIMAN services in order to find room for improvements. The results of user studies can be the basis for new requirements, which can in turn lead to an improvement of the CAIMAN services. Moreover, previously published user studies that are applicable to our scenario can be another source of useful requirements for a more detailed design of the CAIMAN functionalities.

An aspect that has been briefly mentioned in Chapter 3 is the creation phase of document catalogs. Although the catalog creation phase is principally not excluded in the CAIMAN mediation approach, we have focused on the phase when catalogs are relatively stable. The catalog creation phase is an interesting subject for further examinations, because it is then, when most knowledge about a domain is required, in order to be able to structure the domain in a meaningful way. On the one hand, collaborative catalog construction could help the catalog creators to benefit from each other. On the other hand, collaboration may lead to similar or converging catalog structures in the different catalogs. A lot of the problems that occur with different catalog granularities and catalog matching in general could possibly be mitigated through a collaborative catalog construction phase.

Application level design. To be able to conduct empirical evaluations of the CAIMAN services, the CAIMAN framework has to be fully implemented and integrated into an existing community support application that includes catalog management functionalities. The implementation also includes the design and implementation of wrappers for different document management systems, which are often the basis of community catalogs. On the one hand, the full implementation is required for evaluation, on the other hand, the evaluation results can lead to an extension of the CAIMAN functionalities.

9.2.2 Mediation infrastructure

Catalog querying. One central idea of the presented catalog querying approach was to use RDF as the lingua franca for data exchange between all involved catalogs. The presented concept for querying the different catalogs simplifies the integration of new catalog representation techniques besides Topic Maps. An interesting extension of the existing query approach could be the inclusion of additional catalog wrappers, both for (semi-)formal as well as informal catalog sources.

In the current query approach, the user has to provide some information about the catalog structure in order to allow the catalogs to be jointly queried. An extension of the query approach could include heuristics to allow the required information to be automatically extracted from the catalog data.

Document catalog matching. The CAIMAN document catalog matching approach is currently constrained to matching one catalog perspective with one perspective in another catalog. The information, which perspectives are to be matched, has to be supplied by the user. The automatic detection of similar perspectives in mediated catalogs can be a useful extension of the existing matching approach. Alternatively, the automatically detected perspectives can be corrected by the user in a semi-automatic process.

As the CAIMAN approach relies on text classification techniques, which in turn use term statistics, it is constrained to catalogs with documents in the same language. A possible extension to the existing matching approach could be a cross-language matching capability. This extension can be realized by simply using translation on a term basis.

As we have seen in Chapter 6, different granularities of two catalogs can become a problem for matching. The resolution of the granularity problem can be a useful extension to the CAIMAN matching approach. A possible solution to this problem can be found in two steps: first, the occurrence of different granularities has to be localized in the catalogs. An indication for different granularities is that several categories in one catalog are matched with just one category in another catalog. After localization of the granularity difference, the coarser category could be split and clustered in a way that resembles the finer granular categories. The new document clusters could replace the previous coarse category or just be used as "virtual" sub-categories of the original category. If the new, finer granular categories are used for catalog matching, the quality of the knowledge exchange services is likely to improve.

So far, the CAIMAN concept does not include special indexing techniques for hypertext documents or structured documents in general. It has been stated in [Yang et al. 2002] and [Attardi et al. 1999] that special indexing techniques are required for hypertext classification in order to achieve high classification accuracies. We have conducted first experiments that confirmed the need for special indexing techniques. An integration of hypertext indexing techniques into CAIMAN could extend the high matching quality to hypertext catalogs.

Knowledge exchange service performance evaluation. We have evaluated the CAIMAN service quality with an automatic evaluation approach. The automatic evaluation is necessary as a proof of concept that the CAIMAN mediation approach is technically feasible and produces the expected results.

In order to allow statements about the performance on catalogs with specific characteristics, experiments that are targeted on showing the performance differences between different catalogs can be performed. These results can lead to a wider overview of the CAIMAN performance.

A user study to evaluate the subjectively perceived quality of service of the knowledge exchange services can be a useful extension of the existing simulation results.

CAIMAN software architecture.

We have presented the CAIMAN framework, identified the different framework components and designed and explained the functionalities of the components. The CAIMAN components offer functionalities for supporting community knowledge exchange, however they do not represent a full-fledged community support application. We restricted the scope of the CAIMAN framework for two reasons:

1. The design of general community support functionalities and components is not the goal of this work.

2. In order to be successfully deployed in a community, the CAIMAN functionalities have to be integrated into an application that is used on a day-to-day basis by community members.

There are many possible software architectures, which can be the basis of the CAIMAN components. An important requirement is, however, that the CAIMAN components can be easily integrated into existing community support applications and that new components are easily integrable into the CAIMAN architecture itself.

The design of a flexible CAIMAN software architecture that lives up to these requirements is an important precondition for the adoption of the CAIMAN concept in practice.

We have created a first draft for a flexible CAIMAN software architecture and a concept for the integration of CAIMAN into an existing community support architecture. Both are presented in Appendix A. Our architecture draft can be the basis for future more concrete designs and community integration concepts.

9.3 Concluding remarks

Communities of interest have become increasingly valuable sources of knowledge with the advent of the Internet. However, with increasing number of community memberships and catalogs that can be used for knowledge exchange, users can easily become overwhelmed. Moreover, too much effort for knowledge exchange via catalogs impedes the exchange. The CAIMAN framework solves this problem by allowing a community member to use her personal document catalog for all knowledge exchanges. The automated mediation of document catalogs in CAIMAN greatly simplifies knowledge exchange. Moreover, our experimental results showed that the CAIMAN approach performs well enough to bring a substantial benefit to the user.

With the growing availability of sources of information, information overload increases as well. This work can contribute to extracting relevant knowledge from information glut.

Part V

Appendix

Appendix A

CAIMAN Software Architecture Draft

We are going to present some first concepts for a software-agent based architecture that allows for easy integration into existing applications as well as easy integration of new components. The presented architecture, follows principles of community design, which we have discovered in the *CoBricks* project (*Bricks for Communityware*, see also [Lacher and Koch 2000; Koch and Lacher 2000; Koch et al. 2001; Borghoff et al. 2001]). *CoBricks* is a general community support architecture that allows for the reuse of community information and interoperability of services across communities. CoBricks is well-suited as the basis for community support applications and consequently, other architectures for community support applications are likely to include many of the CoBricks principles.

The combination of the CAIMAN components with *CoBricks'* general community support components can be the basis for a future community knowledge exchange support application.

CAIMAN agent architecture. We motivate a coarse software agent based architecture for the CAIMAN framework. Agent oriented software engineering [Weiß 2001] is a high level software design approach that is especially suitable for distributed systems and applications that include heterogeneous components, for the following reasons:

- Agents communicate through *Agent Communication Languages (ACL)*, which are abstract protocols based on the concept of speech acts [Weiß 2001]. In an agent-based architecture with heterogeneous components using a common *Agent Communication Language (ACL)*, the number of interfaces between heterogeneous components is reduced from $n \cdot (n-1)/2$ to n (n the number of components).

- Agent architectures provide for a loose coupling of system components and thus flexibility and scalability in terms of new components.

- An agent-based architecture allows to provide a simplified high-level design of a complex distributed software system.

ACLs typically only define a conversation process between agents and not the content. An additional common content language, like for example KIF[1][Weiß 2001] is required to allow agents to exchange data. ACL messages are typically communicated over any Internet transport protocol, e.g. HTTP.

We transform the CAIMAN components into an agent architecture with partly autonomous agents with clearly defined functionalities. We regard the internal workings of the agents as black boxes

[1]Knowledge Interchange Format

here. The CAIMAN query functionality has been described in Chapter 7 and the CAIMAN matching approach has been described in Chapter 6. The agent architecture gives a coarse impression of interaction of the different CAIMAN components as well as distribution of components. Figure A.1 shows a coarse overview of the concept for a CAIMAN agent architecture, which incorporates many of the principles proposed for CoBricks in [Koch et al. 2001].

