

Downloaded from UvA-DARE, the institutional repository of the University of Amsterdam (UvA)
<http://hdl.handle.net/11245/2.74690>

File ID	uvapub:74690
Filename	Thesis
Version	unknown

SOURCE (OR PART OF THE FOLLOWING SOURCE):

Type	PhD thesis
Title	End-user support for access to heterogeneous linked data
Author(s)	M. Hildebrand
Faculty	FNWI: Informatics Institute (II)
Year	2010

FULL BIBLIOGRAPHIC DETAILS:

<http://hdl.handle.net/11245/1.318913>

Copyright

It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content licence (like Creative Commons).

The openness of the Web allows data from different sources to be combined, enabling information access in ways unforeseen by the original providers. The rise of linked data adds a new dimension to this information revolution. Using Semantic Web representation languages, structured information becomes more easily accessible by machines, enabling emergent technologies to reuse, integrate and process information. The foundations for the Web of data are now in place and institutions are starting to share linked data (e.g. the British government and the BBC).

In his thesis, Michiel Hildebrand investigates access to heterogeneous linked data from an end-user perspective. Based on case studies in the cultural heritage domain he explores how domain experts can benefit from semantically-rich linked data.

With the development and evaluation of prototype systems he demonstrates how a number of specific tasks can be supported. Hildebrand shows that generic solutions can be successfully applied to linked data, by using the semantic relations in the data combined with careful configuration of the search algorithms and interface components,

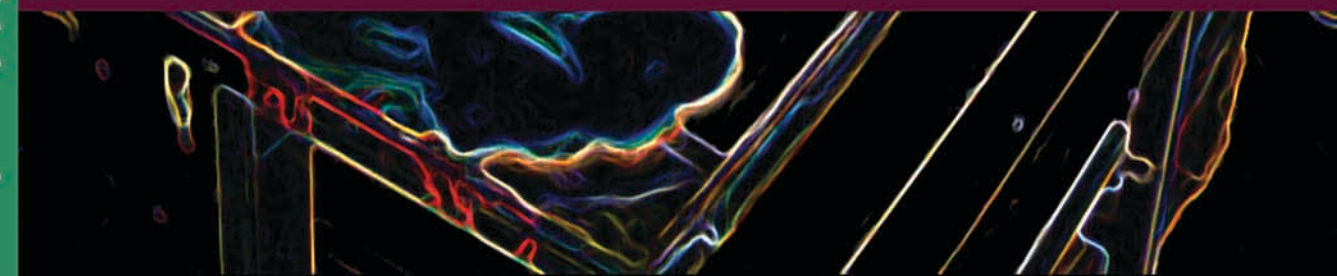
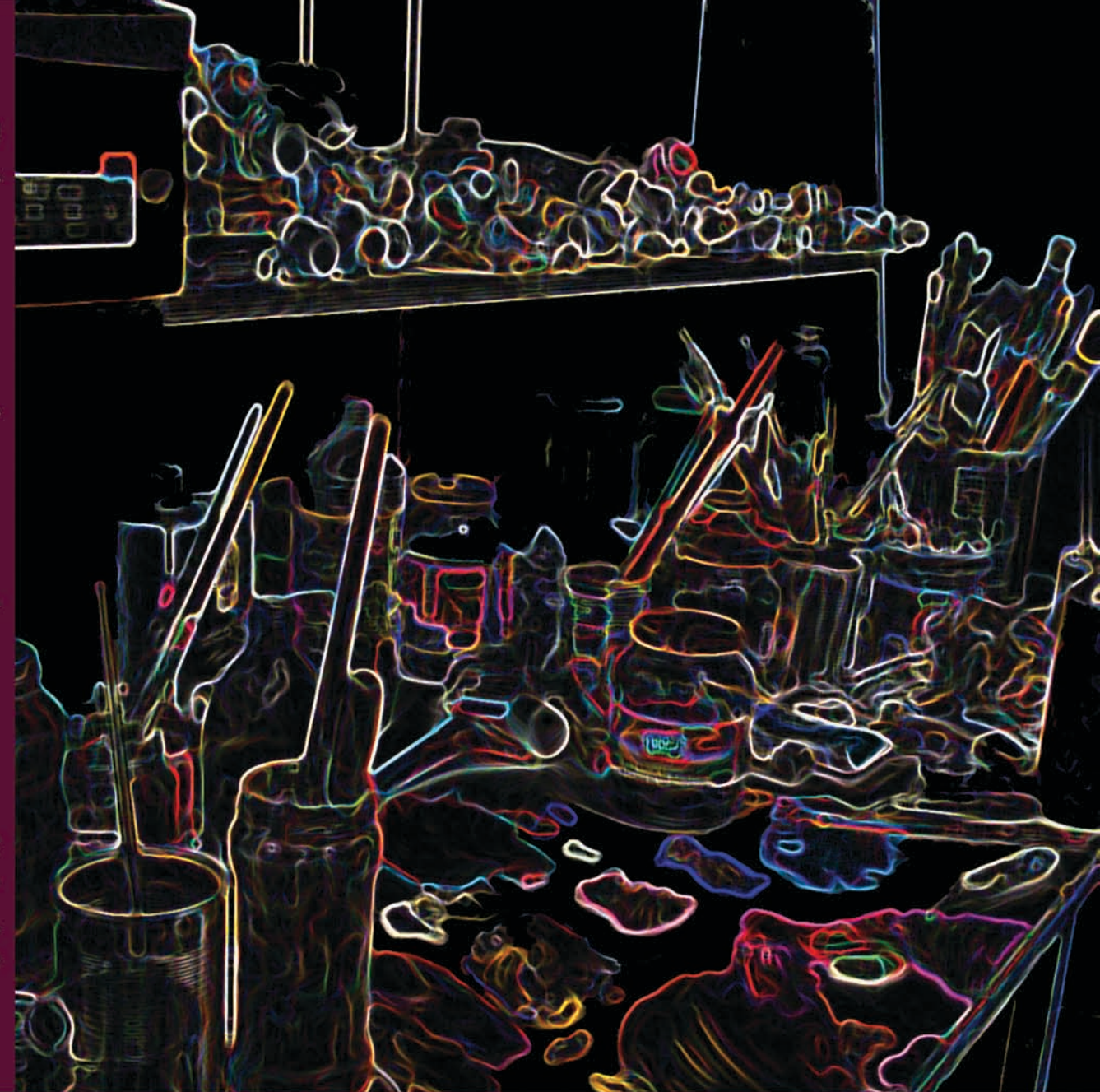
"Although much work lies ahead, we believe that with the right web services in place, and configurable interface components at our disposal, end-user access to heterogeneous linked data has become within reach. Where on Web 2.0, a programmer was required to create a mash-up, integrated access on Web 3.0 will follow from the relations in the data itself."

End-user support for access to heterogeneous linked data

Michiel Hildebrand

Michiel Hildebrand

End-user support for access to heterogeneous linked data



END-USER SUPPORT FOR ACCESS TO HETEROGENEOUS LINKED DATA

End-user support for access to heterogeneous linked data

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. D.C. van den Boom
ten overstaan van een door het college voor promoties ingestelde
commissie, in het openbaar te verdedigen in de Agnietenkapel
op dinsdag 20 april 2010, te 12.00 uur

door

Michiel Hildebrand

geboren te Soest

Promotoren: prof. dr. L. Hardman
prof. dr. A.Th. Schreiber
Copromotor: dr. J.R. van Ossenbruggen
Overige leden: prof. dr. J.A. Hendler
prof. dr. ir. G.J. Houben
prof. dr. M.L. Kersten
prof. dr. M. de Rijke
prof. dr. B.J. Wielinga

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

SIKS Dissertation Series No. 2010-22



The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.



This research was supported by the MultimediaN project funded through the BSIK programme of the Dutch Government and by the European Commission under contract FP6-027026.

Printed by Wörhman Print Service, Zutphen.

Cover design based on artworks and work shop of Daan den Houter.

Contents

Preface	v
1 Introduction	1
1.1 The Semantic Web as a Web of data	1
1.2 Project context: A Web of culture data	2
1.3 Research questions	4
1.4 Approach	5
1.5 Contributions	7
1.6 Structure of the thesis	8
1.7 Publications	8
2 Related work	11
2.1 Introduction	11
2.2 Basic Search Terminology	12
2.3 Analysis of semantic search	13
2.4 Discussion	21
3 Case study I: Subject matter annotation	23
3.1 Introduction	23
3.2 Current annotation practices at the Rijksmuseum	24
3.3 Related work	27
3.4 User study	29
3.5 Requirements analysis	29
3.6 Refinement of requirements and design decisions	34
3.7 Evaluation	44
3.8 Conclusions and future work	50

4 Case study II: Faceted browsing	53
4.1 Introduction	53
4.2 Example scenario	55
4.3 Requirements for multi-type facet browsing	56
4.4 Functional design for multi-type facet browsing	59
4.5 Discussion and Related Work	64
4.6 Conclusion and Future Work	66
5 Case study III: Semantic search	67
5.1 Introduction	67
5.2 Study setup	69
5.3 Data set	70
5.4 Collecting the query test set	72
5.5 Analysis of relations found	75
5.6 Qualitative evaluation of path types	83
5.7 Exploring path type configurations	87
5.8 Implications for design	91
5.9 Conclusion	95
6 ClioPatria: Semantic search and annotation framework	97
6.1 Introduction	97
6.2 Materials and use cases	98
6.3 Required methods and components	101
6.4 The ClioPatria search and annotation toolkit	106
6.5 Discussion	111
7 Configuring Semantic Web interfaces by data mapping	115
7.1 Introduction	115
7.2 Related Work	116
7.3 Combining the Yahoo! User Interface Library with the ClioPatria Interface Model	117
7.4 Configuring interface widgets: a mapping task	124
7.5 Conclusion and Future Work	129
8 Conclusions	133
8.1 The research questions revisited	134
8.2 Discussion and future research	141
8.3 Looking ahead	144
Bibliography	145
Summary	153

CONTENTS	iii
Samenvatting	159
SIKS Dissertatiereeks	164

Preface

Trial and error is likely to be the learning method I have become most accustomed to. Some trials in my life turned out to be rather successful, while others led to regrettable errors. I am very grateful to the people that allowed me to explore and find my way. My parents, Wil and Mariette, have been doing this from the day I was born. After several trials I still can't find the words to express my gratitude for the freedom by which they let me go and the unconditional love by which they keep me close. My other family members, in particular my sister Nanouk and my grandmother Jans, are the perfect complement to this warm and playful environment.

During my PhD research, others took the effort and patience to guide me in my explorations. I had the luxury that eventually three (co-)promoters were actively involved with my supervision. It is to Lynda that I can now call my self a researcher. Whenever I walked into her office (or her house) she took the time to understand what I was trying to explain and then create the conditions under which I could collect the required insights myself. Jacco probably walked as many times into my office, as I walked into his. Our transparent relationship made the day to day work a joy and a difficult moment never last too long. At the same time, he managed to put me to work when I was slacking and continuously push me steps forward. In the background, Guus was giving tactical directions, providing the right contacts and eventually helping me with the nitty gritty details of thesis writing.

On the work floor, Alia was my companion in exploring our scientific aspirations: sharpening our discussion skills and training our patience. Off the work floor (beaches, bars and discos of PhD events), she also makes a good companion. I still do not know if it was me or her that always made us go home so late. The other INS2'ers at CWI also turned out to be professionally as well as socially an acquisition to my ongoing explorations. As can be seen in the bibliography, a certain "Wielemaker, J" had quite an impact on this thesis as well. The computational tools provided by Jan enabled me to rapidly explore ideas, his insights helped me

to spot the ideas not to explore, and his instant responses helped me greatly with the errors I made. The members of the MultimediaN E-Culture project created a research environment that gave my explorations a head start. I owe it to Geertje Jacobs and the employes of the print room from the Rijksmuseum Amsterdam, that I could apply and test these explorations in a realistic and inspiring setting.

The long days and weekends with my text editor had an inverse increased effect on my presence with friends. Luckily, Daniel is worth a thousand friends and a single weekend *cookin' with jazz* is intense enough for a year. Finally, *nha kretxeu* Andra makes me the man that I want to be, and confronts me with myself when I am not. Her love and that of her family makes me realise what is important. May we keep exploring the endless flavours of the world together.

Michiel Hildebrand

March 2010

Chapter 1

Introduction

1.1 The Semantic Web as a Web of data

The openness of the World Wide Web is changing the expectations on information access. Internet users expect information to be available at anytime, with the potential to contribute and link everything to everything else. As a consequence, organisations are starting to open up their previously isolated data silos and services, enabling users to combine data from multiple sources and apply the services of one organisation to the data of another.

These expectations on information access are underpinned by several technological developments. The move towards Web 2.0 has popularised sharing of content, using lightweight data structures such as RSS and JSON, sharing services over public APIs, and sharing interface components, such as JavaScript widgets. These front-end technologies have enabled the construction of mash-ups, web applications that combine data and services from different providers, such that information can be accessed in unforeseen ways.

At the back-end, Semantic Web technologies promise to simplify reuse of data from multiple sources. In a database or XML document, the intended meaning of the data, the semantics, is captured implicitly in the database or XML schema. A mash-up has to be aware of these schemata, which, typically, requires a developer to provide precise instructions on how to use and integrate the data. Semantic Web representation languages, such as RDF, OWL and SKOS, allow aspects of these semantics to be made explicit in a machine-accessible way, enabling intelligent technologies to infer how data should be used and integrated.

In the development of the Semantic Web, we can distinguish the “semantics” from the “web” aspect. In the early years, the focus was on the formal “semantics” suited for knowledge representation on the Web. As a result, the World Wide Web Consortium (W3C) standardised RDF (W3C 1999), and the more expressive OWL (W3C 2004), as languages for representing information on the Web. A set of statements, or triples, defined in these languages, which express different levels of

interoperable semantics, is called an RDF graph. The community provides off-the-shelf “triple stores” to store such RDF graphs and provide basic reasoning facilities over them.

In more recent years, the “web” aspect has gained momentum. Inspired by the Linking Open Data Project¹, a growing number of data collections is being published online and linked together as graph structures distributed over the Web. The result is heterogeneous linked data that to date, for example, contains the structured data from Wikipedia, geographical locations from Geonames, books from Amazon, publications from DBLP, subject headings from the Library of Congress and music related information from MusicBrainz. Furthermore, for some content providers, such as the BBC, publishing linked data has become part of their daily routine (Kobilarov et al. 2009). Although the Web of Data contains some formal ontologies, the bulk of data is formed by domain specific relations and controlled vocabularies. The Simple Knowledge Organisation System (SKOS) (W3C 2005) can be used to publish these vocabularies in an interoperable manner.

The amount of linked data has now reached a state that we can start asking how end users will access this information. Web 2.0 front-end technologies have already resulted in intuitive interfaces for end users to access information from a single data source or, when a mash-up has been created, to a combination of data sources and services. A general problem is how these front-end technologies can be applied to heterogeneous linked data, or the other way around, how the interoperability at the back-end can be exploited by interfaces at the front-end. The central problem is: *how can we support end users with access to heterogeneous linked data.*

An important consideration in information access is the trade-off between task-specific or generic support and homogeneous or heterogeneous data. Task-specific support, typically, requires some control over the data, which limits information access to data with a specific schema. On the other hand, a generic approach, such as a direct visualisation of the data structure, can give access to any collection of heterogeneous data, but provides limited support to the user. In this thesis, we investigate which models and tools can provide task-specific support to access heterogeneous linked data. In particular, we investigate how the semantics in linked data can be used for this purpose.

1.2 Project context: A Web of culture data

Within this thesis we use cultural heritage as an application domain. This domain is well suited for our research. First, the data in this domain is heterogeneous, as it describes many different types of objects and the descriptions vary per institution. Second, the domain contains semantically-rich background knowledge, as it has a

¹<http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>

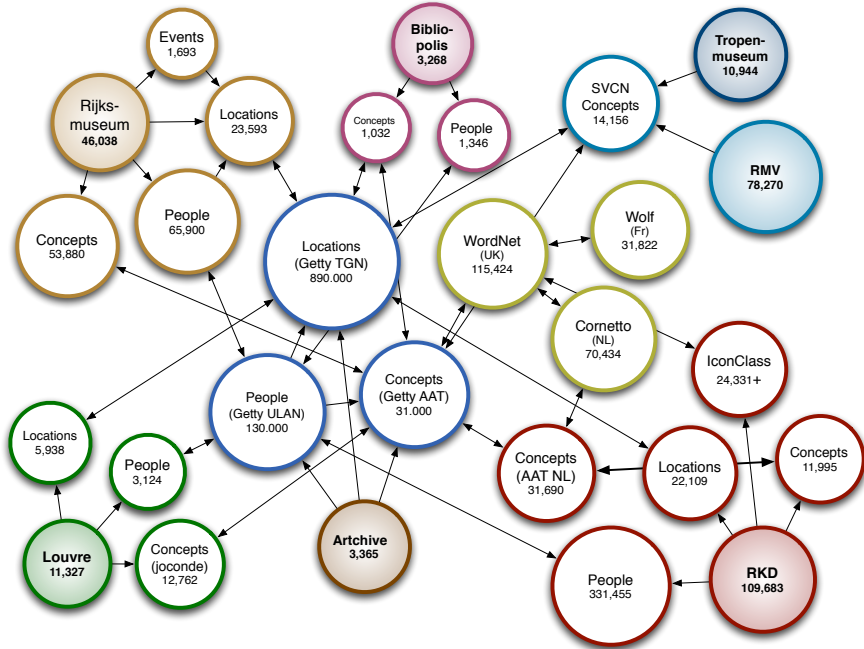


Figure 1.1: Datacloud of the linked cultural heritage data created in the MultimediaN E-Culture project.

long tradition of maintaining rich vocabularies and carefully describing objects with terms from these structured vocabularies. Third, there are users, from cultural heritage experts to interested museum visitors, with information needs that require data from different sources.