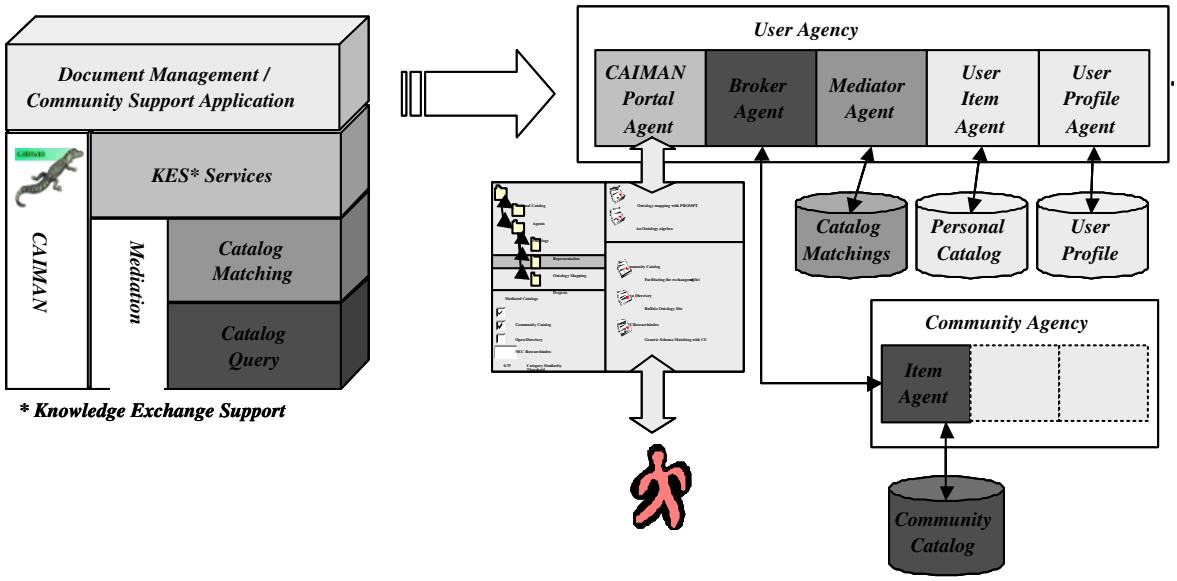


FIGURE A.1: CAIMAN Agent Architecture Concept

On the left hand side of Figure A.1, we see the CAIMAN architecture overview with the different functional components differently shaded. On the right hand side of Figure A.1, we see the division of the components into agents with specifically defined roles. The agent architecture is divided into two groups of agents: the *user agency* and the *community agency*. The two agencies group agents that are under control of the user and agents under control of the community - a global agency would largely resemble the community agency. The agents in the user agency do not necessarily run locally at the user's workstation, but the user needs to have complete control over these agents. The user agents store and manage private user information and thus the user's trust in the managing agents is essential for user participation. The user's trust can be increased significantly if the respective agents are located near the user in a sense that she can monitor and has control over the agents' activities. Moreover, if all user information is stored in a user-centric fashion, this data can be reused within several communities [Koch et al. 2001].

The example in Figure A.1 illustrates the roles of the different agents. We describe an example of how a CAIMAN knowledge exchange service can be rendered by the different agents. For our example, we consider the *related retrieval* service to be requested by the user for a certain category in her personal catalog. The following is assumed as precondition:

- All catalog matchings have been calculated at the time when the user chose the respective catalogs for mediation. The matching results are stored in a catalog matchings base with the user agency (see Figure A.1). We chose to store the matchings with the user to give her control over this data and out of the simple practical reason that most likely no community would be willing and able to store matching data for all of its members.

- The information that is necessary to query the community catalog(s) has been provided by the user and is stored with the broker agent. The catalog structures for mediated catalogs like a community catalog are either available in a format that is queryable by the broker agent or has been converted and cached with the broker agent (see Chapter 7).
- The user profile holds all necessary information about community memberships and catalogs to access during mediation.

The user chooses a category in her personal catalog, and invokes the *related retrieval* service on the category. The user interface as well as the application logic of the services is provided by the *CAIMAN portal agent*. The *related retrieval* service is performed in the following steps:

1. The list of documents that are in the chosen category of the user catalog are requested from the *broker agent* by the *portal agent* and displayed in the user interface.
2. The *broker agent* in turn requests the documents for the respective user category from the *user item agent*.
3. The *portal agent* requests all documents from mediated catalogs that are related to the current user category from the *broker agent*.
4. The *broker agent* requests the information, which catalogs are to be mediated from the *user profile agent*. For each of the catalogs, the broker then requests the matching category from the *mediator agent*. And finally, the broker requests from each of the mediated catalogs, i.e. the respective *item agents*, all documents within the respectively matching categories.
5. The final result is returned to the *portal agent*, which can now display the results of the *related retrieval* service.

The service example explained the roles of the agents in Figure A.1. There are a number of issues concerning an agent based CAIMAN framework that remain as future work:

- **Agent platform:** Agent systems typically require platforms that offer certain basic agent services, such as directory, communication and security services [Weiß 1999].
- **Agent communication:** Given that all agents communicate using ACL, a common content language is required. KIF[2] is a possible candidate. The Foundation for Intelligent Physical Agents (FIPA) has also published several proposals for agent communication [Steiner 1998].
- **Query language:** Depending on the content language, queries concerning document catalog contents and query results can either be "wrapped" as black-box content in a general content language or translated into the agent content language.
- **Distributed query services:** The capability of catalog querying, as described in Chapter 7 has to be integrated into the CAIMAN broker and item agents.
- **Implementation** of a CAIMAN agent-based architecture.

Another open issue is the integration of the CAIMAN components into existing community support applications or frameworks, such as the CoBricks framework.

[2]Knowledge Interchange Format

Integration into the CoBricks community support framework. The *CoBricks (Bricks for Community Support)* framework [3] [Koch et al. 2001; Borghoff et al. 2001] is a general community support framework. CoBricks can be the basis for community support applications that can profit from the integration of the CAIMAN services. We regard CoBricks as a framework that is in many ways representative for community support frameworks and community support applications in general. Thus, we see a concept for an integration of the CAIMAN framework into the CoBricks framework as an effort that could produce many general implications for the integration of CAIMAN into community support applications. Our aim for this section is to provide an impression of the issues regarding the integration of CAIMAN services into a community support application

The CAIMAN agent architecture follows the same design principles as the *CoBricks* framework, thus there are many similarities in the two architectures. Both are flexible agent architectures and both architectures are conceptually divided into a user agency and a community agency to increase the user's trust in the system and allow for user data re-usability.

A brief introduction of the most basic components and features provided by the *Cobricks* framework helps to describe a possible integration with CAIMAN.

The agents in the *user agency* provide means for managing user information which can be reused for participation in several communities. A *user agent* manages the user profile, while an *awareness agent* monitors the user's actions to constantly update the user profile with the newly collected data. A *user broker agent* distributes queries to either the *user item agent* or one or several communities, in which the user is a member. The *user item agent* is basically a wrapper for a database or a file system, which stores the user's personal catalog. All information that is exchanged in the user agency is described in terms of a semantically rich domain model, which is managed by the *user ontology agent*. The agencies can be accessed through a user interface of the user agent or via a web browser through the *portal agent*, which is part of the community agency.

The main task of the *community agency* in *CoBricks* is to support information exchange among members of a community. The *community agent* manages memberships and skill yellow pages of the community. The *community broker agent* distributes information queries to the different *item agents* in the community. A *matchmaking agent* mines social network relationships from the data available in the community. A *filter agent* incorporates several information filtering possibilities. Just like in the user agency, a *community ontology agent* provides the domain model, which is the basis for data exchange among the agents.

The current implementation of the *CoBricks* framework is based on a proprietary agent system which adheres to the FIPA[4] standard. The agents communicate using the FIPA Agent Communication Language (ACL). Currently, simple attribute-value-pairs are used as content language with the FIPA ACL. Additional parts of the infrastructure are a standard directory service (LDAP/X.500) for finding agents and a set of security components that take care of authentication and authorization.

There are a number of issues that need to resolved for integration of the CAIMAN components into the *CoBricks* architecture that we consider future work as extension to this work. The key issues that need to be examined for integration of CAIMAN into the *CoBricks* framework are:

- **Agent communication:** The content languages of a possible CAIMAN agent framework and the CoBricks framework need to be assimilated.

- **Query language:** The query languages used for querying information resources in the community framework as well as in CAIMAN need to be integrated.

[3] See http://www11.in.tum.de/proj/cobricks

[4] See http://www.fipa.org/

- **Distributed query services:** The different query components have to be integrated.
- **Domain model coherence:** To be able to query data from the user profile and item agents in a meaningful way, the CAIMAN agents have to be able to understand and use the CoBricks domain model for querying. One solution could be to merge the user catalog structure and the CoBricks domain model and use the merged model as the basis for all communication in the user agency.
- **User interface:** The application logic of the CAIMAN services has to be integrated into the respective *CoBricks* portal agents.

Appendix B

CAIMAN Implementation

Overview

This section describes some details of usage and implementation of the CAIMAN mediation prototype implementation, which has been used to produce the experimental results presented in Chapter 8. This section complements the description in Chapter 8 in that we present additional technical detail, but we do not repeat the functional description of the prototype, which has been given in Section 8.3. The purpose of this section is to provide guidance for running experiments with the CAIMAN prototype as well as guidance for the use and reuse of components of the prototype implementation in future work.

The CAIMAN mediation prototype is tailored for automatic experimental evaluation of the CAIMAN matching approach and state-of-the-art text categorization techniques on plain text document catalogs. The prototype implementation architecture cannot be transferred to a complete implementation of the whole CAIMAN framework in a straightforward way. However, parts of the implementation are reusable for a future implementation of the CAIMAN framework.

First, we show how the implementation can be used to run evaluation experiments. Thereafter, we provide an overview of the software architecture of the prototype, including static and dynamic overview models. Finally, we describe the package structure and where the different functionalities described in Section 8.3 are implemented.

The prototype has been implemented in JAVA. However, some of the required runtime libraries can only be used on a Linux platform. Excluding the required runtime libraries, the prototype amounts to roughly 13000 lines of code.

B.1 Usage of the prototype

The prototype can be used for running experiments like the ones described in chapter 8. All experiments can be run from a central JAVA class and all parameters are configured via a configuration file.

Prerequisites

In addition to the CAIMAN packages (see section B.4), several external libraries are required to use the CAIMAN prototype. For compilation, the following JAVA libraries are required:

- LibSVM JAVA[1]: This library is required for SVM classification and for the internal object format for word-document-matrices and catalog indices.
- Xerces XML parser[2]: Preprocessing of the Reuters catalog, which consists of SGML files requires this library to extract the plain text documents.
- Similarity Flooding[3]: We use the publicly available implementation by the author of the original similarity flooding publication (see [Melnik et al. 2002]). This distribution also includes the required W3C reference implementation RDF-API as well as the Aelfred XML parser.
- EdenLib[4] is a general purpose library from which we use some text processing functionalities.

In addition to the above compile-time libraries, the following runtime libraries are required as executable binaries, which are started from the JAVA prototype:

- Bow[5]: The Bow toolkit and especially the Rainbow applications for text classification are required for indexing and classification using the Naive Bayes classifier. The path to the binaries for Linux[6] has to be provided in the CAIMAN configuration file.
- LibSVM binaries: The binaries of the LibSVM library are required for scaling training data and training of the classifiers. These tasks cannot be performed with the JAVA library for runtime performance reasons. The path to the respective binaries has to be provided in the configuration file.

Catalog preparation

The catalogs used for experiments have to be available in the file system as a hierarchical folder structure. A sub-folder signifies a sub-category and documents are represented by files contained in the folders. The structure is assumed to be strictly hierarchical, i.e. it is a *catalog perspective* in the sense that has been described in section 2.3.3.1. The top folder, which holds the category folders of the first tree level, has to be configured in the configuration file for the prototype. The categories on different tree levels in the catalog can have the same names, whereas document names must be unique on a catalog-global level. The documents in folders must be plain text unstructured documents.

The online document catalogs for the experiments in Section 8.4 have been crawled using the WebSphinx toolkit[7], and a unix-shell-script included in the CAIMAN distribution, which relies on the *GNU wget* toolkit[8]. The conversion of Postscript and PDF[9] documents to plain text has been performed with the *GNU pstotext* toolkit[10]. For HTML text files, an option in the configuration file allows to ignore the HTML tags during indexing.

For the Reuters[11] catalog, some additional preparations are necessary. The original catalog comes in several large SGML files, which have to be parsed in order to retrieve the documents and their

[1] See http://www.csie.ntu.edu.tw/ cjlin/libsvm/
[2] See http://xml.apache.org/xerces-j/
[3] See http://www-db.stanford.edu/ melnik/mm/sfa/
[4] See http://edenlib.sourceforge.net/
[5] See http://www-2.cs.cmu.edu/ mccallum/bow/
[6] The library only worked under Linux at publishing data of this work.
[7] See http://www-2.cs.cmu.edu/ rcm/websphinx/
[8] See http://www.gnu.org/software/wget/wget.html
[9] Postscripts and PDF are registered trademarks of Adobe
[10] See http://research.compaq.com/SRC/virtualpaper/pstotext.html
[11] See http://www.research.att.com/ lewis/reuters21578.html

classification. The CAIMAN distribution includes the `Preprocessor` class for preprocessing the Reuters catalog in the package `de.tum.caiman.crawl.reuters`. [Groh 2001] describes the preprocessing in more detail.

Experiments configuration

All parameters for the experiments are configured in a configuration file. For each catalog in the experiments, a separate configuration file has to be created. Also, the way in which the catalog is split for the experiments, i.e. either a given fixed split into training and test set or by k-fold cross validation, is the same for all experiments described in one configuration file.

A configuration file directly describes which graphs should be output by the CAIMAN prototype. For each graph, a number of fixed parameters settings can be given for the different mediation approaches and classification techniques. Moreover, a list of abscissa values can be given, for all of which the respective experiment is evaluated.

A graph consists of several subgraphs, which represent the actual lines drawn in the figures in Section 8.4. A subgraph can show one of the different compared approaches by specifying the respective parameters in the configuration file.

The configuration files for CAIMAN have a large number of options, which are described in more detail in [Etzold 2003].

Experiments batch processing

After all planned experiments have been described in the configuration file, the calculation of the results can be started using the class `TestParameters` in the `de.tum.caiman.experiments` package. The configuration file is accepted as an input parameter.

The results of all subgraphs and graphs are calculated and the performance measure results for each of the single subgraphs are printed on the console as progress reports.

After all calculations for a graph have been finished, the results are output in several different formats: as a table in a format that can serve as input to the Gnuplot[12] graph drawing software and as a Postscript figure.

More details on the output files of the batch process can be found in [Etzold 2003].

[12] See http://www.gnuplot.info/

B.2 Static Structure

Figure B.1 shows the main static class structure of the CAIMAN prototype. Some attributes, methods and classes have been left out to keep the overview concise.

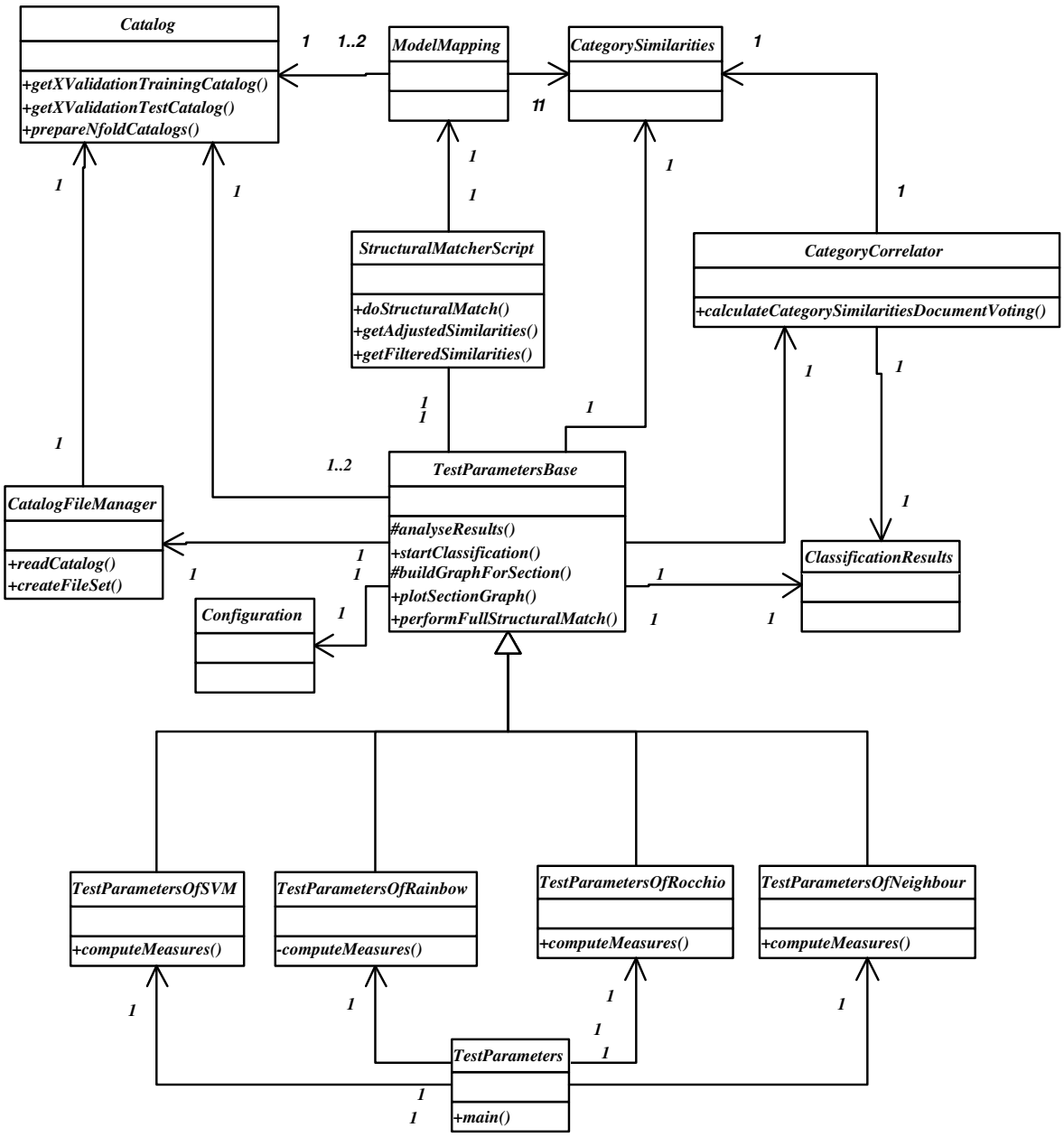


FIGURE B.1: CAIMAN mediation prototype main classes

The coarse functionalities of the classes are explained in Section B.4.