This research is performed in the context of the MultimediaN E-Culture project (Schreiber et al. 2008). This project has converted collections, their corresponding thesauri and additional background knowledge to RDF, OWL and SKOS, and captured the semantic relationships among them. Figure 1.1 shows the data cloud from the resulting web of culture data. Typical for this culture web is the large number of controlled vocabularies. The circles in the figure with a coloured fill represent the *collections* of works described with annotations from many different vocabularies. In addition, the project has included different external vocabularies and aligned them with other sources to create a highly interconnected web of data.

The project also created a flexible software architecture, where a triple store is tightly integrated with reasoning and Web server facilities (Wielemaker 2009a). This architecture is used as the platform for experimentation.

1.3 Research questions

The general problem investigated in this thesis is:

How can we support end-users with access to heterogeneous and semantically-rich linked data?

An important characteristic of *linked data* is its representation in a graph structure, where the nodes and edges (or resources and relations) are explicitly typed. In *heterogeneous* linked data this graph structure contains different types of resources and relations. On the Web, new data — containing different types of resources and relations — can be added at any time. We, therefore, assume that there is limited control over the the available types in the data. We focus our research on access to objects that are described with terms from multiple, structured and interlinked vocabularies. We call these heterogeneous and interlinked vocabularies *semantically rich* background knowledge. Our hypothesis is that the structures in this background knowledge can be used to support end-user access to heterogeneous data.

We consider information *access* as a process where the user formulates an information need, in the form of a query, and a computer system responds by selecting and presenting a number of objects (TRECVID organizers 2007). This requires an interface to submit a query and an algorithm to select objects. We refer to this combination as the search functionality. It remains unclear how search functionality can exploit semantically-rich graph structures. The first research question, therefore, is:

RQ 1. *How can semantically-rich graph structures be used in search functionality to support the user in finding objects in heterogeneous linked data?*

Information access also requires an algorithm to organise the selected results and an interface to present these to the user. We refer to this combination as the presentation method. The second research question focusses on the presentation of results found in linked data:

RQ 2. *How can semantically-rich graph structures be used in the presentation of the results found in heterogeneous linked data?*

The effectiveness of the applications to access heterogeneous linked data is affected by the quality and the modelling of the data, the behaviour of the algorithms and the design of the user interface. These three aspects influence each other

and are difficult to isolate in a realistic experimental setting. The methods to evaluate access to semantically-rich and heterogeneous linked data are, therefore, an intrinsic part of this research.

1.4 Approach

The two research questions mentioned above are first addressed in a literature study. In a survey of search and browsing applications for Semantic Web data, different types of search functionality and presentation methods are analysed. While the applications provide a large variety of different solutions, semantic search and browsing is still an open area. Given a specific domain and search task it is difficult to determine how the semantics in the data should be used to benefit the end user.

This thesis takes a first step in formulating, for a specific domain and a number of tasks, the requirements to support end-user access to semantically-rich linked data. As already motivated in the project context (Section 1.2), cultural heritage is chosen as an application domain. To minimise the effect of the data quality and the modelling decisions we focus on support for expert users, who are already familiar with interaction in a knowledge-rich domain.

The tasks are chosen to cover different dimensions of the search problem encountered in the survey. In the search functionality (RQ 1), we distinguish controlled input for the formulation of structured queries and uncontrolled text-based search to formulate more open-ended requests. In the presentation (RQ 2), we distinguish the search results and the navigation paths used for further interaction. From a data perspective we distinguish artwork collections (foreground) and vocabularies (background). We investigate different combinations of the dimensions in three case studies: annotation, faceted browsing and semantic search.

Annotation: text-based search for vocabularies The first case study investigates the research questions in the context of artwork annotation. In professional annotation the task of the user is to find vocabulary concepts to describe an artwork. The study addresses the first research question by exploring the application of text-based search functionality to multiple vocabularies. It addresses the second research question by exploring the visualisation and organisation of the search results (vocabulary concepts).

The starting point is an existing collection management system where each annotation field provides access to only a single vocabulary. In a user study with professional cataloguers we explore how experts can be supported in effectively finding concepts from multiple vocabularies. The initial requirements are formulated based on an analysis of the current situation. In a process of iterative prototyping, the requirements are refined and different solutions are explored. The solutions in the final prototype are qualitatively evaluated with feedback from end-users.

Faceted browsing: structured query formulation for collections The second case study investigates the research questions in the context of faceted browsing, a popular interface paradigm to explore (artwork) collections. It addresses the first research question by exploring an extension of traditional faceted browsing to support the formulation of structured queries for linked data. It addresses the second research question by exploring methods to organise the large number of navigation paths.

The starting point is traditional faceted browsing for a homogeneous collection with a single schema. Based on a use case, the requirements for faceted browsing on heterogeneous Semantic Web repositories are formulated. Solutions for the required search functionality and presentation methods are explored by the implementation of a prototype system.

Semantic search: text-based search for collections The third case study investigates the research questions in the context of artwork search. In many professional search tasks users want to find artworks that are somehow related to a topic. The study addresses the first research question by investigating how different types of relations in linked data can be used in text-based search functionality. We speculate upon the second research question by deriving implications for the presentation of search results and navigation paths in an interactive search application.

The starting point is the search engines currently used by domain experts, where results are found by a syntactic relation to the query. In a user study with cultural heritage experts we explore how to support the user in finding artworks that are semantically related to the query. The initial requirements for semantic search are formulated based on interviews with a number of domain experts, collecting realistic use cases and feedback on the use of different types of relations. The performance of an initial graph search algorithm is qualitatively evaluated by the analysis of the results found for a number of search log queries.

In addition to these three case studies, the architectural support for the proposed solutions are an intrinsic part of this thesis. The algorithms to support the required server-side search functionality and presentation methods are implemented as web services. To support re-use of our solutions for the development of web applications on other data sets or other domains a solution is explored to support configuration of the individual components.

The studies in this thesis are all carried out in a single domain, namely cultural heritage. Some of our findings, however, are also relevant to other domains with collections of annotated objects and controlled vocabularies. We make the implemented solutions available as open source software and provide support for other researchers to apply them to their own data.

The explored solutions focus on support for expert users and do not directly translate to applications for novices. At this explorative stage of the research field, lacking methods for meaningful quantitative analysis, we make the assumption that domain experts provide the most valuable qualitative feedback.

1.5 Contributions

The work presented in this thesis contributes to the development of interactive applications for end user access to heterogeneous linked data. The theoretical contribution of this thesis is an analysis of the problems related to selecting and presenting results from heterogeneous linked data. We provide:

- An analysis of the search functionality (RQ 1) and result presentation techniques (RQ 2) in state of the art Semantic Web applications. In addition we report on the methods used to evaluate these technologies.
- A statement of requirements on the search functionality (RQ 1) and the result presentation techniques (RQ 2) to apply keyword search and faceted browsing to heterogeneous linked data. In three case studies different aspects of the search problem are explored: finding vocabulary concepts to describe artworks, faceted browsing to formulate structured queries and finding artworks that are semantically related to a query.

The practical results are made available within the open source ClioPatria architecture.² This research has contributed to ClioPatria in two significant ways:

- Co-development of the ClioPatria architecture. In particular, the design and implementation of server-side algorithms and their APIs, for parameterized access over HTTP, and the implementation of cross-browser client-side interface widgets. The paper describing the ClioPatria architecture, written together with Jan Wielemaker, received an honourable mention at the In-Use track of the International Semantic Web Conference 2008.
- Implementation of configurable applications within ClioPatria for vocabulary-based annotation, faceted browsing and semantic search. Early versions of the applications were part of the E-Culture demonstrator that was awarded first prize at the Semantic Web challenge 2006³. A later version is used as a demonstrator for the Europeana thought lab⁴, and as demonstrators in the news (Troncy 2008) and music (Raimond and Sandler 2008) domains and to browse events (Shaw et al. 2009).

²<http://e-culture.multimedien.nl/software/ClioPatria.shtml>

³<http://challenge.semanticweb.org/>

⁴<http://www.europeana.eu/portal/thought-lab.html>

1.6 Structure of the thesis

The high level structure of this thesis consists of a discussion of the related work (Chapter 2), three case studies that each address the two research questions (Chapters 3, 4 and 5) and a discussion of the architectural support that is required for the proposed solutions (Chapters 6 and 7). Finally, Chapter 8 provides the conclusions.

In Chapter 2 we present a survey of semantic search and browsing applications and analyse how results are found and how results are presented.

In Chapter 3 we study term search in multiple semantically rich vocabularies to support professional cataloguers with subject matter annotation.

In Chapter 4 we explore faceted browsing as a generic solution for structured query formulation on heterogeneous Semantic Web repositories.

In Chapter 5 we study the use of semantically-rich vocabularies to support text-based search in artwork collections.

In Chapter 6 we describe the ClioPatria architecture, including the server-side algorithms for term and graph search and the APIs that make them accessible over HTTP.

In Chapter 7 we describe the models of client-side interface widgets for term search and faceted browsing and a method to configure these widgets with domain specific information.

In Chapter 8 we provide our conclusions and discuss our work in a broader context.

1.7 Publications

The research was proposed at the Doctoral Consortium of the International Semantic Web Conference 2008, where it was awarded Best Paper Doctoral Consortium:

- Michiel Hildebrand. Interactive Exploration of Heterogeneous Cultural Heritage Collections. In *Proceedings of the 7th International Semantic Web Conference*, pages 914–919, Karlsruhe, Germany, 2008.

Publications on which the chapters of this thesis are based:

- Chapter 2 contains material that will appear as: Michiel Hildebrand, Jacco van Ossenbruggen and Lynda Hardman. The role of explicit semantics in search and browsing. Chapter in *Multimedia Semantics: Metadata, Analysis and Interaction*, Raphal Troncy, Benoit Huet and Simon Schenk. Wiley, 2010.
- Chapter 3 was published as: Michiel Hildebrand, Jacco van Ossenbruggen, Lynda Hardman and Geertje Jacobs. Supporting subject matter annotation

using heterogeneous thesauri, a user study in Web data reuse. *International Journal of Human Computer Studies*, pages 887–902, Volume 67, Issue 10, October 2009.

- Chapter 4 was published as: Michiel Hildebrand, Jacco van Ossenbruggen and Lynda Hardman. */facet: A Browser for heterogeneous Semantic Web repositories*. In *Proceedings of the 5th International Semantic Web Conference*, pages 272–285, Athens, USA, 2006.
- Chapter 5 is submitted as a journal article: Michiel Hildebrand, Jacco van Ossenbruggen, Lynda Hardman, Jan Wielemaker and Guus Schreiber. Searching in semantically-rich linked data: a case study in cultural heritage.
- The material in Chapter 6 is the result of joint work with Jan Wielemaker, Jacco van Ossenbruggen and Guus Schreiber and published as: Jan Wielemaker, Michiel Hildebrand, Jacco van Ossenbruggen and Guus Schreiber. Thesaurus-based search in large heterogeneous collections⁵. In *Proceedings of the 7th International Semantic Web Conference*, pages 695–708, Karlsruhe, Germany, 2008. Honourable mention at the In-Use track.
- Chapter 7 was published as: Michiel Hildebrand and Jacco van Ossenbruggen. Configuring Semantic Web interfaces by data mapping. In *Workshop for Visual Interfaces to the Social and the Semantic Web*, Sanibel Island, USA, 2009.

Other publications resulting from the research described in this thesis:

- Jan Wielemaker, Michiel Hildebrand and Jacco van Ossenbruggen. Using Prolog as the fundament for applications on the Semantic Web. In *Proceedings of the ICLP'07 Workshop on Applications of Logic Programming to the Web, Semantic Web and Semantic Web Services*, Porto, Portugal, 2007.
- Jacco van Ossenbruggen, Alia Amin and Michiel Hildebrand. Why evaluating Semantic Web applications is difficult. *Semantic Web User Interaction Workshop*, Florence, Italy, 2008.
- Guus Schreiber, Alia Amin, Lora Aroyo, Mark van Assem, Viktor de Boer, Lynda Hardman, Michiel Hildebrand, Borys Omelayenko, Jacco van Ossenbruggen, Anna Tordai, Jan Wielemaker and Bob J. Wielinga. Semantic annotation and search of cultural-heritage collections: The MultimediaN E-Culture demonstrator. *Journal of Web Semantics* 6 (4), pages 243–249, 2008. Winner of the Semantic Web Challenge of 2006.

⁵This publication also included in the PhD thesis of Jan Wielemaker, Logic programming for knowledge-intensive interactive applications (Wielemaker 2009a).

- Alia Amin, Michiel Hildebrand, Jacco van Ossenbruggen, Vanessa Evers and Lynda Hardman. Organizing suggestions in autocompletion interfaces. In *31st European Conference on Information Retrieval*, Toulouse, France, 2009.
- Lynda Hardman, Jacco van Ossenbruggen, Raphal Troncy, Alia Amin and Michiel Hildebrand. Interactive information access on the Web of Data. In *Proceedings of the WebSci'09: Society On-Line*, Athens, Greece, 2009.
- Alia Amin, Michiel Hildebrand, Jacco van Ossenbruggen, Lynda Hardman. Designing a thesaurus-based comparison search interface for linked cultural heritage data. To appear in *Proceedings of the International Conference on Intelligent User Interfaces 2010*, Shanghai, China, 2010.

Chapter 2

Related work

In this chapter we identify different types of user-oriented search functionality and presentation methods to access Semantic Web data. We summarise our literature study of Semantic Web applications for search and browsing. The basic terminology is introduced and the different features found in the literature are analysed in the form of a survey. From this analysis, we identify a number of problems with user-oriented support for accessing Semantic Web data. Some of these are explored in the case studies in Chapters 3, 4 and 5, where we design and evaluate support for a number of the specific features discussed.

This chapter is based on a survey conducted in May 2007 and will appear as the chapter “The role of explicit semantics in search and browsing” in the book *Multimedia Semantics: Metadata, Analysis and Interaction* (Hildebrand et al. 2010). It is co-authored by Jacco van Ossenbruggen and Lynda Hardman. Since the original survey, the area of semantic search has grown and Web companies are starting to integrate semantic technologies into their search engines (Hendler 2010). In particular, advances have been made in the extraction of semantic relations using natural-language processing. This is, however, outside the scope of this thesis. In addition, we observe the uptake of semantic relations to improve the presentation of the search results, e.g. to provide more informative results and/or organise the results. Several of these strategies were already investigated in the prototypes discussed in this chapter.

2.1 Introduction

In recent years several Semantic Web applications have been developed that support some form of search. These applications provide different types of search

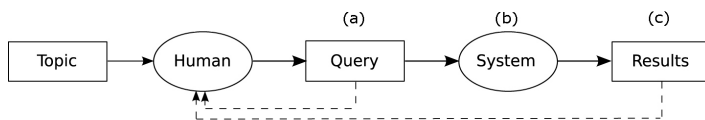


Figure 2.1: High level overview of text-based query search: (a) query construction, (b) search algorithm of the system, (c) presentation of the results. Dashed lines represent user feedback. Image based on <http://www-nlpir.nist.gov/projects/tv2007/>.

functionality and make use of the explicit semantics present in the data in various ways. The goal of this chapter is to give an overview of the semantic search functionalities provided by the current state of the art in Semantic Web applications. Excluding search based on structured query languages such as SPARQL (W3C 2006), we focus on queries based on simple textual entry forms and queries constructed by navigation (e.g. faceted browsing). We provide a systematic understanding of the different design dimensions that play a role in supporting search on Semantic Web data.

The next section establishes the basic search terminology used throughout the chapter. Section 2.3 provides an in-depth analysis of the different types of search functionality based on a survey of 35 Semantic Web applications. We argue that search is far from trivial, and list the different roles semantics currently play throughout the various phases of the search process. We discuss the main conclusions in section 2.4.

2.2 Basic Search Terminology

We introduce the basic definitions used throughout the chapter.