B.3 Control flow

Figure B.2 shows an overview of the sequential steps performed in an experiment of the CAIMAN prototype.

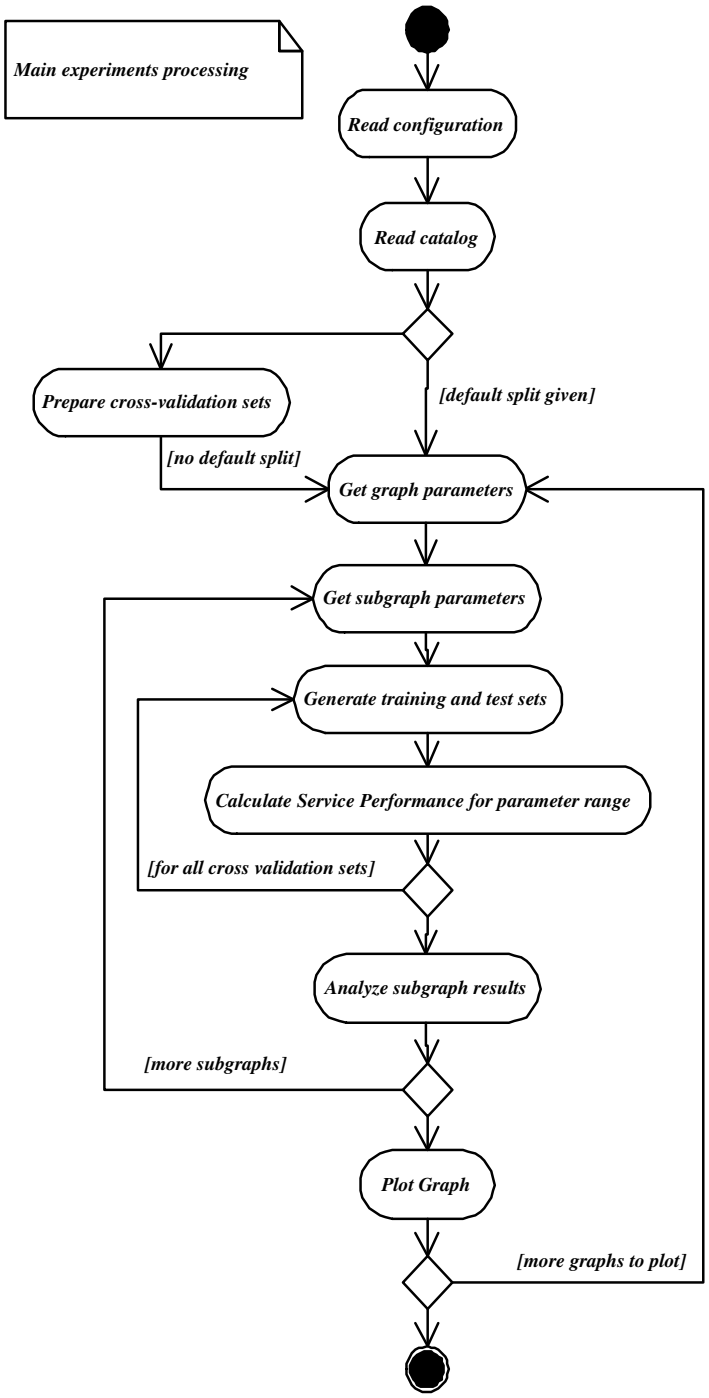


FIGURE B.2: CAIMAN mediation prototype main control flow

Figure B.3 shows an overview of the sequential steps performed to calculate the performance results of one subgraph in an experiment of the CAIMAN prototype.

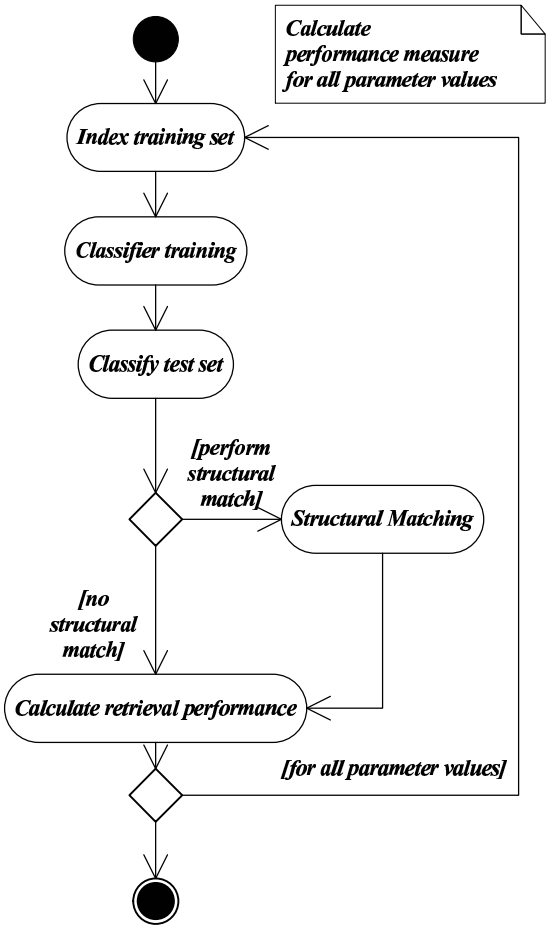


FIGURE B.3: CAIMAN mediation prototype subgraph performance calculation control flow

B.4 Functionality overview

Table B.1 shows an overview and short descriptions of the classes in the *de.tum.caiman* package.

Class	Description
Catalog	Document Catalog access and update
CatalogFileManager	Document Catalog I/O
Category	Category access and update
FileManager	General I/O functionalities
Util	General utility functionalities

TABLE B.1: Overview of the package *de.tum.caiman*

Table B.2 shows an overview and short descriptions of the classes in the *de.tum.caiman.classify.** package.

Package.Class	Description
ClassificationProbabilities	Access to the classification probabilities of documents
ClassificationResults	Access to the classification results of documents
ClassificationResultsFileManager	Classification results I/O
Classifier	Document classification decision
Formula	Simple math expression evaluation for performance measures
HierarchicalFeatureBooster	Feature weighting for hierarchical classification (not used in CAIMAN)
Measure	Calculation of performance measure
MeasureResults	Access to performance measure results for several experiments
NeighbourClassification	Classification with the k-nearest-neighbor technique
NeighbourClassification	Classification with the Naive Bayes technique using the Rainbow library
WordDocMatrix	Access to Word-Document-Matrix (not used in CAIMAN)
rainbow.RainbowUtil	General Utilities for using the Rainbow library
svm.SVMClassification	Classification with the SVM technique
svm.SVMProblemDescription	Representation of test catalog in a SVM-suitable format
svm.SVMUtil	General SVM Utilities

TABLE B.2: Overview of the package *de.tum.caiman.classify.**

Table B.3 shows an overview and short descriptions of the classes in the *de.tum.caiman.crawl.** package.

Package.Class	Description
reuters.Preparser	Extracts the Reuters catalog files from the source SGML files
reuters.Preprocessor	Script Adapter for Preparser
researchindex.ResearchIndexCollector	Crawler for the Researchindex catalog
researchindex.ResearchindexParser	Parser for the Researchindex HTML catalog structure pages
researchindex.RIScriptAdapter	Script Adapter for the Researchindex crawler

TABLE B.3: Overview of the package *de.tum.caiman.crawl.**

Table B.4 shows an overview and short descriptions of the classes in the *de.tum.caiman.experiments* package.

Package.Class	Description
Configuration	Represents the parameters set in a configuration file
GnuplotUtil	Output of the results in Gnuplot format
StructuralMatcherContingencyValues	Basic performance assessment for structural match stand-alone test
StructuralMatcherScript	Manager class for structural matching
StructuralMatcherStatisticalVerifier	High-level performance assessment for structural match stand-alone test
StructuralMatcherVerifier	Structural match stand-alone test
StructuralMatcherVerifierFileManager	Basic I/O for structural match stand-alone test
StructuralMatcherVerifierProperties	Structural match stand-alone test configuration
TestParameters	Script adapter class for TestParametersBase
TestParametersBase	The main control flow for the experiments batch processing
TestParametersOfNeighbour	Calculation of the performance results for k-nearest-neighbor classification
TestParametersOfRainbow	Calculation of the performance results for Naive Bayes (Rainbow) classification
TestParametersOfRocchio	Calculation of the performance results for Rocchio classification
TestParametersOfSVM	Calculation of the performance results for SVM classification

TABLE B.4: Overview of the package *de.tum.caiman.experiments*

Table B.5 shows an overview and short descriptions of the classes in the *de.tum.caiman.graph* package.

Package.Class	Description
MatchFilter	Filtering for the structural catalog matching
ModelMapping	Representation of the graph matching problem
StructuralMatcher	Main graph matching functionalities

TABLE B.5: Overview of the package *de.tum.caiman.graph*

Table B.6 shows an overview and short descriptions of the classes in the *de.tum.caiman.match* package.

Package.Class	Description
CategoryCorrelator	Calculation of the initial similarities from the classification results
CategorySimilarities	Representation of category similarities for two catalogs
CategorySimilarityFileManager	Basic I/O for category similarities
CategorySimilarityScript	Stand-alone similarity calculation test

TABLE B.6: Overview of the package *de.tum.caiman.match*

Appendix C

Query Mapping Rules

In Chapter 7 we have introduced our querying approach, which relies on the TRIPLE RDF query, inference and transformation language [Sintek and Decker 2002]. Here, we are going to present some additional technical details about the representation of query mapping rules with TRIPLE.

The *Triple* query engine can process RDF queries [Sintek and Decker 2001]. It is based on the publicly available XSB[1] logic programming library. The goal of *Triple* is to provide a query engine that is capable of querying heterogeneous sources of semi-structured data. Query mapping in our approach is enabled by modelling the semantics of the different catalog data models in *TRIPLE rules*.

Query and rule syntax. The TRIPLE rule and query syntax is loosely based on F-logic [Kifer et al. 1995] syntax. An expression `s[p->o]` corresponds directly to an RDF statement with subject `s`, predicate `p`, and object `o`. Such an expression is either a fact or a rule. To query different information sources, which is not directly possible in RDF and F-logic, so called source expressions are used. A statement `s[p->o]@s1` refers to an RDF statement `(s,p,o)` in the information source `s1`. The information source `s1` is a collection of facts, which is called *model* in TRIPLE terminology. Variable bindings are realized in TRIPLE through the quantifiers *FORALL* and *EXISTS*. For the full specification of the TRIPLE syntax, see [Sintek and Decker 2002]. Rules are also encoded in models.

For the query example in Chapter 7, the catalog data is encoded in a TRIPLE model. To map queries to a certain catalog, we simply add mapping rules to the TRIPLE model of the catalog. As a consequence, of the added rules, new facts are added to the model, when it is queried. Thus, what is essentially done, is not a query mapping, but a mapping of the respective catalog data. More details on TRIPLE rules can be found in [Sintek and Decker 2001].

An example TRIPLE model with rules that allow the mapping of queries to the CIA World Factbook catalog is shown in Figure C.1.

An example TRIPLE model with rules that allow the mapping of queries to the DMOZ Open Directory web catalog is shown in Figure C.1.

[1] See http://xsb.sourceforge.net/

```
@factbook {
FORALL currentCategory, subCategories
   currentCategory[subCategory->subCategories] <-
      EXISTS classInstanceAssociation
         classInstanceAssociation[
            associationMember->subCategories;
            associationMember->currentCategory;
            associationTemplate->psi1:at-class-instance].
FORALL userCategory, communityCategory, relation, object
   communityCategory[relation->object] <-
      userCategory[relation->object] AND
      userCategory[matches->communityCategory].
}
```

FIGURE C.1: TRIPLE model that allows query mapping for the Factbook catalog.

```
@dmoz {
FORALL currentCategory, subCategories
   currentCategory[subCategory->subCategories] <-
      currentCategory[narrow->subCategories].
FORALL userCategory, communityCategory, relation, object
   communityCategory[relation->object] <-
      userCategory[relation->object] AND
      userCategory[matches->communityCategory].
}
```

FIGURE C.2: Part of the TRIPLE model that allows query mapping for the DMOZ catalog.

Appendix D

Sample Document Catalog Data

Open Directory Contents RDF

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE rdf:RDF [<!ENTITY dmoz "http://dmoz.org/">]>
<r:RDF xmlns:r="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:d="http://purl.org/dc/elements/1.0/"
     xmlns="http://directory.mozilla.org/rdf/">
<Topic r:ID="&dmoz;">
  <tag catid="1"/>
  <d:Title>Top</d:Title>
</Topic>
<Topic r:ID="&dmoz;Regional">
  <tag catid="2"/>
  <d:Title>Regional</d:Title>
  <link r:resource="http://www.officialcitysites.org/"/>
</Topic>
<ExternalPage r:about="http://www.officialcitysites.org/">
  <d:Title>Official City Sites</d:Title>
  <d:Description>Directory of official city, county, state and country
sites from Australia, Canada, United Kingdom and United States. Also
includes chamber of commerce and visitor bureau sites.</d:Description>
</ExternalPage>
<Topic r:ID="&dmoz;Regional/Europe">
  <tag catid="3"/>
  <d:Title>Europe</d:Title>
</Topic>
<Topic r:ID="&dmoz;Regional/Europe/Denmark">
  <tag catid="4"/>
  <d:Title>Denmark</d:Title>
  <link r:resource="http://www.cia.gov/cia/publications/factbook/geos/da.html"/>
</Topic>
<ExternalPage r:about="http://www.cia.gov/cia/publications/factbook/geos/da.html">
  <d:Title>The World Factbook - Denmark</d:Title>
  <d:Description>About the Danish geography, people, government,
economy, communications, transportation, military and transnational
issues.</d:Description>
</ExternalPage>
</r:RDF>
```

FIGURE D.1: Open Directory Catalog Contents in RDF

Open Directory Structure RDF

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE rdf:RDF [<!ENTITY dmoz "http://dmoz.org/">]>

<r:RDF xmlns:r="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:d="http://purl.org/dc/elements/1.0/"
     xmlns="http://directory.mozilla.org/rdf/">

<Topic r:ID="&dmoz;">
  <tag catid="1"/>
  <d:Title>Top</d:Title>
  <narrow r:resource="&dmoz;Reference"/>
  <narrow r:resource="&dmoz;Regional"/>
  <narrow r:resource="&dmoz;Science"/>
</Topic>

<Topic r:ID="&dmoz;Regional">
  <tag catid="11"/>
  <d:Title>Regional</d:Title>
  <narrow r:resource="&dmoz;Regional/UK"/>
  <narrow r:resource="&dmoz;Regional/Europe"/>
  <narrow r:resource="&dmoz;Regional/Australia"/>
  <related r:resource="&dmoz;World"/>
</Topic>

<Topic r:ID="&dmoz;Regional/Europe">
  <tag catid="111"/>
  <d:Title>Europe</d:Title>
  <narrow r:resource="&dmoz;Regional/Europe/Czech_Republic"/>
  <narrow r:resource="&dmoz;Regional/Europe/Denmark"/>
  <narrow r:resource="&dmoz;Regional/Europe/Estonia"/>
  <related r:resource="&dmoz;Society/Ethnicity/Scandinavian/Danish"/>
  <related r:resource="&dmoz;World/Dansk"/>
</Topic>

<Topic r:ID="&dmoz;Regional/Europe/Denmark">
  <tag catid="1111"/>
  <d:Title>Denmark</d:Title>
  <narrow r:resource="&dmoz;Regional/Europe/Denmark/Arts_and_Entertainment"/>
  <narrow r:resource="&dmoz;Regional/Europe/Denmark/Business_and_Economy"/>
  <narrow r:resource="&dmoz;Regional/Europe/Denmark/Counties"/>
  <related r:resource="&dmoz;Regional/Europe/Regions/Nordic_Countries"/>
  <related r:resource="&dmoz;Society/Ethnicity/Scandinavian/Danish"/>
  <related r:resource="&dmoz;World/Dansk"/>
</Topic>

</r:RDF>
```