Search process We follow the TRECVID 2007 Evaluation Guidelines (TRECVID organizers 2007) and divide the search process into three different phases: query construction, execution of the core search algorithm and presentation and organisation of the results. We also take into account user feedback on the query and on the results (see also Figure 2.1).

Semantic search We use the term semantic search when semantics are used during any of the phases in the search process.

Type of results Traditional search engines on the web typically assume the result of a search to be a set of web objects, e.g. text documents, images or video. This also holds for some Semantic Web applications. Others, however, return sets of matching URIs, sets of matching triples (an RDF sub-graph)

or a combination of these. Often the behaviour of a system depends on the type of result it assumes, so we make this explicit wherever this is relevant.

Overview pages and surrogates We refer to presentation of the (k first) results as the *overview page*, which typically represent results using *surrogates*. For example, resulting HTML pages can be represented by their title and a text snippet, while image or video results are often represented by thumbnails.

Local view page Surrogates typically provide links to the full presentation of the result they represent. The latter is either the full presentation of an information object (e.g. the full HTML page that is linked from the snippet presented on the overview page), some human readable representation of metadata associated with the result, or a human readable representation of a resulting sub-graph. Following (Rutledge et al. 2005), we refer to the latter presentation as a *local view page*.

Syntactic matching Syntactic search matches the query against the textual content of the objects (if applicable), the literals in the RDF metadata, the URIs in the system, or a combination of these.

Semantic matching We use the term semantic matching for those algorithms that in addition to syntactic matching also use the graph structure of the RDF metadata and/or its semantics to find results.

2.3 Analysis of semantic search

We systematically scanned all proceedings of the International and European Semantic Web Conference series as well as the Journal of Web Semantics, to compile a list of end-user applications described or referred to. For each system we collected basic characteristics such as the intended purpose, intended users, the scope, the triple store and the technique or software that is used for literal indexing. The resulting document was made available online¹ and announced on three public mailing-lists². Additionally, we sent personal emails to the authors of papers and developers of all included systems. This resulted in an updated version of the online document. This update was based on 15 email threads, in which additional information was provided for 11 systems and 6 additional systems within the scope of the survey were recommended, giving a total of 35 systems.

Based on the data resulting from the survey we performed a more thorough analysis of the three individual phases in the search process of Figure 2.1. For each of these we consider the underlying functionality and the features of the corresponding user interface. The results of this analysis are summarised in Table 2.3,

¹http://swuiwiki.webscience.org/index.php/Semantic_Search_Survey

²public-xg-mmsem@w3.org, semantic-web@w3.org, public-semweb-ui@w3.org

Query construction		
Feature	: Functionality	Interface Components
Free text input	: Keywords, natural language	Single text entry, property-specific fields
Operators	: Boolean constructs, syntactic disambiguation, semantic constraints on input/output	Application-specific syntax
Controlled terms	: Disambiguate input, restrict output, select predefined queries	Value lists, faceted browser, graph
User feedback	: Pre-query disambiguation	Autocompletion
Search algorithm		
Syntactic matching	: Exact, prefix or substring match, minimal edit distance, stemming	Not applicable
Semantic matching	: graph traversal, query expansion, spread activation, RDF-S/OWL reasoning	Not applicable
Result presentation		
Data selection	: Selected property values, class-specific template, display vocabularies	Visualised by text, graph, tagcloud, map, timeline, calendar
Ordering	: Content and link structure based ranking	Ordered list
Organisation	: Clustering by property, by result path or dynamic	Tree, nested box structure, clustermap
User feedback	: Post-query disambiguation, recommendation of related objects	Facets, tagcloud, value list

Table 2.1: Functionality and interface support in the three phases of semantic search.

and the table also provides the structure of the remainder of this section. We discuss query construction in section 2.3.1, the search algorithms in 2.3.2 and the presentation of the results in 2.3.3. Note that the examples and references merely serve as illustrations, the full analysis with references is available in the online survey.

2.3.1 Query Construction.

The search process starts with the user constructing a query that reflects his or her information needs. We describe the functionality for this process as provided by the systems in the survey and how this functionality is supported at the interface.

Functionality Constructing a query in free text requires little knowledge of the system and the data structure. The price users have to pay is ambiguity: words can have multiple meanings (lexical ambiguity) and a complex expression can have multiple underlying structures (structural ambiguity). Ambiguous input often leads to irrelevant search results. To reduce ambiguity several systems allow additional query constructs beyond free text input: structural and semantic operators and controlled terms. We describe free text input and the additional query constructs and the role of user feedback in the process of matching free text with controlled terms.

Free text input is supported in existing systems in three ways. First, full text search allows the user to find all objects with matching textual content or metadata. In many semantic search engines full text search is the main entry point into the system (Ding et al. 2005; Schreiber et al. 2006; Celino et al. 2006; Guha et al. 2003; DERI 2007). Second, free text input can be restricted to match a value of a specific property. In faceted browsers this is the case when searching for a value within a particular facet (SIMILE 2005; m.c. schraefel et al. 2005; Hildebrand et al. 2006). Finally, systems such as Aqualog (Lopez et al. 2005) and Ginseng (Bernstein et al. 2006) support free text input in the form of natural language expressions.

Syntactic operators explicitly define the interpretation of complex search terms. Well known examples are the boolean operators AND and OR. Several applications employ third party search libraries such as Apache Lucene, which typically provide an extensive collection of syntactic control structures³.

Semantic operators add explicit meaning to a query. In SemSearch, for example, the user specifies to which RDFS or OWL class a result should belong. The authors illustrate this with the example `article:motta` for which the system retrieves all objects that are of type `article` and match the search term `motta` (Lei et al. 2006).

Controlled terms provide the use of predefined concepts. In QuizRDF (Davies and Weeks 2004) the user selects an RDF class to determine the type of the search term. Other systems provide autocompletion to support users with keyboard-based input of controlled terms (SIMILE 2005; m.c. schraefel et al. 2005; Hildebrand et al. 2006; Kiryakov et al. 2004; Hyvönen and Mäkelä 2006). A different approach is seen in DBin (Tummarello et al. 2006) and Haystack (Quan and Karger 2004), which allows the user to select predefined queries.

³<http://lucene.apache.org/java/2.3.2/queryparsersyntax.html>

User feedback on the input is useful when there are multiple controlled terms that match with the free text input. Several systems allow the user to select the intended term before it is processed by the search algorithm (Celino et al. 2006; Hildebrand et al. 2006; Hyvönen and Mäkelä 2006). This form of user feedback allows pre-query disambiguation. In contrast, post-query disambiguation is performed on the results of the search algorithm.

Interface The basic interface components to enter or construct a query are text entry boxes and value selection lists. These components are used in various designs, of which we mention three. If applicable, we describe the link from the interface components to the underlying data structure. Furthermore, we describe the interface aspects of user feedback on the input and several proposals for more advanced query construction.

A *single text entry field* is sufficient for free text input, e.g. Google and several systems that we analysed (Schreiber et al. 2006; Celino et al. 2006; Guha et al. 2003; Davies and Weeks 2004; Ding et al. 2004). Additional features included by some systems are selectable result types (Ding et al. 2004) and options for the search algorithm or presentation of the results (Davies and Weeks 2004).

Property-specific search fields support query construction guided by a specific set of possible search values (Kiryakov et al. 2004; Mika 2006; Heflin and Hendler 2000; Metaweb 2007). The value sets are typically defined by the range of the corresponding RDF property.

Faceted browsing allows the user to constrain the set of results within a particular facet. Typically, facets are directly mapped to properties in RDF. Alternatively, the mapping is made by projection rules. The advantage of an indirect mapping is that this allows the developer to define facets that match the user's needs while keeping the data structure unchanged (Suominen et al. 2007a). Faceted browsing is applied to Semantic Web data by (SIMILE 2005; m.c. schraefel et al. 2005; Hildebrand et al. 2006; Suominen et al. 2007a; Fluit et al. 2003a; Hyvönen et al. 2005; Oren et al. 2006) as well as by the company Siderean in the Seamark Navigator⁴.

User Feedback is typically provided after the query has been entered, or dynamically during the construction of the query as a form of *semantic autocompletion*. The former method is used in Squiggle (Celino et al. 2006) and MuseumFinland (Hyvönen et al. 2005) where the disambiguation of the matching query terms is presented after submitting the query. In semantic autocompletion the system suggests controlled terms with a label prefix that matches the text typed in already. Hyvönen and others describe the idea of semantic autocompletion and several implementations in (Hyvönen and Mäkelä 2006). In faceted interfaces autocompletion is often used within a single facet (SIMILE 2005; m.c. schraefel et al. 2005; Hildebrand et al. 2006; Kiryakov et al. 2004).

⁴<http://www.siderean.com/>

In general there is a clear need for simple interfaces, such as the single text entry field. On the other hand, interface designs that support more complex interaction styles potentially give the user more control, which is useful for the formulation of more precise information needs.

2.3.2 Search algorithm

All text-based search involves some form of syntactic matching of the query against textual content and/or metadata, an aspect well covered in Information Retrieval (Baeza-Yates and Ribeiro-Neto 1999). Semantic matching can extend syntactic matching by exploiting the typed links in the semantic graph. Before we describe semantic matching we briefly describe syntactic matching, focusing on the indexing functionality and the support that is already provided by the low level software on which various systems are built.

Syntactic matching All systems in our study index the textual data in their collection for performance reasons. Which textual data is indexed, e.g. the content, the metadata or the URIs, is important for the search functionality of the system. In an ontology search engine such as Swoogle, users might want to search on URIs (Ding et al. 2005). In annotated image collections the metadata forms the primary source for indexing (Schreiber et al. 2006; Celino et al. 2006; Hyvönen et al. 2005). For images that occur in web pages the contextual text provide an alternative source (Celino et al. 2006; Group 2006). Indices can be based on the complete word or on a stemmed version. Some interface functionalities require additional features. Autocompletion interfaces, for example, require efficient support for prefix matching. Some triple stores provide built-in support for literal indexing, for example, OpenLink Virtuoso⁵ and SWI Prolog's Semantic Web library⁶. Alternatively, a search engine, such as Lucene⁷, can be used together with a triple store.

Semantic matching After syntactic matching, the structure and formal semantics of the metadata can be used to extend, constrain or modify the result set. Note that in a connected RDF graph, any two nodes are connected by a path in this graph. Naive approaches to semantic search are computationally too expensive and increase the number of results dramatically. Systems thus need to find a way to reduce the search space and to determine which semantically related objects are really relevant.

Inspired by the semantic continuum described by Ushold (Ushold 2003), we distinguish three levels of semantic matching: graph traversal, explicit use of the-

⁵<http://www.openlinksw.com/>

⁶<http://www.swi-prolog.org/packages/semweb.html>

⁷<http://lucene.apache.org/>

sauri relations and inferencing based on the formal semantics of RDF, RDFS and OWL.

Graph traversal takes only the structure of the graph into account. Several techniques are in use to constrain graph search algorithms. In Tap, constraints define which relations to traverse for the instances of a particular class (Guha et al. 2003). Alternatively, a weighted graph search algorithm may constrain the possible path structures and path length. Such an algorithm requires the assignment of weights to the edges in the graph, where the weights reflect the importance of the corresponding RDF relations. In e-Culture, weights are manually assigned to RDF relations (Schreiber et al. 2006). SemRank automatically computes weights based on statistics derived from the graph structure (Anyanwu et al. 2005). Spread activation (Rocha et al. 2004) is another computationally attractive technique for graph traversal, which can incorporate weights as well as the number of incoming links.

Thesaurus relations are sometimes used for query expansion. With the acceptance of SKOS (W3C 2005) as a standard representation for thesauri, semantic matching with hierarchical broader term (BT) and narrower term (NT) and the associative related term (RT) can be implemented in a generic way. The Squiggle framework is an example in which this is done (Celino et al. 2006). Facet browsers typically rely on hierarchical thesauri relations to restrict their result sets (Hildebrand et al. 2006; Hyvönen et al. 2005). Within the FACET project the integration of thesauri in the search process is studied extensively. This resulted in a demonstrator as well as a proposal for a semantic expansion service (Binding and Tudhope 2004a), which in turn formed the basis for the experimental SKOS API⁸.

RDFS/OWL reasoning can also influence the search results. Several systems support RDFS subsumption once an RDF class is selected in the interface (Lei et al. 2006; Tummarello et al. 2006; Auer et al. 2006; Duke et al. 2007). In Dose, specialisation and generalisation over the subclass hierarchy is used dynamically according to the number of search results (Bonino et al. 2004). Some systems support partial OWL reasoning, and process, for example, only the OWL identity relations. In Flink (Mika 2005) and SWSE (DERI 2007) these are extensively used to model the identity between extracted entities.

2.3.3 Presentation of Results

We describe how explicit semantics are used to extend the baseline functionality in the presentation of search results, and the techniques that are used to visualise the results in the interface. As a baseline we consider the presentation of search results by popular search engines such as Google: the selected information for presentation is the URI or label of the result, surrogates of the content (e.g. text snippets or

⁸<http://www.w3.org/2001/sw/Europe/reports/thes/skosapi.html>

image thumbnails) and, optionally, additional information such as the file size. The results are typically organised in a plain list and ordered by relevance. We describe, for each aspect, how additional semantics are used to extend this baseline.

Functionality We consider three aspects of the presentation: *selecting* what data to present, *organising* the results and *ordering* the results. In addition, we discuss the function of user feedback on the results.

Selecting what to present — This issue is tightly bound to the question what the search engine considers to be a “result”. If the result is a Web page or other information object, traditional surrogates are typically used. When the search result is a set of URIs referring to nodes in an RDF graph, or a set of RDF triples, systems need to invent new ways to represent the results in their overview page. In most systems we studied, the decision on what (meta)data is used for the surrogates is hardwired into the system. QuizRDF supports template definitions for each RDF class (Davies and Weeks 2004). Dbin (Tummarello et al. 2006) create templates for specific user tasks and domains. Display vocabularies such as Fresnel (Bizer et al. 2005), as used by Longwell (SIMILE 2005), provide full control over what data to select for presentation and how to present it.

Organising the results — Semantics can also play a role in grouping semantically similar results together in the presentation, a feature commonly referred to as *clustering*. Assuming that users are interested in the results of only one cluster, clustering can also be considered as a form of post-query disambiguation. In our study we found several forms of clustering. In many systems, the values of a particular property are used to group the result set on common characteristics within a particular dimension. In (Guha et al. 2003) results with similar types are clustered together. In faceted browsers similar behaviour is found, systems described in (SIMILE 2005; Hildebrand et al. 2006; Hyvönen et al. 2005) all support clustering on the values of a particular facet. Noadster uses concept lattices to determine dynamically which properties to use for a given result set (Rutledge et al. 2005). In e-Culture (Schreiber et al. 2006), the RDF path between the literal that is syntactically matching the query and the result may span more than one property. Clustering the results on these paths illustrate the interpretations of the query.

Ordering of results — The order of the search results can be determined with different techniques. Ranking of results based on relevance is a well covered topic in Information Retrieval (Baeza-Yates and Ribeiro-Neto 1999). Numerous algorithms have been developed, evaluated and applied in successful applications. Term frequency-inverse document frequency (tf-idf) is an often used syntactic measure to determine the importance of a word based on the number of occurrences in a document relative to the number of occurrences in the entire collection. Many systems in our study use Lucene, which provides ranking based on tf-idf. In addition to textual content, the link structure is another source for ranking. Swoogle uses

a variant of PageRank (Page et al. 1998) to measure the relevance of RDF documents. PageRank was adapted to compensate for different types of relations that link RDF documents and terms (Ding et al. 2005). In SWSE (Hogan et al. 2006) a variant of PageRank based on the principle of focussed subgraphs (Kleinberg 1999) is used.

User Feedback — In our study, we did not encounter typical IR user feedback where the matching and ranking algorithms is influenced by the user’s feedback. We mainly encountered user feedback to disambiguate, specialise, generalise or expand the result set. Most systems support expansion of a query by adding a keyword or by selecting a value from a property field or facet. In several systems, post-query disambiguation of free text input is supported through the selection of an RDF type (DERI 2007; Davies and Weeks 2004; Duke et al. 2007; Berlin 2007). Alternatively, queries can be specialised or generalised with concepts from narrower or broader thesaurus relations (Celino et al. 2006; Hildebrand et al. 2006; Hyvönen et al. 2005). An unwanted side effect of query refinement is the risk of ending up with no results. This can be avoided by restricting the user beforehand to use only those terms that lead to results. This is one of the principles behind faceted browsing interfaces (Yee et al. 2003).

We observed that domain-specific applications use the semantics to organise the search results into clusters. Domain-independent search engines typically rely on ranking techniques for effective presentation of the search results.

Interface Most systems provide a straight-forward interface that directly reflects the structure of the selected data and how it is organised. Typical examples include numbered lists for a linearly ranked set of results or visual grouping of clustered results in nested box layout structures. Since RDF is represented as a graph, visualising the data as a graph may seem a straightforward choice. However, from a user interface perspective, “big fat graphs” quickly become unmanageable (m.c. schraefel and Karger 2006), with only a few exceptions including the visualisation of social networks between small groups of people (Mika 2005). We discuss some other visualisation techniques (see (Geroimenko and Chen 2003) for a more extensive overview) we encountered.