FIGURE D.2: Open Directory Catalog Structure in RDF

CIA World Factbook Topic Map

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<topicMap xmlns="http://www.topicmaps.org/xtm/1.0/"
xmlns:xlink="http://www.w3.org/1999/xlink" id="factbook.hytm">

<topic id="c144">
  <instanceOf>
    <topicRef xlink:href="#country"></topicRef>
  </instanceOf>
  <subjectIdentity>
    <subjectIndicatorRef
      xlink:href="http://psi.ontopia.net/cia/factbook/#DA">
    </subjectIndicatorRef>
  </subjectIdentity>
  <baseName id="id1106">
    <baseNameString>Denmark</baseNameString>
  </baseName>
  <baseName id="id1108">
    <scope>
      <topicRef xlink:href="#csfname"></topicRef>
    </scope>
    <baseNameString>Denmark</baseNameString>
  </baseName>
  <occurrence id="id1110">
    <instanceOf>
      <topicRef xlink:href="#mainpage"></topicRef>
    </instanceOf>
    <resourceRef
      xlink:href="http://www.cia.gov/cia/publications/factbook/geos/da.html">
    </resourceRef>
  </occurrence>
</topic>

<topic id="country">
  <subjectIdentity>
    <subjectIndicatorRef
      xlink:href="http://psi.ontopia.net/cia/factbook/#Country">
    </subjectIndicatorRef>
  </subjectIdentity>
  <baseName id="id1176">
    <baseNameString>Country</baseNameString>
  </baseName>
</topic>

<topic id="csfname">
  <baseName id="id1138">
    <baseNameString>Conventional short form</baseNameString>
  </baseName>
</topic>