Tagclouds indicate the importance of textual metadata with variations in the font size. OpenAcademia (Mika 2006) visualises the concepts related to research publications and search. DBpedia.org (Berlin 2007) presents the available RDF types of the search results.

Clustermaps visualise the overlap between classes of instances, without needing an explicit concept representing this overlap (Fluit et al. 2003b). For example in AutoFocus a clustermap visualises the results of individual constraints as well as result sets that satisfy multiple constraints (Fluit et al. 2003a).

Data type-specific visualisations are used in several systems to present space and

time on a geographical map, timeline or calendar. The Simile timeline⁹, several map visualisation tools and Google Calendar can be used through publicly available APIs. Hence, we do not list the individual systems that make use of these.

Local view pages provide a detailed presentation of the metadata associated with single URI. Systems such as Tabulator (Berners-Lee et al. 2006) and Disco (Bizer and Gauß 2007) are based on the notion of a *concise bounded description* or CBD¹⁰ and present the statements where the current focus URI is a subject. Others systems' local view pages may contain all statements in which the URI occurs either as a subject or object. Sesame's URI explorer pages¹¹ Noadster (Rutledge et al. 2005) and E-culture (Schreiber et al. 2006) also include statements where the URI plays the role of the property.

2.4 Discussion

In a survey we investigated the search functionality and result presentation of 35 Semantic Web applications. We conclude that the applications support a wide variety of different types of tasks and provide access to different types of data sets. In addition, they provide different types of support for the three stages of the search process: query formulation, search algorithm and result presentation. It, however, remains unclear how well these technologies improve support for end-users. With a few notable exceptions (Ding et al. 2004; Sure and Iosif 2000), the search algorithms analysed in this study are not or only briefly evaluated on the quality of their search results for end-users. A similar argument applies to the interface components found in our study. For none of the systems we could find user evaluations that would stand the criteria commonly found in the HCI community.

In Information Retrieval, there is a long tradition of evaluating the quality of retrieval systems. Conference series such as TREC and INEX contribute to an (evolving) community consensus about which dimensions to evaluate, and how to measure a system's performance on that dimension. It is safe to say that within the Semantic Web community, we have not yet developed a similar consensus about the use of explicit semantics to improve search, and how to evaluate and to compare semantic search systems.

We conclude that given a specific domain and search task it is difficult to determine how the semantically-rich graph structure should be used to benefit the end user. In the following three chapters we, therefore, investigate support for specific tasks and use qualitative evaluations to gather insights in the requirements to support end-user access to semantically-rich linked data.

⁹<http://simile.mit.edu/timeline/>

¹⁰<http://www.w3.org/Submission/CBD/>

¹¹<http://www.openrdf.org/>

Chapter 3

Case study I: Subject matter annotation

In this chapter we investigate, for a specific task, the use of graph structures in search functionality and result presentation to support users searching in multiple heterogeneous vocabularies. The task this case study focusses on is annotation of historical prints by domain experts. In cooperation with professional cataloguers we develop a prototype that supports annotation with terms from multiple vocabularies and we qualitatively evaluate this prototype in a user experiment. The study shows that a search algorithm requires different configurations to provide effective support for different annotation fields. In addition, it shows that different types of terms and vocabularies require different presentation and organisation methods.

This chapter was published as “Supporting Subject Matter Annotation Using Heterogeneous Thesauri: A User Study In Web Data Reuse” in the International Journal of Human Computer Studies (Hildebrand et al. 2009) and was co-authored by Jacco van Ossenbruggen, Lynda Hardman and Geertje Jacobs.

3.1 Introduction

We report on a user study that investigates how museum professionals search for appropriate terms within multiple thesauri during an annotation task. The study was performed within the *Print Room Online* project at the Rijksmuseum Amsterdam for a period of 11 weeks, and includes a field study to gather information about the current annotation practices, the iterative design of a prototype interface to support annotation of subject matter and a user experiment to test the final prototype. We discuss the outcome of this study in terms of the requirements on

the underlying RDF data, the application's search functionality and user interface design.

Our prototype can be seen as an example of an application that reuses available Web resources and re-purposes rich and highly heterogeneous linked data to support users in a specific task. Although our insights are collected in very specific domain and for a specific task, our observations can be generalised in two ways. Firstly, to annotation scenarios at other museums, (audio/video) archives and libraries, as many issues also apply to their subject matter annotation tasks. Secondly, to other scenarios in which the reuse of Web data should aid the end user, as the issues we tackle are likely to occur in Semantic Web applications dealing with heterogeneous data. We generalise our findings on the needs for information disambiguation, alignment, multilingualism, compound query support and result visualisation and organisation to make them relevant for a wider range of applications that reuse Web resources and/or Semantic Web technology.

This chapter is organised as follows. In Section 3.2 we document the current annotation practice at the *Print Room Online* project of the Rijksmuseum. We discuss other approaches to thesaurus-based annotation in Section 3.3. In Section 3.4 we sketch the phases of the study that are covered in the following sections: we identify the requirements for subject matter annotation in Section 3.5, refine these requirements by process of an iterative user interface design in Section 3.6, and test the resulting prototype in a user experiment discussed in Section 3.7. We present conclusions and future work in Section 3.8.

3.2 Current annotation practices at the Rijksmuseum

The Print Room of the Rijksmuseum in Amsterdam, the Netherlands, has a collection of about 700,000 prints, drawings and photographs. Within the project *Print Room Online* the Rijksmuseum aims to register the basic properties of each print, such as the object ID, storage location, title, creator and measurements. In addition, the museum aims to make the collection accessible to the public by making high quality digital scans and adding subject matter annotations. The latter refers to the description of what is depicted on a print and is the focus of this chapter. The upcoming three years the project will catalogue 100,000 objects and make them accessible through the museum's website, www.rijksmuseum.nl.

We describe the annotation environment of the *Print Room Online* project and the practices from before the start of our user study. The annotation is performed by seven professional cataloguers. These are highly educated domain specialists, each with knowledge of a particular part of the domain. To improve consistency, the project management has developed an extensive annotation guideline document based on the Spectrum¹ and CIDOC² guidelines. The guidelines for the

¹<http://www.collectionstrust.org.uk>

²<http://cidoc.mediahost.org>

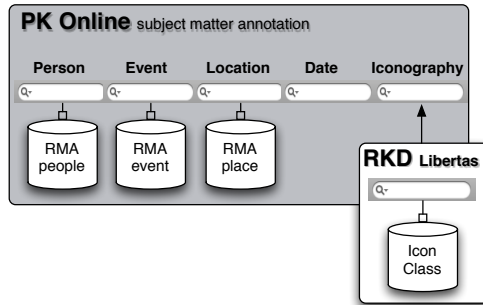


Figure 3.1: Sketch of the current Rijksmuseum setup for subject matter annotation. Only the five fields that are actually used are shown here. The person, event and location fields each give access to an associated, internal thesaurus. The codes in IconClass are looked up using another tool, the *Libertas* browser, after which they are copy-pasted into the Iconography field. The dates are entered in a free text field that has no associated thesaurus.

basic registration are straightforward and could be easily applied within the Rijksmuseum's current environment. For the subject matter annotation there is, however, little consensus within the cultural heritage community and very limited tool support. The management team of *Print Room Online* decided to use a temporary solution. The cataloguers are instructed to describe the depicted person/organisation, event name, place and date. In addition, one or more codes from an externally developed classification system, (ICONCLASS³), should be added when applicable. To save time and to achieve the required throughput rates the subject matter annotation is limited to the main theme being depicted on the object.

Figure 3.1 shows a schematic view on the annotation fields and thesauri used. The terms used are selected from three internally curated thesauri⁴ covering people, places and events. The internal thesauri are stored and accessed in the museum collection management system of which the current annotation facilities are an integral part. Another application, the online IconClass *Libertas* browser, is needed to search for the codes from ICONCLASS.

After about one year, 30 747 objects from the Print Room have been annotated. We analysed the annotations made for these objects. Table 3.1 shows per thesaurus the number of terms used to annotate the objects. This shows that 55% of the

³<http://www.iconclass.nl>

⁴We use the abbreviation RMA to refer to a thesaurus of the Rijksmuseum Amsterdam

	People	Place	Event	IconClass	Total
#	9,245	9,034	6,509	30,981	55,796
%	17	16	12	55	100

Table 3.1: For each thesaurus, the total number of terms used for the annotation of the Rijksmuseum Print Room objects and the percentage relative to the total number of terms used.

	People	Place	Event	Total
Terms in thesaurus	65,325	44,541	2,824	112,690
Terms used (total)	9,245	9,034	6,509	24,778
Terms used (unique)	1,574	1,523	492	3,589
New terms added #	516	169	205	890
New terms added %	33	11	42	25

Table 3.2: Thesaurus terms used for subject matter annotation for the 30 747 Print Room objects. Note the high percentage of missing depicted persons and events that needed to be added.

annotation terms used are from the externally developed thesaurus, ICONCLASS. Note, in this research we have considered the classification system ICONCLASS as any other thesaurus. Table 3.2 shows more details about the usage of the museum in-house thesauri. The table lists the total number of terms in each thesaurus, the total number of terms used for subject matter annotation, the number of unique terms used, the number of terms that had to be added to the thesaurus during the *Print Room Online* project and the percentage of added terms w.r.t. to the unique terms. The table shows that a large number of new thesaurus terms had to be added, in particular for the depicted persons and the depicted event fields. According to the cataloguers, it takes on average about 15 minutes to add a new term to a thesaurus, including the research time. Converted, this means that one person has been adding thesaurus terms for a full month, instead of cataloguing objects.

The museum management realises that curating these thesauri is expensive, and that despite the large efforts, their coverage remains limited. In-house thesauri often reflect the specific perspective of a relatively small group of domain experts, which may cause difficulties when the resulting object descriptions are

exposed to a wider user group, for example on the museum's public website. Internal thesauri, including those of the Rijksmuseum, tend to be mono-lingual, which makes the annotations less useful for search applications in a multi-lingual context. Additionally, the quality of the thesauri tends to degrade over time. For example, as different members of the organisation add new terms to a thesaurus, the differences in style and the unintended creation of duplicates makes searching for appropriate terms more difficult. Finally, creating a vocabulary with a sufficiently large coverage is virtually impossible when the vocabulary is created by a small group of experts. A result of this is that cataloguers frequently find that the term they need is missing, and are forced to collect the required information before it can be added to the vocabulary. This is a time consuming task that significantly slows down the annotation process.

In contrast to in-house developed thesauri, other, more widely available thesauri⁵ are developed and maintained by other parties. These thesauri often reflect the perspective of a broader team of experts, are partially available in multiple languages, are actively maintained with clear quality control guidelines, and, finally, typically provide a much wider coverage of the target domain. The museum often uses these thesauri as a starting when creating new terms for their own thesauri. Recent standardisation efforts, such as SKOS (W3C 2005), significantly lower the technical boundaries to publish thesauri on the Web and reuse them in a specific annotation tool. A drawback of some of these thesauri (e.g. WORDNET) is that they are too general and lack terminology required by a specific museum. A more general drawback of using external thesauri is that the museum loses full control on the content of the thesauri used, and that the thesauri may overlap (that is, the combined thesauri will most likely contain duplicates) and that the thesauri might describe terms from a different perspective than that of the museum.

An annotation tool that would be able to effectively use both internally and externally developed thesauri could, in theory, combine the advantages of both approaches. The key research question is whether we can design an annotation tool in which cataloguers can quickly find an appropriate term from multiple heterogeneously structured thesauri. In particular, we are interested in the requirements on the thesauri and other data needed, on the underlying search algorithms deployed to search multiple thesauri, and on the user interface design, the visualisation and organisation of the term search results.

3.3 Related work

For an overview of image annotation on the Semantic Web we refer to the report of the W3C Incubator Group on this topic (W3C 2007). Here, we focus on different

⁵Examples include thesauri from the Getty institute, in the Netherlands the artist and art historic (ICONCLASS) thesauri from the Institute for Art History (RKD) and various versions of WORDNET in different languages. More details are presented later in this chapter.

techniques to support end-users with finding (annotation) terms from vocabularies.

Hollink et al. (Hollink et al. 2003) describe an early semantic image annotation tool that supports (subject matter) annotation using terms from different thesauri. An interesting feature is the support for restrictions to limit the search results of annotation fields to terms from specific parts of a thesaurus hierarchy. For example, when searching for terms of a depicted activity only terms from the “activity” branch of WORDNET were suggested. Although the tool allowed searching in multiple thesauri for a single annotation field it does not scale well to the number of terms we expect to use for the Rijksmuseum. In particular, we need more scalable visualisation and organisation of the search results that can also be adapted for the characteristics of specific thesauri.

Finding terms from thesauri is supported in several systems. The FACET project provides several services on thesauri, in particular, to use the thesauri for semantic expansion (Binding and Tudhope 2004b). The Finish Ontology Service Infrastructure, FINNONTO (Hyvönen et al. 2008), provides several web services for SKOS thesauri, and support simultaneous access to multiple thesauri. Their services can also be used with a client side autocompletion widget to look up thesauri terms.

Generic Semantic Web search engines such as Sindice (Tummarello et al. 2007), Swoogle (Ding et al. 2005) and Falcon (Cheng et al. 2008) give access to many vocabularies at once. Often a query leads to several pages of search results, requiring the user to select the most appropriate term. Determining the most appropriate term means, in this case, studying the RDF document a term belongs to, something we can not expect from our end users.

Autocompletion is a technique that continuously provides suggestions while the user is typing. Autocompletion has been applied in applications for many decades. It is in particular successful for “term” search tasks with a limited vocabulary (Hildebrand et al. 2007), such as email addresses. Autocompletion is also applied to thesaurus term search. Hyvönen et al. provide an overview of different types of semantic autocompletion in (Hyvönen and Mäkelä 2006). Sinkkilä et al. propose to combine context navigation with autocompletion (Sinkkilä et al. 2008). They did not experiment with the applicability of semantic autocompletion to end users. In previous user studies we showed that the most suited visualisation and organisation of autocompletion results differs per thesaurus (Amin et al. 2009).

For the annotation prototype of the Rijksmuseum our aim is to use autocompletion with multiple large and heterogeneous thesauri to efficiently find terms. This means we need to be able to access multiple thesauri simultaneously, provide scalable search and presentation, and configure the visualisation and organisation of the search results for different types of thesauri.

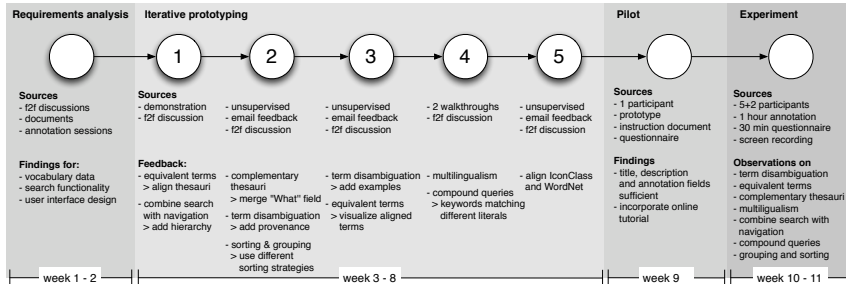


Figure 3.2: The study at the Rijksmuseum Print Room included a requirements analysis (Sect. 3.5), an iterative user interface design phase (Sect. 3.6), a pilot to test the experimental setup and the user experiment (Sect. 3.7).

3.4 User study

We performed a study at the Rijksmuseum Print Room. The aim of our study is a) to formulate requirements for multi-thesauri term search in subject matter annotation and b) acquire insights into the use of Semantic Web technologies when applied in a real life setting. We decided to collect qualitative information about the annotation practices and test different solutions for multi-thesauri term search. As our study is performed with a small group of experts it is in this stadium not realistic to strive for quantitative data.

The study was performed over a period of 11 weeks and consisted of several phases, as depicted in Fig. 3.2. The first phase consisted of an analysis of the project's requirements. We present the details of the requirements analysis phase, the main findings and its implications in Sect. 3.5. Based on these findings we developed a first prototype, which was refined in a process of iterative user interface design. In Sect. 3.6 we describe the second phase: the implementation of the prototype, the feedback acquired during prototyping, our interpretation of the feedback in terms of design dimensions and the decisions we took. In the third phase we tested our experimental setup with a pilot study, after which we performed a user experiment in phase 4. We describe the details of the experimental design and the key observations from the experiment in Sect. 3.7.

3.5 Requirements analysis

The initial contact with the Rijksmuseum *Print Room Online* project consisted of two sessions in which we acquired information about the current annotation process and formulated the requirements for the prototype. The first session consisted

of face-to-face discussions with the project leader, lead cataloguer and a curator of the Rijksmuseum. In the second session, we observed the lead cataloguer annotating several prints during working hours. In addition, we analysed and discussed the project's extensive cataloguing guideline document and a brief additional document in which the project management sketched their requirements and wishes for subject matter annotation.

3.5.1 Findings

Due to the extensive guidelines, developed within the *PK Online* project, the basic registration is performed relatively consistent. The annotation of the subject matter, however, remains problematic, as the Rijksmuseum's thesauri do not provide sufficient coverage (see Tab. 3.2) nor sufficient quality (clogging and limited additional information) needed for adequate annotation. As the project management believes that the integration of externally developed thesauri could improve subject matter annotation in particular, we focused hereon and did not incorporate the basic registration into our prototype.

Based on discussion and observations from several annotation sessions we identified three types of term search tasks relevant for annotation:

1. The user already knows which term to add and needs to (quickly) find this term in one of the thesauri.
2. The user has yet to discover the most suitable term and needs to explore the thesauri to find it.
3. The user suspects a term is not present in any of the thesauri, and needs to confirm this before adding it to one of them.

Our aim is to support all three tasks in a single user interface. Below, we formulate the initial requirements for the vocabulary data, the search functionality and the user interface design.

3.5.1.1 Vocabulary data

In the Museum's current collection management system, the annotation fields for person, event and place require terms to be selected from one specific thesaurus. Annotating prints for which all required terms are already in the thesauri is an efficient process, with most of the time being spent on researching what is being depicted, and relatively little time spent on actual data entry. This picture, however, changes dramatically when terms are missing. First, the cataloguer needs to do an exhaustive term search to be sure the term is really missing. Then she needs to research and formulate a request to add the term to the thesaurus, detailing exactly what term needs to be added, along with the additional information

that needs to be recorded (e.g. biographical data for persons, geographical data for places), and the provenance data of the literature sources upon which this information is based. When the request has been filed, the cataloguer continues her work, but needs to remember to come back to the annotation record of the associated artwork to add the annotation once the missing term has been added to the thesaurus. The whole process is extremely time consuming and disrupts the normal annotation work flow. Not surprisingly, the project would like to include additional thesauri to increase the term coverage for all three fields currently associated with the internal thesauri (depicted persons, events and places).

One externally developed thesaurus, ICONCLASS, is already used for the iconographic annotations. Surprisingly, this thesaurus has not been integrated into the museum's annotation interface or collection management system. Instead, the cryptic ICONCLASS term identifiers (e.g. 45K21) are looked up in an independent web-based interface for ICONCLASS and copy-pasted from the web browser into the annotation field or typed in manually. While cataloguers regularly make mistakes during this process, it is hard to detect such errors because the current tool does not contain the textual labels associated with the cryptic term identifiers.

The terms from ICONCLASS are well suited for the annotation of biblical and mythological stories depicted on museum objects. For annotating more general objects and concepts depicted on other prints and photographs, a more general thesaurus would be required. Since such a thesaurus is currently not available within the project, such annotations are either omitted or added to a free text description field, thereby losing all advantages of thesaurus-based annotation. While for the depicted persons, events and places, the project would like to add external thesauri to increase coverage (e.g. having more terms of similar nature), here an additional thesaurus would need to provide a different type of term (e.g. general terms instead of specific terms).

3.5.1.2 Search functionality

In the annotation fields of the museum's current system, cataloguers use keyword search to find terms from a specific thesaurus. A single query gives access to a separate page with matching terms from a single thesaurus, which is directly associated with an annotation field. In the current interface, many queries already yield long lists of results, and cataloguers fear this will only get worse when more thesauri are added. The long lists make even the first search task, finding a known term, relatively difficult.

The current tool provides no other means to access a thesaurus beyond keyword search. This limits the cataloguer with the second term search task, when the right annotation term is not known in advance and the cataloguer has to discover the most suitable term. In a cleanup process of the RMA PEOPLE thesaurus, the Rijksmuseum staff removed many duplicate artist names and added mappings to

non-aligned equivalent terms. The occurrence of these duplicates is an indication that the third task, confirming that a term is not present, is also not well supported.

3.5.1.3 User interface design

The annotation interface of the museum's current system consists of many tabs, showing a maximum of 10, each containing many annotation fields. For the annotation of the depicted subject matter, within the project only five out of the 33 available subject matter annotation fields are actually used. The project management expressed the need to simplify the annotation interface by significantly reducing the number of annotation fields and improving the layout.

The search results in the Rijksmuseum's current annotation interface are visualised by simply showing the term itself. Details about a term are only available in a separate window that is shown on request. Comparing terms within the search results is made more complicated with this type of result visualisation, as it requires multiple clicks. What information should be used for visualisation depends on the thesaurus. For example, in the IconClass browser, used by the Rijksmuseum cataloguers, the term identifier is also shown in the search results. This identifier indicates where in the hierarchy the term occurs, which helps experienced cataloguers to quickly determine if a result is the right term. For other thesauri, completely different information might be more suited for the visualisation of search results.

3.5.2 Implications

The findings above were discussed and translated to design decisions for the initial prototype in terms of the vocabulary used, the supported search functionality and the interface design.

3.5.2.1 Vocabulary data

After discussion with the project manager and lead cataloguer, we decided to build a prototype which integrated data from five external thesauri, in addition to the three internal thesauri developed and maintained by the Rijksmuseum. Our aim was to cover different aspects with multiple thesauri. As the quality of the data is important for the museum we only integrated internationally respected sources. Figure 3.3 shows the thesauri used per annotation field. We decided to use two additional thesauri with individuals: Getty's United List of Artist Names⁶ (ULAN) and DBpedia's RDF version of person data⁷ from Wikipedia; and one additional source of place names: Getty's Thesaurus of Geographic Names⁸ (TGN). We also

⁶http://www.getty.edu/research/conducting_research/vocabularies/ulan

⁷<http://dbpedia.org>

⁸http://www.getty.edu/research/conducting_research/vocabularies/tgn

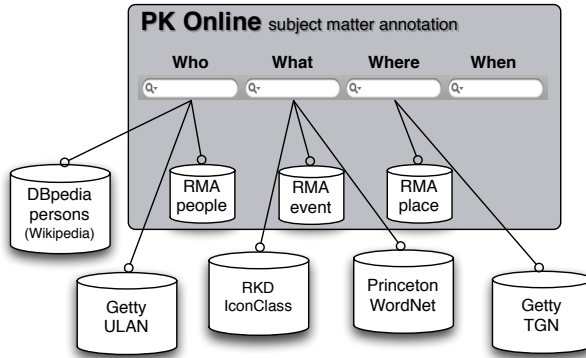


Figure 3.3: Prototype setup of the subject matter annotation fields and the multiple thesauri used for each field. The initial prototype contained two extra fields (iconography and event) which were later merged into a single *What* field. As in the original annotation interface, the dates are entered into a free field.

added the RKD ICONCLASS thesaurus and, as a source for more general terms, W3C's RDF version (van Assem et al. 2006) of Princeton's WORDNET⁹. Getty's Art and Architecture thesaurus could be another source for generic concepts. We expected the combination of ICONCLASS and WORDNET to provide reasonable coverage, and thus chose not to add this thesaurus. An additional thesaurus with relevant historical events and a thesaurus dedicated to persons depicted on portraits was high on the project's wish list but were not available during this study.

3.5.2.2 Search functionality

As we decided to provide autocompletion suggestions to the user, the search algorithm should support fast prefix search. To support search within multiple thesauri the search algorithm of the prototype should be able to cope with the differences among the thesauri and allow different search strategies to be configured for each thesaurus. The interface will contain different annotation fields that should give access to different types of thesauri terms. For example, only locations should be suggested in the annotation field for depicted locations. To select terms of the right type some form of result filtering is, thus, required. In the RMA, Getty and DBPedia thesauri type information about the terms is available, whereas in WORDNET and ICONCLASS it is not directly indicated if terms are persons, loca-

⁹<http://www.w3.org/2006/03/wn/wn20>

tions or concepts. It is thus not straightforward to filter out the different terms from WORDNET and ICONCLASS.

3.5.2.3 Initial interface design

The project management suggested that the initial prototype interface should focus on the *Who*, *What*, *Where* and *When* of an object. Only the active fields of museum's annotation interface (5 out of 33): *iconography*, *person*, *event*, *place* and *date* were incorporated into the prototype. An extra field was required to add more general terms about *what* is depicted.

To allow more efficient search in the now much larger set of vocabularies, we decided not to use the current tool's interface where each term search results in a separate "screen" with matching results to choose from. Instead, we designed the prototype interface around annotation fields with "autocompletion". For the purpose of annotation this means that thesaurus terms can be suggested directly within the annotation field, allowing the user to quickly try alternatives and create an annotation with a minimal number of interaction steps. Part of this study is to investigate if autocompletion can support all three term search tasks. Since the project did not use a controlled vocabulary for the date field, nor wished to do so, we did not focus on the interface to enter dates and continued to use a free text annotation field.

3.6 Refinement of requirements and design decisions

Based on the initial requirements we developed a first prototype, which was refined through an iterative process of redesign and feedback by the project members. All five prototypes were web applications accessible in a standard web browser. This allowed the project members to use the prototype without supervision, in their own time and environment. The first prototype was demonstrated and discussed face-to-face with the project leader. The second and third prototypes were explored unsupervised by the project leader and lead cataloguer, who provided feedback by email and afterwards the findings were discussed face-to-face. The fourth prototype was used for an interactive walk-through by two professional cataloguers, who gave feedback during the walk-through. The final prototype was again explored unsupervised by the project leader and lead cataloguer, with feedback by email and face-to-face discussion. We provide some brief information about the implementation and illustrate the main functionality of the final prototype before explaining the refinements of the requirements and our design decisions.

3.6.1 Prototype implementation

The client-side interface is implemented in HTML and JavaScript. The interface widgets are developed on top of the Yahoo! User Interface (YUI) library¹⁰ and our configurable autocompletion widget, described in (Hildebrand et al. 2007). The server-side search algorithms are implemented in SWI-Prolog using its Web and Semantic Web libraries (Wielemaker et al. 2007). The search functionality is accessible through an API over HTTP. A request consists of one or more keywords and a parameter list with search, organisation and visualisation options. The server returns a structured response in JSON notation that contains matching thesaurus terms and the requested display information. All software is distributed as part of ClioPatria, the open source framework developed as part of the MultimediaN E-Culture project (Wielemaker et al. 2008).

We required RDF versions of all thesauri to be able to use them in our server middleware. The Getty and the Rijksmuseum thesauri were already converted to RDF within the MultimediaN E-Culture project (Schreiber et al. 2008). Some additional mappings were made to match the SKOS standardisation. W3C's version of WORDNET and the DBpedia person data were already available in RDF. For ICONCLASS we used the RDF version developed by the STITCH project¹¹.

A screen shot of the annotation interface of the final prototype¹² is shown in Fig. 3.4. The interface has a two column layout. The left side includes an editable title, an image (if available) and a description of the current object. The right side includes the subject matter annotation fields. Each annotation field consists of a header with the name of the field and a brief description of the available terms. Below each header comes a text-input field and a list of the annotations that have been added. The annotation fields for *Who*, *What* and *Where* use autocompletion to suggest terms while the user is typing. An annotation is made by selecting one of the suggestions. An annotation can be removed by clicking the delete button (cross) on the right. All changes directly update the data stored in the back-end. Selecting the link labeled “done” brings the user back to an opening screen, where the annotation of a new object can be started. The link labelled “cancel” does the same and, in addition, removes the annotations already made.

3.6.2 Feedback

The iterative prototyping and the feedback provided by the project members identified a number of topics that we used to refine the requirements from the first phase. We highlight several issues that we believe are relevant to Semantic Web technologies in general. Figure 3.2 provides a chronological overview, listing the key feedback topics per prototype. Below we explain for each topic the relevance

¹⁰<http://developer.yahoo.com/yui>

¹¹<http://www.nwo.nl/catch/stitch>

¹²<http://e-culture.multimedien.nl/pk/annotate?uri=RP-P-0B-77.320>

The execution of Johan van Oldenbarnevelt

RP-P-OB-77.320

Print with a scene

update cancel

Who Historical persons

person

Oldenbarnevelt, Johan van x

What Iconclass (en), WordNet (en), events (nl)

(mythological) concept, object or event

beheading x

Where Name of place or region

geographical place

Den Haag x

When Date, year or period

enter date

done | cancel

Figure 3.4: Interface layout of the final prototype. In the left column an editable title field, image and editable description field. In the right column the four subject matter annotation fields. Image of the print used with permission, courtesy of the Rijksmuseum Amsterdam.

to Semantic Web technology, the practical problem encountered in our study, our interpretation in terms of the requirements and the design decision we made.

3.6.2.1 Term disambiguation

An important motivation for the use of unique identifiers (URIs) for terms on the Semantic Web is to solve ambiguity. Each URI only refers to, for example, one person, location or concept. For annotation the user has to choose which URI is the most appropriate for the current task. A label attached to a URI might not be sufficient to determine this — the ambiguity of the labels was why we needed unique identifiers in the first place. WORDNET, for example, contains many homonyms, that is, words that have different meanings. The RMA PEOPLE and ULAN thesauri contain many different terms with the same or a very similar name.

As in many Semantic Web applications, we thus have to decide what informa-



Figure 3.5: Descriptions of individuals in the *Who* field. Underneath the name a short biography is displayed. This contains the nationality, role/profession and birth/death date. Note that for the first person listed, only the profession is available in the data. The abbreviation RMA, shown to the right, indicates the thesaurus source.

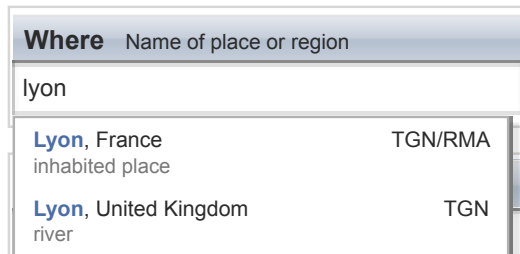


Figure 3.6: Descriptions of locations in the *Where* field. The name of the location is shown, in addition to the associated country. Underneath the name the place type is displayed. Again, the thesaurus source is shown to the right. The first term occurs in both RMA PLACES as well as Getty's TGN, the latter providing most of the additional information.

tion about each term should be used to present this term unambiguously. Even when only a single matching term is found, presenting such extra information can also help the user to confirm that this term is indeed the one they had in mind. During the discussion of the first prototype it became clear that for each field, different types of additional information should be shown in addition to the term labels. Figures 3.5 and 3.6 show, for example, the different information used for presenting suggestions for the *Who* and *Where* fields. As already mentioned in the requirements, the cataloguers often use the cryptic identifiers (called *notations*

in SKOS) of the ICONCLASS terms, and during the walk-throughs with the fourth prototype it became clear that for disambiguating persons the birth and death dates are very important clues. After the second prototype the cataloguers also requested the addition of provenance information: “*Always indicate where a term comes from. This could be important when deciding on a particular term.*” In later prototypes the cataloguers requested even more information to be shown, such as thumbnails with examples of other objects annotated with the term.

On the data level, no changes were required to support this request (although the thesauri do not always contain the necessary extra information for each term).

The required extensions to the underlying search functionality were more extensive. For the different fields and different thesauri we needed to include different types of information. Some of the required information was not covered by the standard SKOS model, and some information required some extra computation on the underlying data. To flexibly support thesaurus-specific configurations, we created a plug-in model on the server executing the generic search algorithm. Plug-ins were used to define the specific information that should be collected for each search result and to compose, for example, a short biography for an individual out of the nationality, role/profession and birth/death date.

In the user interface, the amount of extra information requested by the cataloguers resulted in more information than would naturally fit into the autocomplete suggestions list. We only put what we deemed the most important information in the primary list. The remaining items were shown in a secondary panel, displayed when a term is highlighted. This panel shows, where available, a short description, examples of other objects annotated with that term and the relevant part of the thesaurus hierarchy (see right hand side Fig. 3.7).

3.6.2.2 Equivalent terms

In a setting where multiple data resources are aggregated from the web it is very likely that the data contains duplicate terms. In this specific context this means that equivalent terms are found in multiple thesauri. When project members tried out the autocomplete in the first prototype, having the same label occurring multiple times was found to be very confusing. We also observed that the presentation of alternative labels for equivalent terms took up valuable real-estate in the autocomplete result list. We decided that the search results should contain only a single suggestion for each set of equivalent terms. To present the search results from multiple sources in such a way it is thus very important that equivalent terms are aligned.

On the Semantic Web, this means we have to determine if different URIs refer to the same term. For ontologies and vocabularies this is known as an alignment problem. In our prototype, alignments between the equivalent terms within and across thesauri were required to prevent duplicate results in the interface. Note

that alignments are normally used to extend the number of search results by allowing results indexed with terms from one vocabulary to be found with terms from another vocabulary. In contrast, we need the alignments to reduce the number of duplicate search results.