</topicMap>
```

FIGURE D.3: CIA World Factbook XTM source

Bibliography

S. Abiteboul, S. Cluet, and T. Milo. Correspondence and translation for heterogeneous data. In *Proceedings of the 6th International Conference on Database Theory (ICDT 1997)*, Jan. 1997.

E. Adar, D. Karger, and L. A. Stein. Haystack: Per-user information environments. In *Proceedings of the 1999 ACM CIKM International Conference on Information and Knowledge Management*, pages 413–422, Nov. 1999.

R. Agrawal and R. Srikant. On integrating catalogs. In *Proceedings of the Tenth International World Wide Web Conference (WWW 10)*, May 2001.

K. Ahmed. Developing a topic map programming model. In *Proceedings of the Knowledge Technologies Conference 2001*, Mar. 2001.

G. Attardi, A. Gull'i, and F. Sebastiani. Automatic web page categorization by link and context analysis. In C. Hutchison and G. Lanzarone, editors, *1st European Symposium on Telematics, Hypermedia and Artificial Intelligence (THAI-99)*, pages 105–119, 1999.

R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Publishing, May 1999.

T. J. Beckman. The current state of knowledge management. In J. Liebowitz, editor, *Knowledge Management Handbook*, pages (1–1)–(1–22). CRC Press, 1999.

S. Bergamaschi and D. Beneventano. Integration of information from multiple sources of textual data. In M. Klusch, editor, *Intelligent Information Agents*, pages 53–77. Springer Verlag, 1999.

T. Berners-Lee. The semantic web. In *Proceedings of XML 2000 Conference*, Dec. 2000. Keynote presentation, available at http://www.w3.org/2000/Talks/1206-xml2k-tbl/Overview.html.

T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):28–37, May 2001.

M. Biezunski, M. Bryan, and S. Newcomb. ISO/IEC 13250: Topic Maps: Information Technology – Document Description and Markup Languages. Dec. 1999. International ISO/IEC Standard, available at http://www.y12.doe.gov/sgml/sc34/document/0129.pdf.

M. Biezunski and S. R. Newcomb. XML Topic Maps Processing Model 1.0. Dec. 2000. Available at http://www.topicmaps.org/xtm/1.0/xtmp1.html.

M. Biezunski and S. R. Newcomb. TOPICMAPS.NET's processing model for XTM 1.0, version 1.0.1. May 2001. Available at http://www.topicmaps.net/pmtm4.htm.

M. Bonifacio, P. Bouquet, and A. Manzardo. A distributed intelligence paradigm for knowledge management. In *Proceedings of 2000 AAAI Spring Symposium: Bringing Knowledge to Business Processes*, pages 69–76. AAAI Press, Mar. 2000.

U. M. Borghoff, M. Koch, M. S. Lacher, J. H. Schlichter, and K. Weißer. Informationsmanagement und Communities - Überblick und Darstellung zweier Projekte der IMC-Gruppe München. *Informatik Forschung und Entwicklung*, 16(2):103–109, Jul. 2001.

U. M. Borghoff and R. Pareschi, editors. *Information Technology for Knowledge Management*. Springer Press, Berlin, 1998.

U. M. Borghoff, R. Pareschi, H. Karch, M. Nöhmeier, and J. Schlichter. Constraint-based information gathering for a network publication system. In *Proc. 2nd Conf. on the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM96)*, pages 45–59, Apr. 1996.

U. M. Borghoff and J. Schlichter. On combining the knowledge of heterogeneous information repositories. *J. of Universal Computer Science*, 2(7):515–532, Jul. 1996.

U. M. Borghoff and J. H. Schlichter. *Computer-Supported Cooperative Work*. Springer Press, 2000.

S. Bowers and L. Delcambre. Representing and transforming model-based information. In *Proceedings of the International Workshop on the Semantic Web (SemWeb), in conjunction with ECDL 2000*, Sep. 2000.

J. S. Brown and E. S. Gray. The people are the company. *Fast Company Magazine*, page 78, Nov. 1995.

H. Bullinger, T. Baumann, N. Fröschle, O. Mack, T. Trunzer, and J. Waltert. *Business Communities*. Galileo Business. Galileo Press, Bonn, Germany, Jan. 2002.

L. Carotenuto, W. Etienne, M. Fontaine, J. Friedman, H. Newberg, M. Muller, M. Simpson, J. Slusher, and K. Stevenson. Communityspace: Towards flexible support for voluntary knowledge communities. In *Proc. Workshop on Workspace Models for Collaboration*, Apr. 1999.

S. Chakrabarti, B. Dom, R. Agrawal, and P. Raghavan. Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies. *The VLDB Journal*, 7(3):163–178, Aug. 1998.

B. Chandrasekaran, J. R. Josephson, and V. R. Benjamins. What are ontologies, and why do we need them ? *IEEE Intelligent Systems*, 14(1):20–26, Jan./Feb. 1999.

C. Chang and C. Lin. Libsvm: a library for Support Vector Machines (Version 2.31). Technical report, National Taiwan University, Taipei, Taiwan, Sep. 2001.

D. Clark. Mad cows, metathesauri and meaning. *IEEE Intelligent Systems*, 14(1):75–77, Jan./Feb. 1999.

C. Clifton, E. Housman, and A. Rosenthal. Experience with a combined approach to attribute-matching across heterogeneous databases. In *Proceedings of the Seventh Conference on Database Semantics (DS-7)*, Oct. 1997.

T. H. Davenport and L. Prusak. *Working knowledge*. Harvard Business School Press, 1998.

A. Doan, P. Domingos, and A. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *SIGMOD 2001 Electronic Proceedings*, Jul. 2001. available at http://www.acm.org/sigmod/sigmod01/eproceedings.

A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web. In *Proc. of the 11th International World Wide Web Conference (WWW2002)*, May 2002.

P. Dourish, K. Edwards, A. LaMarca, and M. Salisbury. Presto: An experimental architecture for fluid interactive document spaces. *ACM Transactions on Computer-Human Interaction*, 6(2):131–161, 1999a.

P. Dourish, J. Lamping, and T. Rodden. Building bridges: Customisation and mutual intelligibility in shared category management. In *Proceedings of the ACM Conference on Supporting Group Work (GROUP '99)*, Nov. 1999b.

S. T. Dumais, J. Platt, D. Hecherman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the 7th ACM International Conference on Information and Knowledge Management (CIKM-98)*, pages 148–155, 1998.

D. Etzold. Parameter-optimization for machine learning techniques for automated text classification. Technical report, Technische Universität München, Department of Informatics, Mar. 2003.

D. Fensel, A. Witt, J. Angele, S. Decker, M. Erdmann, H. Schnurr, S. Staab, and R. Studer. On2broker in a nutshell. In *Proceedings of the 8th World Wide Web Conference*, May 1999.

L. N. Foner. Yenta - a multi-agent, referral-based matchmaking system. In *Proc. 1st International Conference on Autonomous Agents*, Feb. 1997.

L. N. Foner. *Political Artifacts and Personal Privacy: The Yenta Multi-Agent Distributed Matchmaking System*. PhD thesis, Massachussets Institute of Technology, Jun. 1999.

J. P. Forgas. *Interpersonal behaviour*. Pergamon Press, 1985.

H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. Integrating and accessing heterogeneous information sources in TSIMMIS. In *Proc. AAAI Symposium on Information Gathering*, pages 61–64, Mar. 1995.

N. Glance, D. Arregui, and M. Dardenne. Knowledge Pump: Community-centered collaborative filtering. In *Proc. Fifth DELOS Workshop: Filtering and Collaborative Filtering*, ERCIM Workshop Proceedings 98-W001, Nov. 1997.

N. Glance, D. Arregui, and M. Dardenne. Knowledge Pump: Supporting the flow and use of knowledge in networked organizations. In U. Borghoff and R. Pareschi, editors, *Information Technology for Knowledge Management*. Springer Press, Berlin, 1998.

C. Goller, J. Löning, T. Will, and W. Wolff. Automatic document classification: A thorough evaluation of various methods. In *Proc. Internationales Symposium für Informationswissenschaft (ISI 2000)*, Nov. 2000.

G. Groh. Applying text classification methods to the mapping of simple extensional ontologies for community information management, Mar. 2001. Master's Thesis, Technische Universität München.

T. R. Gruber. Ontolingua: A mechanism to support portable ontologies. Technical report, Stanford University, 1992.

T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5 (2):199–220, 1993.

J. Grudin. Groupware and social dynamics: eight challenges for developers. *Communications of the ACM*, 37(1):93–105, Jan. 1994.

W. Hacker, editor. *Kognitive und motivationale Aspekte der Handlung*, volume 22 of *International Congress of Psychology*. Huber Press, 1983.

A. Halevy. Answering queries using views: A survey. *VLDB Journal*, 10(4):270–294, 2001.

J. Hammer, J. M. Hugh, and H. Garcia-Molina. Semistructured data. In *First East-European Workshop on Advances in Databases and Information Systems - ADBIS 97*, Sep. 1997.

E. Heinen and B. Dietel, editors. *Industriebetriebslehre*, volume 9. Gabler Press, 1991.

G. A. Hillery. Definitions of community: Areas of agreement. *Rural Sociology*, 20:111–123, 1955.

M. N. Huhns and M. P. Singh. Ontologies for agents. *IEEE Internet Computing*, 1(6):81–83, Nov./Dec. 1997.

M. N. Huhns and L. M. Stephens. Personal ontologies. *IEEE Internet Computing*, 3(5):85–87, Sep. 1999.

D. Huynh, D. Karger, and D. Quan. Haystack: A platform for creating, organizing and visualizing information using RDF. In *Proc. 12th International World Wide Web Conference*, May 2003.

T. Ishida. *Community Computing*. John Wiley and Sons, 1998a.

T. Ishida, editor. *Community Computing and Support Systems*, volume 1519 of *Lecture Notes in Computer Science*. Springer Press, Berlin, 1998b.

T. Ishida. Towards computation over communities. In T. Ishida, editor, *Community Computing and Support Systems, Social Interaction in Networked Communities*, volume 1519 of *Lecture Notes in Computer Science*, pages 1–10. Springer Press, Berlin, 1998c.

R. J.Bayardo, W. Bohrer, R. Brice, A. Cichocki, J. Fowler, A. Helal, V. Kashyap, T. Ksiezyk, G. Martin, M. Nodine, M. Rashid, M. Rusinkiewicz, R. Shea, C. Unnikrishnan, A. Unruh, and D. Woelk. Infosleuth: agent-based semantic integration of information in open and dynamic environments. In *Proc. ACM SIGMOD International Conference on Management of Data*, pages 195–206, May 1997.

T. Joachims. A probabilistic analysis of the rocchio algorithm with TFIDF for text categorization. In *Proc. 14th International Conference on Machine Learning (ICML97)*, pages 143–151, 1997.

T. Joachims. Text categorization with Support Vector Machines: learning with many relevant features. In *Proc. 10th European Conference on Machine Learning (ECML 98)*, 1998.

H. Kautz, B. Selman, and M. Shah. Referral web: combining social networks and collaborative filtering. *Communications of the ACM*, 40(3):63–65, Mar. 1997.

M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented and frame-based languages. *Journal of the ACM*, 42(4):741–843, 1995.

J. H. Koch. *Unterstützung der Formierung und Analyse von virtuellen Communities*. PhD thesis, Dept. of Computer Science, Technische Universität München, Jul. 2002. ISBN 3-631-50288-5.

M. Koch and M. S. Lacher. Integrating community services - a common infrastructure proposal. In *Proc. 4th Knowledge-Based Intelligent Engineering Systems and Allied Technologies*, pages 56–59, Aug. 2000.

M. Koch, M. S. Lacher, and W. Wörndl. The CommunityItemsTool - interoperable community support in practice. In *Proc. WETICE-2001 - Workshop on Web-based Infrastructures and Coordination Architectures for Collaborative Entreprises*, Jun. 2001.