The original RMA thesauri already contained multiple terms for the same person, location or event, with the alignment relations between them. The first prototype did not use these alignments. We corrected this oversight in the next version. We also had to create alignments between the individuals in RMA PEOPLE and Getty's ULAN and the places in RMA PLACES and Getty's TGN. Our relatively simple mapping tool only aligned identical terms (using `skos:exactMatch` relations), ignoring potential broader or narrower relationships across thesauri.

In the underlying search functionality, we initially extended the search algorithm to use the alignment relations to filter out duplicate terms. On the interface level, however, this yielded unexpected results. Since for each suggested term extra information is displayed (e.g. biographical information, short descriptions, alternative terms), removal of duplicates also resulted in the loss of such extra associated information. The project members had several requests and questions concerning this when confronted with the next prototype: *“What happened to the terms left out? We can't see all the names. [...] Some terms from the RMA thesauri have limited additional information, is it possible to enrich this with information from equivalent terms from other thesauri?”*

To address this, we again modified the underlying search algorithm, but now to use the equivalence relations to collect, for each result, all relevant information on its equivalent terms. This solution allowed the user interface to present a set of equivalent terms as a single result, but also to use the extra information available from all the thesauri. We also wanted the interface to show only one preferred label for such a set of equivalent terms, but different thesauri often indicate different preferred labels for the same concept. We solved this ad-hoc for this specific project, because the members expressed a clear preference for the labels in the Rijksmuseum thesauri.

3.6.2.3 Complementary thesauri

Some data sources available on the Web can be naturally combined for a specific function. For example, in the prototype the different thesauri with individuals were all used in a single annotation field to describe depicted persons. For other sources such a combination may seem less obvious. In the initial prototype WORDNET, ICONCLASS and the RMA EVENT thesaurus were all accessible from separate annotation fields. After the cataloguers tested the second prototype it became clear that the difference between the WORDNET and ICONCLASS fields was unclear. *“The distinction between these two fields is not intuitive for the cataloguer. One term can occur in both the What and the Iconography fields. Perhaps it is easier*

for the cataloguer if the information is presented in one field, but with a distinction between the ICONCLASS and WORDNET terms.". Different thesauri may be complementary to each other for a specific task. In our prototype the advantage of combining complementary thesauri in a single annotation field is that alternative suggestions from different thesauri are given simultaneously for a single query.

The cataloguers indicated that the *What* field should provide access to ICONCLASS, WORDNET and the RMA EVENT thesaurus simultaneously. To realise this request, on the data level we added a superclass containing the terms of all three thesauri, since the search algorithm already supported subclass reasoning in the filtering of the search results. In the user interface, we configured a single autocompletion field to search all three thesauri.

3.6.2.4 Multilingualism

The Web contains sources in different languages. Ontologies and vocabularies may contain labels and descriptions in multiple languages. Limiting the data sources used in an application to a particular language may, however, rule out many useful terms. The RMA thesauri used in the prototype were all in Dutch, but we only managed to find additional sources in English. Even ICONCLASS, that was originally developed in the Netherlands, is not yet available in Dutch. Feedback on the third prototype indicated that: *"The different languages in the data could cause a problem. For example when searching in the What field it is not possible to search for siege and find an equivalent Dutch term from the RMA EVENT thesaurus."*

We decided not to change the data or the underlying search functionality, but to use the experiment to explore the practical implications of having both Dutch and English thesauri in a single annotation field. Given the expertise of our users, we only changed the user interface to briefly indicate in the description of the annotation field headers which language to use for which thesaurus (see the header of the *What* field in the top left of Figure 3.7 for an example).

3.6.2.5 Combining search with navigation

When searching for terms on the Semantic Web the user may not always know in advance what exactly she wants to find. Instead she might prefer to explore the available data to determine which term is most suited for her. For explorative tasks, keyword search can be a good starting point, but not always sufficient. When trying out the first prototype the cataloguers were pleased with the autocompletion functionality, as it allowed them to quickly find known terms from multiple thesauri. When using it to search in ICONCLASS, however, some functionality compared to the online IconClass browser was missing. The cataloguers often perform a global search to find a relatively generic term, which they use as a starting point

for further navigation along the hierarchical structure to find a more specific term. This helps them to find suitable terms without knowing the exact term in advance.

To support a similar type of functionality we decided to, on the data level, use the `skos:broader` relation to model all hierarchical relations among terms. The RDF version of ICONCLASS already used this relation to model its concept hierarchy. For WORDNET, we mapped its lexical hypernym relation, and for TGN and RMA PLACES we mapped their geographical containment relation to `skos:broader`.

In the underlying search functionality, we created a separate API that provides, on request, the data for the partial hierarchy as well as data about the children when the hierarchy is further expanded. Using this, we extended the user interface to show parts of the hierarchy for each autocomplete suggestion. The screen shot in Fig. 3.7 shows the hierarchy for the highlighted suggestion in a secondary panel. The hierarchy contains the term itself, all its ancestors and the direct children. More descendants are available on request by further expanding the direct children.

3.6.2.6 Compound queries

When searching on the Semantic Web with a compound query, the query could match a single literal containing all these keywords, or match multiple literals of different related terms. In our study we were confronted with both these variants of compound queries.

First, during the walk-through of the fourth prototype we noticed that when searching for specific thesauri terms, such as the historical and religious events in ICONCLASS, a single keyword is often not enough. For example, there are 490 terms in ICONCLASS matching “Mary”. Adding an additional keyword can greatly reduce the number of search results, for example, when the query “Mary” is extended with the keyword “assumption” only 5 results are left. The ICONCLASS browser and the current annotation interface of the Rijksmuseum did not support queries consisting of multiple keywords. the cataloguers would, therefore, first search on a generic term and then navigate to the more specific term. Often the cataloguers, however, knew the exact term or at least multiple keywords contained in the term. In the search algorithm we added support to match multiple keywords within a single literal.

Second, in the *Where* field of the fifth prototype we extended the algorithm to also match multiple keywords against different locations and use the hierarchical relations to find the best term. This allows the user, for example, to add the name of the country to a query for a city name. At the data and interface level no additional support was required for compound queries.

The screenshot shows a search interface for the term "siege". On the left, there are two main sections: "Iconclass" and "WordNet".

Iconclass results:

- [45K21] **siege**
Society, Civilization, Culture
- [45K] **siege, position war**
Society, Civilization, Culture
- [94H] **last months of the siege and the fall of Troy**
Classical Mythology and Ancient History

WordNet results:

- siege** (beleaguering, besieging)
blockade
- siege of Orleans** (Orleans)
beleaguering
- Siege Perilous**
seat

On the right, a secondary panel titled "siege, position war (more info)" provides a hierarchical structure:

- subject on about 26 artworks
- Images of prints
- [-] Society, Civilization, Culture
 - [-] warfare; military affairs
 - [-] **siege, position war**
 - [-] fortifications, military engineering
 - [-] attack ~ siege
 - [-] defence ~ siege
 - [-] capture of city (after the siege)

Figure 3.7: Annotation field with autocomplete. Suggestions are shown for the query “siege”. Results from both ICONCLASS and WORDNET are shown (left), each presented in a separate group. A secondary panel (right) shows more information for the highlighted term (“[45K] siege, position war”), including the hierarchical structure the term is part of. The hierarchy contains the term itself in bold, its ancestors and its direct children. Images of the prints used with permission, courtesy of the Rijksmuseum Amsterdam.

3.6.2.7 Sorting and grouping search results

When a search application provides many search results for a query, some form of organisation of these results is required to support the user with finding the right result. Ranking search results is an efficient technique, but can only be applied if data provides a good criterion to rank on. Initially, we sorted the search results on the frequency of use, showing those used most frequently for annotation as the first autocomplete suggestions. After trying the second prototype the cataloguers, however, indicated that: *“alphabetical sorting would be better for individuals and events”*. During the discussion they indicated that alphabetical sorting helps them to quickly scan a list of names.

Alphabetical sorting is useful where there is a relatively large number of results and the cataloguers have to find the right term among them, or to determine that a term does not exist. In these tasks a frequency ranking is not helpful as it does not give any insight about where a term can be found in the list. A potential problem with alphabetical sorting is that the format of the labels can vary among terms. For example, in the RMA PEOPLE thesaurus the preferred label consists of the forename and then the surname, while it is reversed in ULAN. We decided not to change the original names in the thesauri and accept the effects on the sorting strategy.

In the *Who* field it is common to have a large number of search results, as there are many historical individuals with a similar name. At the same time, there are still many individuals not contained in any of the thesauri. To support the cataloguer in finding terms from a long list, or determining that a term is missing, we used alphabetical sorting in the *Who* field. In the *Where* field the opposite was the case. The two place thesauri together provide high coverage for the annotation task at the Rijksmuseum, and there are few duplicate place names¹³. We thus ranked the places according to their frequency of use.

In the *What* field we needed three different sorting strategies. The cataloguers were used to navigating the ICONCLASS hierarchy starting from the highest matching term. We thus decided for ICONCLASS to show the terms highest in the hierarchy first. For WORDNET, we used the frequency counts stored in this thesaurus, which indicate the relative frequency of use among homonyms in English. For the events in the Rijksmuseum thesaurus we used alphabetical sorting.

To support the different sorting strategies we used the same plug-in mechanism as for the result visualisation. The plug-ins define which information should be used for sorting, which is then used by the algorithm to sort the terms. Because all the sorting is done by the underlying search algorithm, no further changes were needed in the user interface implementation.

Grouping similar types of results is another organisation strategy that is often applied to search results. Grouping has the advantage of displaying a wider variety of choices in the same screen real estate. In the prototype the cataloguers wanted to search simultaneously on multiple complementary thesauri in the *What* field and at the same time keep a clear distinction between the terms from each thesaurus. We decided to create the distinction between the thesauri terms by visually grouping the search results coming from the same thesaurus together.

On the data level, we needed to add the `skos:inScheme` property to each term to explicitly specify to which thesaurus it belongs. In the underlying search functionality, we extended the search algorithm and API with the option to group results by any property. In the user interface the different groups were realised by

¹³Note that the North America section of TGN contains many places with the same name. Because items from this part of the world are very rare in the Print Room, we decided to omit this part of TGN.

adding group headers to the result list, as shown in Figure 3.7. For each group a maximum of three suggestions were shown. Clicking the group header allowed the user to view all suggestions within that group.

3.7 Evaluation

The goal of the user experiment was to qualitatively evaluate the solutions proposed in the prototyping phase, by asking all cataloguers to use our prototype to annotate a number of “new” artworks (that is, artworks they have not annotated before) from their own area of expertise. To test our design for the user experiment we first performed a pilot with one cataloguer from the Rijksmuseum.

Our aim was to use the final prototype in a realistic environment. In our initial design the cataloguer would start describing the object using their own annotation interface and switch to the prototype for the subject matter annotation. During the pilot it became clear that only the editing of the title and description interacted with the subject matter annotation. The cataloguer would, for example, start with a description, make some annotations and then realise the description should be changed. The cataloguer also used the autocompletion suggestions to find the correct spelling for names she wanted to use in the title. Halfway during the pilot study the project leader of *Print Room Online* suggested that, for a realistic environment, it would suffice to enter also the title and description fields in the prototype, and not to use the museum’s own annotation interface during the experiment at all. The second half of the pilot was successfully continued using the simplified setup.

In the pilot it also became clear that some functionality that was supported by the prototype was not noticed by the cataloguer. We decided to add an online tutorial, at the start of the experiment, to acquaint the participants with the supported functionality. Below, we first describe the design of the experiment after which we present the general observations on the coverage of the date used and the use of autocompletion. Second, we give a qualitative evaluation of the design decisions made in the prototyping phase.

3.7.1 Experimental design

The participants of the experiment consisted of five professional cataloguers and two museum professionals who occasionally create annotations. Each participant took part in a one hour annotation session, annotating about six “new” objects. The objects were chosen by the project leader and matched the cataloguers expertise. Before the session the participant read an instruction manual¹⁴ and went through the interactive tutorial of about 15 minutes. After the annotation session

¹⁴http://e-culture.multimedien.nl/rma/prototype_manual.pdf

they filled in a questionnaire¹⁵. We asked a similar set of questions for each annotation field. The questionnaire focused on the topics for which we had provided solutions in the prototype phase. For example, to test term disambiguation we asked the participants: “How confident were you that the selected term was the term you intended?”, “Was it clear for each suggestion from which thesaurus it came?” and we asked them to rank the different types of additional information that were presented. Further questions were about the usefulness/annoyance of the (missing) alignments between equivalent terms, usefulness of autocompletion, the sorting and grouping strategies and the formulation of compound queries. In addition we asked the participants for demographic information and their experience with autocompletion.

Besides the questionnaires we used two other sources for evaluation. First, the observation of the annotation sessions. All sessions were screen captured and observed in real time by two researchers. The captured videos were annotated, focusing on the query construction and the result selection. Second, query logs showed precisely which characters were typed in the autocompletion fields and which annotations were made. The results that we present are based on the combined analysis of the questionnaires, observation notes, screen recordings and query logs.

3.7.2 Qualitative evaluation of design decisions

The key findings of the evaluation are summarised in Tab. 3.3. Below, we discuss the findings for each design problem.

3.7.2.1 Term disambiguation

In the questionnaire we asked the participants to rate the usefulness of the additional information for the different types of terms. As we only recorded the opinion of seven participants we do not make any statistical claims about the importance of particular types of information. We merely want to illustrate that cataloguers prefer different types of auxiliary information for different types of terms. For individuals the short biography is always considered the most important type of information. In particular, we observed that the cataloguers used the birth and death dates to compare against the creation date of the print. The description and alternative spellings are also considered useful. For locations the place type, the hierarchy and the description are all considered important. Hierarchy information is also rated as very important for ICONCLASS, but not at all for WORDNET. For the latter, the description and synonyms are preferred.

During the prototyping the project leader and lead cataloguer both stressed the importance of the provenance information. Most participants, however, stated

¹⁵<http://e-culture.multimedien.nl/rma/questionnaire.pdf>

Design problem	Findings
Term disambiguation	Different types of terms require different types of additional information to disambiguate them.
Equivalent terms	Conservative alignment is important for annotation to remove duplicates while preventing false positives.
Complementary thesauri	Simultaneous search in complementary thesauri helps cataloguers to choose the most suited term as they can directly compare alternatives.
Multilingualism	Professional cataloguers can deal with sources in multiple languages. Autocompletion helps to deal with multilingualism, as queries in different languages can be tried quickly.
Combine search with navigation	Autocompletion helps in annotation, as users often have to extend a query and try multiple different queries. Autocompletion can be successfully combined with a partial hierarchy to support selection of more specific terms. For some cases it might be useful to have access to a full search/navigation interface.
Compound queries	To find specific thesaurus terms from a set of very similar terms the user should be able to specify a query with multiple keywords.
Sorting and grouping	Professional cataloguers prefer a transparent sorting method, such as alphabetical sorting. Depending on the type of terms a frequency ranking might help for less experienced cataloguers. Within a single annotation field it helps to distinguish terms with a clearly different type.

Table 3.3: Summary of the key findings of the user study with respect to the seven design problems discussed in Section 3.6 and 3.7.

in the questionnaire that they were not interested in seeing the provenance information. [P4]: *“I don’t care much about knowing where a term comes from. I just want the right term (=most specific).”*¹⁶. Our explanation for this discrepancy is that during the prototyping the provenance was crucial to get an idea about the added value of the additional different thesauri. The lead cataloguer was also very much interested in seeing which terms were missing from their own thesauri to assess their quality. For the actual annotation tasks, however, this turned out to be less important than we had expected. Furthermore, the cataloguers did not consider if the thesauri were from authoritative sources, as they counted on the

¹⁶Quotes from the participants have been anonymized and are indicated with P1 to P7).

project management to handle this.

3.7.2.2 Equivalent terms

Only three cataloguers reported to have seen duplicate terms from different thesauri. While two of them indicated they were not so disturbed by this, the third indicated: [P4]: *“I would be forced to check both suggestions to see which one is most suited for me – more work.”*. Most cataloguers also didn’t notice that some suggestions were, in fact, a combination of multiple aligned equivalent terms and they also indicated not to care about this. We got the impression that the cataloguers are not bothered with the occurrence of incidental duplicates, [P6]: *“understandable when multiple thesauri are used”*, but that systematic occurrence of duplicates would disturb their efficiency. Aligning equivalent terms is thus important, but does not need to be perfect: a few false negatives result in relatively harmless and occasional duplicates¹⁷.

3.7.2.3 Complementary thesauri

All cataloguers were pleased with the possibility to simultaneously search in the different thesauri of the *What* field. [P1]: *“Finding alternatives in WORDNET is a plus if ICONCLASS falls short.”*. WordNet was, in particular, useful to describe [P2]: *“concrete things”*. Due to the different perspectives between ICONCLASS, art-historic, and WORDNET, linguistic, we did not provide alignments between these two thesauri. In case similar suggestions were provided from both, the participants were instructed to decide per annotation which thesaurus was most appropriate. In practice, they tended to use ICONCLASS as the primary source: [P4]: *“I am tempted to select terms from IconClass and use WordNet as a backup simply because they are presented in this order.”*.

3.7.2.4 Multilingualism

As expected, the language variation among thesauri terms required the participants to do some extra work: [P3]: *“Sometimes a person name is translated from French to Dutch, Louis=Lodewijk. You have to know this.”*. In practice, the cataloguers sometimes had to try queries in multiple languages. In general, the translation itself caused few problems for the professionals. Only for the look-up of English terms from WORDNET and ICONCLASS they occasionally had to use a Dutch to English dictionary, which slowed down the process considerably. This is, however, also a problem in the current situation where project members have to search using English terms in the ICONCLASS web interface.

¹⁷False positive alignments have a more severe impact. If term A and B are falsely aligned, they will be presented as a single result, and the user will no longer be able to select one of them. In this scenario, therefore, a conservative alignment approach is best suited.

3.7.2.5 Combining search with navigation

All participants indicated in the questionnaire that autocompletion was “very useful”. [P6]: *“Autocompletion increases speed of working and consistent use of terms.”*. During the experiment we observed that participants often used multiple queries to find a term or to determine that a term does not occur in one of the thesauri. [P5]: *“You have to figure out yourself how it could already be stored in the thesaurus.”*. The feedback provided by the autocompletion suggestions helped the cataloguers to see that a) there are too many results to investigate, b) the results do not contain the intended term and c) there are no results at all. As autocompletion provides instant feedback for every character typed, the user can quickly switch between scanning the list of results, reducing or extending the query or creating a new query. An example of (a) is that P3 queried on “hendrik” and the system indicated that there were 778 matching individuals. This made her decide to continue typing to specialise the query to “hendrik IV”, which returned only 5 individuals. To find a Dutch historical individual, P5 quickly tried different spelling variations of the name *dijck*, *dyck* and *dijk*, using the suggestions to quickly determine that the right term was not found (b) or that there were no results at all (c).

The hierarchical structure that was shown for the autocompletion suggestions was used several times to select a more specific term. When P5 investigated the suggestion “peace negotiations” from ICONCLASS the hierarchy contained “signing of peace treaty, concluding the peace” and this more specific term was selected. Presenting only the partial hierarchy was sufficient to support this interaction. The participants, however, also indicated they wanted to navigate the full hierarchy as provided in the ICONCLASS web interface: [P1]: *“I missed the overview in the full list with suggestions.”*. [P3]: *“I would like to see the whole hierarchy, but that could also be because I am used to it.”*.

3.7.2.6 Compound queries

In the questionnaire all cataloguers indicated that the support for compound queries was very useful. A query consisting of multiple keywords allowed them to quickly find specific known terms. This was in particular useful to find specific terms from ICONCLASS, which contains many similar terms about the same topic. [P3]: *“The Online IconClass browser does not support compound queries requiring more time to find the right term. Now you can find a specific term, such as a biblical event in one go. Very nice!”*.

3.7.2.7 Sorting and grouping search results

The five professional cataloguers that participated in the study carefully investigated the search results before selecting a term, whereas the two less experienced

participants tended to select the first appropriate term. In cases with more than one search result, the professionals scanned the entire list for possibly better candidates. The ranking of the search results is, therefore, for them not so important. [P2]: “*Ranking is not so important as I can scroll through the list.*”. In fact, the cataloguers prefer the search results to be sorted in a way that is understandable. [P5]: “*Alphabetical sorting is what I expect for a name list.*”. The alphabetical sorting of the search results that we used for the *Who* field helped the user, as it makes it clear where to look for a term and when to stop looking. Alphabetical sorting also helps the user when not all results are directly shown, as it is clear which results can be expected if more results are requested. The non-alphabetical sorting methods we used for the other annotation fields were judged differently by the participants. They were not particularly liked or disliked.

The participants indicated that some type of grouping of the search results was perceived as useful. In particular, the separate group for events was considered useful: [P2]: “*Clearly separates the events, very useful when searching.*”. The distinction between terms from ICONCLASS and WORDNET was not considered intuitive by all participants. There is some overlap between ICONCLASS and WORDNET, and presenting these similar terms in different groups was often found confusing. As indicated in the previous observation, seeing the source of the term is not so important to the cataloguer. The positive feedback on the separate event group could be an indication that a grouping by different types of terms is more suitable. This would require, first, to filter out the persons and locations in ICONCLASS and WORDNET use these in the *Who* and *Where* fields. Second, the remaining concepts in ICONCLASS and WORDNET should then be further classified in a fashion that is logical for the cataloguers, for example, by distinguishing named events from generic objects.

	Who	What	Where	Total
Terms used	14	65	16	94
Terms not found #	9	9	1	19
Terms not found %	41	14	6	20

Table 3.4: Per annotation field the number of terms found and not found during the user experiment.

3.7.3 Term coverage

Table 3.4 shows the number of terms found and not found in the user experiment per annotation field. These numbers give some insight into the type of terms the

	Who	What	Where	Total
Only RMA	13	1	3	17
Only external	0	64	4	68
RMA & external	1	0	9	10
Total	14	65	16	94

Table 3.5: Breakdown per thesaurus of the terms found in the experiment.

cataloguers used during the experiment. Most annotations were added in the *What* field. The weak coverage of the thesauri for depicted people is still present. Table 3.5 shows a breakdown of the terms found into those found only in an RMA thesaurus, in one of the external thesauri or in both RMA and an external thesaurus. The two external thesauri with individuals (ULAN and DBPedia) did not provide terms useful in this annotation task. For the geographical places the external thesaurus TGN provided additional coverage. Many depicted place names are in both RMA PLACES and TGN. This overlap can also be considered an improvement, as TGN provides additional information that is lacking in RMA PLACES. Participants found this additional information helpful in term disambiguation.

For the terms needed in the *what* field, the combination of WORDNET and ICONCLASS provided adequate coverage. The lack of an additional event thesaurus is illustrated by the fact that only one term was selected from the RMA EVENT thesaurus. During the experiment we also observed three cases in which an historical event was missing. Note that a relatively high number of terms was selected from WORDNET, an atypical source in this art-historical context. These were mainly general terms that were either not present in ICONCLASS, or only in a very specific biblical or mythological context. The addition of WORDNET was considered a big added value by most project members.

3.8 Conclusions and future work

We have derived requirements and design decisions to support a term search task on heterogeneous data in a real life setting. The study sets a first step to picture the landscape of multi-thesauri term search on the Web of Data, which enables further quantitative studies to be performed on thesaurus based annotation.

From the perspective of the museum staff, the experiment was successful. They appreciated the integration of the collection data, internal thesauri and external thesauri into a single tool, the autocompletion search functionality and the extra information that is displayed for each matching term. They also realise, however, that deploying a similar “open” annotation tool in their daily work-flow will require

several actions. First, all other tools in the tool chain should be adapted to allow the use of references to external thesaurus terms. This would require support for URIs instead of keywords or internal database keys. Second, it requires a dialogue with other cultural heritage institutions about the organisational aspects, such as the control and maintenance of the data. The museum will have to deal with complex questions about content, organisation and technical solutions. This project will be quite a challenge for an art institution. The Rijksmuseum is exploring the possibilities.

The extra thesauri deployed provided mainly a *quantitative* addition, by providing more terms of a particular type. In some cases, the addition was more of a *qualitative* nature, for example where the more general WORDNET terms were better suited to describe photographs than the specific terms from ICONCLASS. The integration of externally developed thesauri also proved beneficial because they provided more information about each term and were multilingual. In the prototype the coverage of persons and events was still limited, as suitable thesauri were not available. A source for depicted persons could be an internal thesaurus from the iconographic institute of the RKD containing Dutch historic figures depicted on portraits.

Some of the disadvantages of re-using data from heterogeneous thesauri can be overcome with careful data alignment and enrichment, suitable search functionality and domain specific configuration of the user interface.

On the data level, thesauri may partially overlap, and we use common vocabulary alignment methods to detect equivalent terms so we could avoid duplicates in the user interface. In our case a conservative alignment method was most suited as wrong alignments are harmful, because they remove possible candidates from the search results whereas a few duplicates in the interface are acceptable. To distinguish ambiguous terms from one another we present each term with additional associated information. For this, mappings to SKOS significantly help to address the heterogeneous structure of the various thesauri, as such mappings yield common properties for preferred and alternative labels, scope notes and broader/narrower relations across all thesauri. In addition, we need extensions to SKOS for representing common biographical and geographical properties. In future work we would like to partially align WORDNET and ICONCLASS and enrich these vocabularies to identify terms describing persons, location names and events.

We require search functionality that goes beyond the functionality of a standard SPARQL endpoint. Fast prefix string matching on RDF literals is required to support autocompletion, along with filtering of the matching results based on the type, source or other properties of the associated term. Queries with multiple keywords such as “Paris France” require additional support since they potentially match on terms with a label matching one keyword and a related term matching another keyword. This needs to be configurable by the client as it may be different depending on the task and thesauri used. The search engine also needs to have

configurable support to combine several search results related by `skos:exactMatch` relations into a single, coherent search result. Finally, to support hierarchical navigation in the autocompletion we also need efficient ways to retrieve the full path to the root of the hierarchy for each search result.

In future work we would like to improve the search algorithm by suggesting relevant results based on contextual information. One source of contextual information is other thesaurus terms already added to a museum object. Another source could be the free text in the title and the description, where automatic named entity extraction could provide context or be suggested as annotation terms. The contextual information could also be used to suggest annotation terms.

At the interface, different thesauri require different sorting, ranking and grouping strategies. For the visualisation, different types of information and different forms of presentation are required in order to best support the end user in disambiguating and selecting the most appropriate term. In addition to keyword search, some thesauri also require navigation interfaces to explore broader and related terms. In current work we are developing a method that allows the configuration of the visualisation and organisation by mapping domain specific semantics to an intermediate model.

Chapter 4

Case study II: Faceted browsing

In this chapter we investigate the exploration of multiple heterogeneous linked data sources through faceted browsing. We explore solutions for the required search functionality and presentation methods by the implementation of a prototype system. We focus on the use of semantics present in the data to extend traditional facet browsing functionality i) to support the formulation of structured queries on linked data and ii) to improve presentation methods to organise the large number of navigation paths. The study shows that generic faceted browsing functionality can be defined directly on top of the RDF triple model. The extra semantics available in the data provide a means to extend this functionality and to organise the navigation paths.

This chapter was published as “/facet: A Browser for Heterogeneous Semantic Web Repositories” in the Proceedings of the International Semantic Conference 2006 (Hildebrand et al. 2006) and was co-authored by Jacco van Ossensbruggen and Lynda Hardman.

4.1 Introduction

Facet browser interfaces provide a convenient and user-friendly way to navigate through a wide range of data collections. Originally demonstrated in the Flamenco system (Yee et al. 2003), facet browsing has also become popular in the Semantic Web community thanks to MUSEUMFINLAND (Hyvönen et al. 2005) and other systems (SIMILE 2005). An individual facet highlights one dimension of the underlying data. Often, the values of this dimension are hierarchically structured. By visualising and navigating this hierarchy in the user interface, the user is able to specify constraints on the items selected from the repository. To use an example from the art domain: by navigating the tree associated with a “location created” facet from the root “World”, via “Europe” to “Netherlands”, the results set is

constrained to contain only paintings that have been painted in the Netherlands. By combining constraints from multiple facets, a user is able to specify relatively complex queries through an intuitive Web navigation interface. All values of a dimension that would lead to an over-constrained query are dynamically removed from the interface, preventing the user from running into frustrating dead ends containing zero results.

We are working on repository exploration in the context of a national e-culture project (Schreiber et al. 2006). Our project's goals are similar to those of MUSEUMFINLAND, and aim at providing a syntactic and semantic interoperable portal to on-line collections of national museums. A major difference, however, is that we work with each museum's original metadata as much as possible. This means, for example, that we do not map all metadata relations of the various museums to a common set of ontological properties, nor do we map all metadata values to terms from a common thesaurus or ontology.

Initially, we experimented with traditional facet browsers, which assume a fixed set of facets to select and navigate through relatively homogeneous data. This, however, conflicts with our approach for the following reasons. First, our data set is too diverse to use a single set of facets: facets that make perfect sense for one type of object are typically inappropriate for other types. A related problem is that we cannot fix the facets at design time. When new data is added, the system should be able to add new facets at run time. This requires an extension of the facet paradigm to cater for objects of multiple types, to associate a set of appropriate facets to each type dynamically and to navigate and search larger sets of facets. Second, we use a rich and extensive set of art-related background knowledge. As a result, users expect to be able to base their selection not only on facets of museum artefacts, but also on facets from concepts from the background knowledge, such as artists and art styles. This requires two other extensions: one that allows users to switch the topic of interest, for example from artworks to art styles; and another one that allows selection of objects of one type based on the facets of another. For example, a set of artworks can be selected based on the properties (facets) of their creators.

This article discusses these extensions as they are realised in /facet, the browser of the project's demonstrator¹. The article is structured as follows. The next section introduces a scenario to illustrate the requirements for enhanced facet-browsing across multiple types. Section 4.3 discusses the requirements in detail and section 4.4 explains our design solutions in a Web-based interface. Section 4.5 discusses related work and open issues.

¹See (Schreiber et al. 2006) for a more detailed description and <http://e-culture.multimedien.nl/art/facet> for an on-line demo of /facet.

4.2 Example scenario

Throughout the chapter, we use examples from the art domain. The system itself, however, is domain independent and used on several other domains². The scenario is divided in two parts: the first part illustrates typical usage of facet browsers; the second part illustrates search tasks that go beyond the current state of the art and introduce new requirements for facet browsers.

Our protagonist is Simon, a high school student who recently visited the Dutch Kröller-Müller museum. The museum's collection features several works from Vincent van Gogh. Back at home, Simon has to write an essay on post-impressionism. He remembers seeing a particular post-impressionist painting but can no longer remember the name of the painter nor the title of the painting. The only thing he remembers is that the painting depicted a city street at night time. He uses a facet browser to restrict the search space step by step. He selects the current location of the painting (Kröller Müller), the art style of the painting (post-impressionist), its subject type (cityscape), and the subject time (night). He finds the painting he was looking for among the few results matching his constraints (Vincent van Gogh's "Cafe Terrace on the Place du Forum").

He now wants to further explore the work of Van Gogh, and selects this painter from the creator facet, and resets all previous selections. The interface displays the 56 paintings from Van Gogh that are in the repository. The facets now only contain values of the remaining paintings. For example, the create location facet instantly shows Simon that van Gogh made paintings in the Netherlands and in France, and how many in each country. Simon asks the system to group the results on create location and notices the significant difference in the color palette Van Gogh used in each country. By selecting "France" he zooms in further to explore potential differences on the city level.

In addition to the types of browsing possible in typical facet browsers, Simon also wants to explore works from painters born in the area of Arles. Unfortunately, the artworks in the repository have not been annotated with the birthplace of their creator. Simon uses multi type facet browsing and switches from searching on artworks to searching on persons. The interface now shows the facets available for persons, which include place of birth. Searching on Arles, he sees that four painters with unfamiliar names have been born here, but that the repository does not contain any of their works. Expanding his query by navigating up the place name hierarchy, he selects artists from the Provence-Alpes-Côte d'Azur, the region Arles is part of. He quickly discovers that Paul Cézanne, a contemporary of Van Gogh, was born in Aix-en-Provence in the same region.

Simon reaches his original goal by switching back from searching on persons to searching on artworks. Despite this switch, the interface allows him to *keep* his

²Demos on various domains are available at the /facet website
<http://slashfacet.semanticweb.org/>

constraint on Provence-Alpes-Côte d’Azur as a place of birth. It thus shows only artworks created by artists that were born in this region.

Backstage area . The experimentation environment in which */facet* was developed, contains sufficient data to cover the scenario above. It uses a triple store containing three different collections with artwork metadata: the collection of the Dutch National Museum of Ethnology³, the ARIA collection from the Rijksmuseum⁴, and Mark Harden’s Artchive collection⁵. RDF-versions of WORDNET⁶ and the Getty AAT, ULAN and TGN thesauri⁷ are also included. For the annotation schema, we use Dublin Core⁸ and VRA Core 3⁹.

In total, the store contains more than 10.8 million RDF triples. Artwork images are served directly from the websites of the museums involved. All the collection metadata has been converted to RDF, with some minimal alignment to fit the VRA Core 3 schema. In addition, explicit links were created from the art works to the Getty thesauri: literal names of painters and other artists were automatically converted to a URI of the ULAN entry; literal names of art styles and art materials to a URI of the AAT entry; and literal place names to a URI of the TGN entry. For example, in the scenario, some `vra:Works` have a `dc:creator` property referring to the painter `ulan:Person` Paul Cézanne, born in `tgn:Place` Aix-en-Provence in the `tgn:Region` of Provence-Alpes-Côte d’Azur. Additionally, some artworks have been manually annotated using concepts from the Getty thesauri and WORDNET. In the remainder of this chapter, we will use the following prefixes for the corresponding namespaces: `wn`, `aat`, `tgn`, `ulan`, `vp`, `dc` and `vra`.