D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In D. H. Fisher, editor, *14th International Conference on Machine Learning (ICML-97)*, pages 170–178. Morgan Kaufman Publishers, Jul. 1997.

E. Kolodizki. Addition of graph matching techniques to an existing catalog mediation infrastructure. Technical report, Technische Universität München, Informatics Dept., Dec. 2002.

M. S. Lacher and G. Groh. Facilitating the exchange of explicit knowledge through ontology mappings. In *Proc. 14th International FLAIRS conference*. AAAI Press, May 2001.

M. S. Lacher and M. Koch. An agent-based knowledge management framework. In *Proc. AAAI Spring Symposium 2000*, pages 145–147, Mar. 2000.

O. Lassila and R. R. Swick. Resource Description Framework (RDF) Model and Syntax Specification, Feb. 1999. W3C Recommendation 22, Feb. 1999.

J. Lave. Situated learning in communities of practice. In L. B. Resnick, J. Levine, and S. Teasley, editors, *Perspectives on socially shared cognition*, pages 63–82. American Psychological Association, 1991.

D. Lenat. Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):32–38, Nov. 1995.

D. D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In C. Nedellec and C. Rouveirol, editors, *10th European Conference on Machine Learning (ECML-98)*, volume 1398 of *Lecture Notes in Computer Science*, pages 4–15. Springer, Apr. 1998.

W. Li and C. Clifton. SEMINT: A tool for identifying attribute correspondences in heterogeneous databases using neural networks. *Data and Knowledge Engineering*, 33(1):49–84, Apr. 2000.

J. Madhavan, P. A. Bernstein, P. Domingos, and A. Y. Halevy. Representing and reasoning about mappings between domain models. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence and Fourteenth Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI 2002)*, Aug. 2002.

J. Madhavan, P. A. Bernstein, and E. Rahm. Generic schema matching with cupid. In P. M. G. Apers, P. Atzeni, S. Ceri, S. Paraboschi, K. Ramamohanarao, and R. T. Snodgrass, editors, *27th International Conference on Very Large Data Bases (VLDB 2001)*. Morgan Kaufman, Sep. 2001.

R. K. Maier and O. W. Klosa. Wissensmanagementsysteme. Technical report, Universität Regensburg, Department for Management Information Systems, Feb. 2000.

C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

A. McCallum. A comparison of event models for Naive Bayes text classification. In *Proc. AAAI-98 Workshop on Learning for Text Categorization*. AAAI Press, 1998.

S. Melnik and S. Decker. A layered approach to information modeling and interoperability on the web. In *Proc. ECDL 2000 Workshop on the Semantic Web*, Sep. 2000.

S. Melnik, H. Garcia-Molina, and A. Paepcke. A mediation infrastructure for digital libraries. In *Proceedings of the Fifth ACM Conference on Digital Libraries*, pages 123–132, Jun. 2000.

S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity Flooding: A versatile graph matching algorithm and its application to schema matching. In *Proc. 18th International Conference on Data Engineering (ICDE)*, Mar. 2002.

P. Mitra and G. Wiederhold. Resolving terminological heterogeneity in ontologies. In *Proc. of Workshop on Ontologies and Semantic Interoperability at the 15th European Conference on Artificial Intelligence (ECAI 2002)*, Jul. 2002.

P. Mitra, G. Wiederhold, and M. Kersten. A graph-oriented model for articulation of ontology interdependencies. In *Proc. of VII. Conference on Extending Database Technology (EDBT 2000)*, pages 86–100, Mar. 2000.

G. D. Moore. RDF and Topic Maps - An exercise in convergence. In *Proceedings of XML Europe 2001*, May 2001.

E. D. Mynatt, A. Adler, M. Ito, and V. L. ODay. Design for network communities. In *Proc. ACM SIGCHI Conf. on Human Factors in Computing Systems*, pages 210–217, 1997.

M. Nöhmeier. *Einsatz von Agententechnologie zur Unterstützung von Informationsaustausch in globalen Informationsräumen*. PhD thesis, Dept. of Computer Science, Technische Universität München, 1998.

I. Nonaka and H. Takeuchi. *The Knowledge-Creating Company*. Oxford University Press, 1995.

N. F. Noy and C. D. Hafner. The state of the art in ontology design. *AI Magazine*, (Fall 1997):53–74, 1997.

N. F. Noy and D. L. McGuiness. Ontology development 101: A guide to creating your first ontology. Technical report, SMI, Stanford University, 2001.

N. F. Noy and M. A. Musen. SMART: Automated support for ontology merging and alignment. In *Proc. of Twelfth Workshop on Knowledge Acquisition, Modeling and Management (KAW 99)*, Oct. 1999.

N. F. Noy and M. A. Musen. PROMPT: Algorithm and tool for automated ontology merging and alignment. In *Proc. of 17th National Conference on Artificial Intelligence (AAAI-2000)*, pages 450–455, 2000.

L. Palopoli, G. Terracina, and D. Ursino. The system DIKE: Towards the semi-automatic synthesis of cooperative information systems and data warehouses. In *Proc. of ADBIS-DASFAA Symposium 2000*, pages 108–117, Sep. 2000.

Y. Papakonstantinou, H. Garcia-Molina, and J. Widom. Object exchange across heterogeneous information sources. In P. S. Yu and A. L. P. Chen, editors, *Proceedings of the Eleventh International Conference on Data Engineering*, pages 251–260. IEEE Computer Society, Mar. 1995.

S. Pepper and G. D. Moore, editors. *XML Topic Maps (XTM) 1.0*. Mar. 2001. TOPICMAPS.ORG Specification, available at http://www.topicmaps.org/xtm/1.0/.

M. Polanyi. *Personal Knowledge*. University of Chicago Press, 1974.

A. Pretschner and S. Gauch. Ontology based personalized search. In *Proc. 11th IEEE Intl. Conf. on Tools with Artificial Intelligence*, pages 391–398, Nov. 1999.

G. Probst, S. Raub, and K. Romhardt. *Wissen managen (Managing knowledge)*. Gabler Press, FAZ Press, 1997.

S. Raghavan and H. Garcia-Molina. Crawling the hidden web. In *Proc. of the 27th Conference on Very Large Databases (VLDB 2001)*, pages 129–138, Sep. 2001a.

S. Raghavan and H. Garcia-Molina. Integrating diverse information management systems: A brief survey. *IEEE Data Engineering Bulletin*, 24(4):44–52, Dec. 2001b.

E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 10(4):334–350, Nov. 2001.

G. Reinmann-Rothmeier and H. Mandl. *Individuelles Wissensmanagement*. Hans Huber Press, 2000.

H. Rheingold. *The Virtual Community*. Addison-Wesley, 1993.

J. Rucker and M. J. Polanco. Siteseer: personalized navigation for the web. *Communications of the ACM*, 40(3):73–76, Mar. 1997.

J. Schlichter, M. Koch, and C. Xu. Awareness - the common link between groupware and community support systems. In T. Ishida, editor, *Community Computing and Support Systems*, Lecture Notes in Computer Science 1519, pages 77–93. Springer Verlag, Jun. 1998.

M. P. Schmidt. *Knowledge Communities - Mit virtuellen Wissensmärkten Wissen in Unternehmen effektiv nutzen*. Addison-Wesley, München, 2000.

D. Schuler. Community networks: Building a new participatory medium. *Communications of the ACM*, 37(1):38–51, Jan. 1994.

D. Schuler. Creating public space in cyberspace - the rise of the new community networks. *Internet World*, Dec. 1995.

F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys (CSUR)*, 34(1):p. 1–47, Mar. 2002.

M. Sintek and S. Decker. TRIPLE - an RDF query, inference, and transformation language. In *Proc. of Workshop Deductive Databases and Knowledge Management (DDLP 2001), co-located with Internation Conference on Applications of Prolog (INAP 2001)*, 2001.

M. Sintek and S. Decker. TRIPLE- A query, inference, and transformation language for the semantic web. In *Proc. 1st International Semantic Web Conference*, page 364, Jun. 2002.

J. F. Sowa. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks Cole Publishing Co., Pacific Grove, CA, 2000.

S. Staab, J. Angele, S. Decker, M. Erdmann, A. Hotho, A. Mädche, H. Schnurr, R. Studer, and Y. Sure. Semantic community web portals. In *Proc. of the 9th Intl. WWW Conference*, May 2000a.

S. Staab, J. Angele, S. Decker, M. Erdmann, A. Hotho, A. Mädche, H. Schnurr, R. Studer, and Y. Sure. AI for the web - ontology-based community web portals. In *Proc. 17th National Conf. on Artificial Intelligence and 12th Innovative Applications of AI Conf.*, pages 1034–1039. AAAI Press / MIT Press, Jul. 2000b.

S. Staab, M. Erdmann, A. Maedche, and S. Decker. An extensible approach for modeling ontologies in RDF(S). In *Proceedings of ECDL 2000 Workshop on the Semantic Web*, pages 11–22, 2000c.

D. Steiner. Die fipa-Initiative für Agenten-Standardisierung. *it und ti - Informationstechnik und Technische Informatik*, 40:46–49, Apr. 1998.

G. Stumme and A. Maedche. FCA-MERGE: Bottom-up merging of ontologies. In *Proc. Seventeenth International Joint Conference on Artificial Intelligence (IJCAI 2001)*, pages 225–234. Morgan Kaufman, Aug. 2001.

D. Suciu. An overview of semi-structured data. *SIGACTN: SIGACT News*, 29:28–38, 1998.

H. Takeda, T. Matsuzuka, and Y. Taniguchi. Discovery of shared topics networks among people - a simple approach to find community knowledge from www bookmarks. In M. Riichiro and S. John, editors, *6th Pacific Rim International Conference on Artificial Intelligence (PRICAI 2000)*, Lecture Notes in Computer Science 1886, pages 668–678. Springer, Sep. 2000.

A. S. Tanenbaum. *Computer Networks*. Number 2. Prentice-Hall, Englewood Cliffs, NJ, 1987.

C. J. van Rijsbergen. *Information Retrieval*. Butterworths, 1979.

H. Wache, T. Vögele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner. Ontology-based integration of information - a survey of existing approaches. In *Proc. of IJCAI-01 Workshop on Ontologies and Information Sharing*, pages 108–117, 2001.

G. Weiß, editor. *Multiagent Systems - A Modern Approach to Distributed Artificical Intelligence*. MIT Press, 1999.

G. Weiß. Agentenorientiertes software engineering. *Informatik Spektrum*, 24(2):98–101, Apr. 2001.

B. Wellman. A computer network is a social network. *SIGGROUP Bulletin*, 19(3):41–45, Dec. 1998.

C. A. Welty. The ontological nature of subject taxonomies. In *Proc. 1998 International Conference on Formal Ontology in Information Systems (FOIS98)*. IOS Press, Jun. 1998.

E. Wenger. *Communities of Practice: Learning, Meaning and Identity*. Cambridge University Press, 1998.

E. C. Wenger and W. M. Snyder. Communities of practice: The organizational frontier. *Harvard Business Review*, pages 139–145, Jan.-Feb. 2000.

R. Whiting. Myths & realities - what is behind one of the most-misunderstood IT strategies. *Information Week online*, Nov. 1999.

G. Wiederhold. Mediators in the architecture of future information systems. *IEEE Computer*, 25(3): 38–49, 1992. reprinted in 'Readings in Agents, M. N. Huhns, M. P. Singh (eds.), pages. 185-196, 1997.

G. Wiederhold. Mediation in information systems. *ACM Computing Surveys*, 27(2):265–267, Jun. 1995.

Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '99)*, pages 42–49, Aug. 1999.

Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *ICML-97, 14th International Conference on Machine Learning*, pages 412–420. Morgan Kaufman Publishers, San Francisco, USA, Jul 1997.

Y. Yang, S. Slattery, and R. Ghani. A study of approaches to hypertext categorization. *Journal of Intelligent Information Systems*, 18(2):219–241, Mar. 2002.