4.3 Requirements for multi-type facet browsing

While the second half of the scenario sketches a seemingly simple means of accessing information, a number of issues have to be addressed before it can become a reality. Most facet browsers provide an interface to a single type of object. Including multiple types, however, leads to an explosion in the number of corresponding properties and thus the number of available facets. A facet browser still needs to be able to present instances of all the types and allow a user to select a particular type of interest. In addition, the relations between the types also need to be made

³<http://www.rmv.nl/>

⁴<http://www.rijksmuseum.nl/collectie/>, thanks to the Dutch CATCH/CHIP project (<http://chip-project.org/>) for allowing us to use their translation of the dataset to RDF.

⁵<http://www.artchive.com/>

⁶<http://www.w3.org/2001/sw/BestPractices/WNET/wn-conversion.html>

⁷http://www.getty.edu/research/conducting_research/vocabularies/, used with permission

⁸<http://dublincore.org/documents/dcq-rdf-xml/>

⁹<http://www.w3.org/2001/sw/BestPractices/MM/vra-conversion.html>

explicit and selectable by the user. To a large extent, the requirements we discuss are a direct consequence of these two key points.

4.3.1 Dynamically selecting facets

Fortunately, a first way to deal with the increased number of facets lies in the facet paradigm itself. One of the key aspects of all facet browsers is that, while constraining the dataset, all links that would lead to an over-constrained query are automatically removed from the interface, thus protecting the user against dead ends. As a consequence, if no instance in the current result set has a particular property, the facet associated with this property is removed from the interface. In our multiple type scenario, this means that if two types have no properties in common, the entire set of facets displayed is replaced when the user switches from one type to the other.

Facets in context of the `rdfs:subClassOf` hierarchy For most classes that have no subclasses, just hiding facets of properties that have no corresponding instances will result in an interface with a set of facets that intuitively belong to instances of that class¹⁰. For the super-classes, however, it is not immediately obvious what this “intuitive” set of facets is.

A first possibility is to associate with a specific class the *union* of the facets of its subclasses. This has the advantage that users can immediately start browsing, even if they have selected a class too high up in the hierarchy. By selecting a facet that only applies to instances of one of the subclasses, the result set is automatically constrained to instances of the intended class. A major drawback is that the number of facets displayed rapidly grows when moving up the class hierarchy, culminating in the complete set of all facets for `rdf:Resource`.

An alternative is to use the *intersection* of the facets of the `rdfs:subClassOf` hierarchy. This has the advantage that the user only sees facets that are common to all subclasses, and in practice these are, from the perspective of the super-class, often the most important ones. A drawback is that when moving up the hierarchy, one quickly reaches the point where the intersection becomes empty, leaving no facet to continue the search process. This forces the user to navigate down the class hierarchy to return to a usable facet interface.

A final possibility is to view the association of a set of facets with a certain class as an aspect of the personalisation of the system. While personalisation is one of the key aspects in our project, it is beyond the scope of this thesis.

Facets in context of the `rdfs:subPropertyOf` hierarchy As described above, the `rdfs:subClassOf` hierarchy helps to reduce the number of facets by only showing

¹⁰For simplicity, we ignore the question of whether or not to show sparsely populated properties, of which only a few instances have values.

facets that are relevant to a particular class. A similar argument applies to the `rdfs:subPropertyOf` hierarchy. On the one hand, the property hierarchy worsens the problem by introducing even more facets: in addition to the facets corresponding to the “leaf node” properties, their super-properties also become facet candidates. On the other hand, the property hierarchy also provides an opportunity for an interface to organise and navigate the property (and thus the facet) hierarchy, allowing the user to select facets as part of the interaction.

4.3.2 Search in addition to navigation

While the beauty of facet browsing lies in the ease of constructing queries by navigation, an often heard critique is that navigating deep tree structures is complex, in particular for users who are not expert in the domain modelled by the hierarchy. A second critique is that facet browsers become complex in applications with many facets and when users do not know what facets to use for their task. Multi-type facet browsing only makes this problem worse, by radically increasing the number of facets in the system. A search interface in addition to the navigation interface is thus required, and the two interaction styles should be well integrated and complement each other.

4.3.3 Creating multi-type queries

The example of selecting artworks created by artists born in a particular region requires a facet on a object of one type (`ulan:Person`) to be applied to find objects of another type (`vra:Work`). This is just one example of how such combinations can be used to exploit background knowledge in the selection process.

Using facets across types only makes sense if the objects involved are semantically related so the browser is required to know which relation to use. For the end user, the power of the facet interface lies in the ease of combining multiple facets to construct a complex query. This should be no different in a multi-type browser. So in addition a transparent interface needs to be available to easily constrain a dataset of one type, based on facets of another type.

4.3.4 Run-time facet specification

Manual definition of relevant facets and hard-coding them in a facet browser might be feasible for homogeneous data sets. This approach does not scale, however, to heterogeneous data sets, where typically each type of object has its own set of associated facets. Even for simple applications, the total number of facets might rapidly grow to several hundreds. Instead of hard-coding the facets in the browser software, some means is needed to externalise the facet definitions and make them configurable independently from the software. This simplifies maintenance and, by simply reloading a new configuration, allows adding and changing facets at

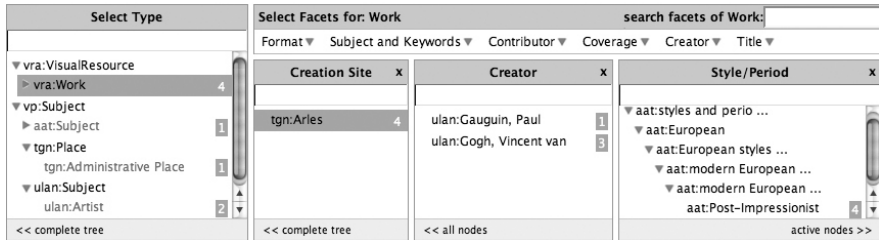


Figure 4.1: Snapshot of the `/facet` GUI. `vra:Work` has been selected in the type facet on the left, so only facets applicable to artworks are visible. Simon has restricted the results to have `tgn:Arles` as the place of creation. The interface shows Simon that the four matching paintings are created by either `ulan:Gauguin` or `ulan:Van Gogh` (picked from a flat list of artists), and that all four have `aat:post-impressionist` as the art style (shown in the context of its place in the AAT hierarchy).

runtime. The system also needs to be able to derive facet configurations from the dataset automatically. This allows the facet browser to run instantly on any dataset without prior manual configuration, while also allowing later manual refinement of the generated configuration. The latter is important, since it allows developers to tune the interface for specific end users, who might not be best served by a generic tool that gives access to all data.

4.3.5 Facet-dependent interfaces

A typical facet browser visualises the possible values of the facet either as a hierarchy or as a flat list. Related interfaces, such as those of `mSpace` (m.c. schraefel et al. 2005) and `Piggybank` (Huynh et al. 2005), have shown that some facets are better shown using a more specialised visualisation or interaction technique, such as geographical data displayed in an interactive map. To be able to tune a generic, multi-type facet browser to a tool for end-users that have a specific task in a specific domain, we require a mechanism for supporting visualisation and interaction plug-ins.

4.4 Functional design for multi-type facet browsing

We have explored the design consequences of these requirements in `/facet`. This section explains and motivates our design decisions in the prototype.

4.4.1 Browsing multiple object types

To support facet search for all object types, the /facet user interface needs a way to search for objects other than artworks¹¹. A natural and convenient way to integrate such functionality is by regarding the `rdf:type` property as “just” another facet. The facet applies to all objects and the values from its range are typically organised by the `rdfs:subClassOf` hierarchy, allowing navigation just as for any other facet. Since the semantics of this facet is derived directly from that of `rdfs:type`, by making a selection users indicate the type of object they are interested in. This constraint automatically selects which other facets are also active.

This is illustrated in Figure 4.1, which shows the upper half of the /facet interface. On the left is the type facet with a part of the domain’s class hierarchy. Simon has already selected artworks (e.g. objects of `rdf:type vra:Work`) and, as a result, only facets applicable to artworks are available from the facet bar at the top. Simon has expanded three of these from sub-menus of the facet bar: Creation Site, Creator and Style/Period. He selected “Arles” in the Creation Site facet. Apparently, the dataset contains only four objects of type `vra:Work` that were painted in Arles, indicated next to the selected type and location `tgn:Arles`. Simon has made no selections in the Creator and Style/Period facets, indicating that all four paintings are “post-impressionist” and that one painting is by Gauguin and three are by Van Gogh.

4.4.2 Semantic keyword search

In Figure 4.1, the art style’s full path in the AAT concept hierarchy is automatically unfolded because all paintings with “Arles” as the Creation Site share the same style “post-impressionist”. Showing the tree structure has the advantage that Simon could quickly select related art styles by simply navigating this hierarchy. This illustrates a well-known disadvantage of navigating complex tree structures: if Simon had instead started by selecting the art style, he would need to have known the AAT’s art style classification to navigate quickly to the style of his choice, which is hidden six levels deep in the hierarchy.

To overcome this problem, we added a keyword search box to each facet, with a dynamic suggestion facility, (b) in Figure 4.2. This allows Simon to find the style of his choice based on a simple keyword search. This interface dynamically starts suggesting possible keywords after Simon has typed a few characters. Note that the typical “no dead ends” style of facet browsing is retained: only keywords that produce actual results are suggested. Backstage, this means that in this case the suggested keywords are picked from the (labels of) concepts under the AAT “Style and Periods” subtree that are associated with art works in the current result set. In practice, the intended keyword is typically suggested after only a few keystrokes.

¹¹We still use artworks as the default type to give users a familiar interface when starting up the browser.

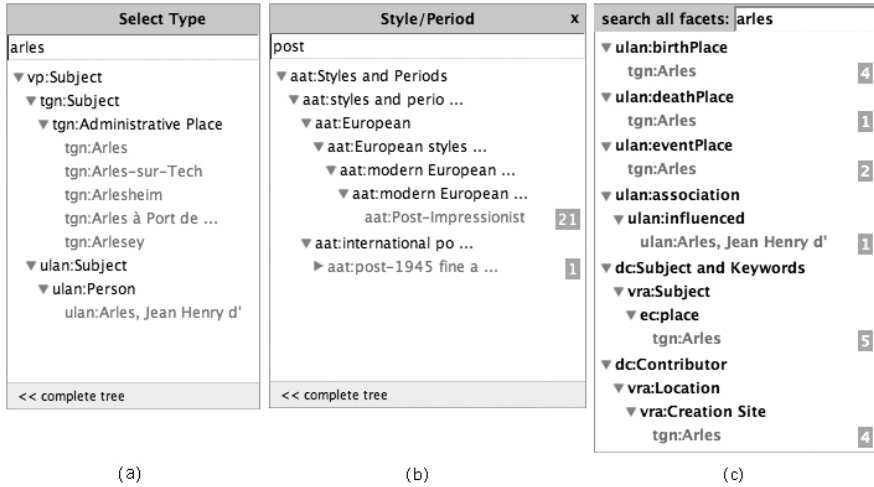


Figure 4.2: Three types of keyword suggestion and search. (a) show search on all instances, helping to select the right type. (b) shows search within a single facet, helping to move in complex facet hierarchies. (c) searches across all active facets, showing the user the different uses of the keyword “Arles” in different facets.

This makes the interaction often faster than navigating the tree, even for expert users who know the tree structure by heart.

The keyword search discussed above addresses the problem of navigation difficulties within the hierarchy of a single facet. Another problem could be picking the right facet in the first place. The keyword search box shown in (c) of Figure 4.2 addresses this problem. It provides the same search as the facet keyword search in (b), only across all facets of the selected type. For the figure, no type was selected and all facets have been searched. Arles is suggested as a TGN concept used in the facet corresponding to the `vra:location.creationSite` property (for paintings created in Arles), but also as the place used in the facet of the `vra:subject` property (for paintings that depict Arles), the birth and death place of Persons, etc. As a result, this search box can be used to find the right facet, but also to disambiguate keywords that have different meanings or are used in different ways.

A final problem can be that the user does not know the type to select to start with. This is addressed by adding also a keyword search box to the type facet, as shown in (a) in Figure 4.2. This searches over all literal properties of all instances and highlights matching instances and their types in the context of their location

The screenshot shows the 'MultimediaN E-Culture Facet Browsing' application. The interface is divided into several sections:

- Select Type:** A tree view showing the hierarchy: `vra:VisualResource` > `vra:Work` (123) > `vp:Subject` > `aat:Subject` (2) > `ulan:Subject` > `ulan:Person` (1).
- Select Facets for: Work:** A search bar and a list of facets: Contributor, Coverage, Creator, Format, Subject and Keywords, Title.
- Facet Selection:** Three facet panels are active:
 - Creator:** `ulan:Cézanne, Paul` (123)
 - Date:** 1867 (1), 1873 (1), 1875 (1), 1875-1876 (1), 1877-1878 (1)
 - Material.Medium:** `aat:oil paint` (108), `aat:water-base paint` (1)
- Constraints:**
 - `ulan:Person birthPlace = Provence-Côte d'Azur`
 - `ec:Work Creator = Cézanne, Paul`
- Results grouped by Creator:** `ulan:Cézanne, Paul` (123). Four artworks are shown:
 - `The Abduction` (Cézanne, Paul 100.0)
 - `Still Life with Apples` (Cézanne, Paul 100.0)
 - `Still Life with Flowe ...` (Cézanne, Paul 100.0)
 - `Apples and Oranges (P ...` (Cézanne, Paul 100.0)
- Timeline:** A horizontal axis from 1840 to 1900. A bar for 'Cézanne, Paul' spans from approximately 1860 to 1900. Below it, 'Impressionist' is marked from ~1865 to ~1885, and 'Post-Impressionist' is marked from ~1885 to ~1900.

Figure 4.3: Facet search on type `vra:Work`, but with a still active constraint on `ulan:Person` (birthplace Provence-Alpes-Côte d'Azur). Also note the timeline in the bottom, visualising multiple time-related facets. Images courtesy of Mark Harden, used with permission.

in the class hierarchy.

4.4.3 Specifying queries over multiple object types

We strive to support selection of facets from objects with different types in a transparent way, without further complicating the interface. In the example scenario, Simon searched on objects of `ulan:Person`, selecting `ulan:Provence-Alpes-Côte d'Azur` as the place of birth.

After making this selection, Simon can just switch back to searching on art-

works by selecting `vra:Work` in the type facet. In `/facet`, this would yield a page such as the one shown in Figure 4.3. Note that under the facets, the currently active constraints are shown, including the `ulan:Provence-Alpes-Côte d'Azur` constraint on the `ulan:birthPlace` facet of `ulan:Person`. For comparison, also a facet on `vra:Work` has been selected, in this case `ulan:Paul Cezanne` as the `dc:creator`.

To realise the example above, the facet browser needs to know the relation that can be used to connect a set of `vra:Works` with a set of `ulan:Persons` born in `ulan:Provence-Alpes-Côte d'Azur`. The current prototype searches for such properties at run time, and in this case finds the `dc:creator` property, as intended. To keep the user interface simple, we only support one property (that is, the first suitable candidate found by the system) to connect the different sets. Properties with the same domain and range can be used for normal facet browsing within a single type, but not for relating instances of different types.

4.4.4 Run-time facet specification

The facets that are shown in the interface can be configured in a separate file. Because a facet is defined in terms of RDF classes and properties, the configuration file itself is also in RDF, using a simple RDF vocabulary.

The vocabulary defines instances of `Facet` by three key properties. For example, the `birthday` facet is modelled by the `hierarchyTopConcept` and `hierarchyRelation` properties defining the hierarchy to be shown in the interface, by specifying the top of the tree (`tn:World`) and the `rdf:Property` used for the hierarchical relation (in this case `vp:parent`, the universal parent relation that is used across the Getty vocabularies). The `resourceProperty` defines how places are related to the painters, in this case by the `ulan:birthPlace` property.

Some other properties are optional and used to speed up or improve the user interface. The explicit definition of the type of objects the facet applies to, for example, makes it much more efficient to quickly switch to the right set of facets when users move from one type to the other. The `rdfs:label` property can be used to specify the name of the facet, which defaults to the label or name of the corresponding property.

To generate a first configuration file (that can later be hand edited), `/facet` analyses the dataset and generates a set of RDF facet definitions similar to the `birthPlace` facet example given above. For each property, the current algorithm search for a hierarchical relation in the set of related values to find the top concepts. If this relation is not found, or if the values are literals, it generates a facet with a flat list of values. For the scenario dataset, 22 hierarchical facets, 84 literal facets and 154 facets with a flat list of terms were found.

4.4.5 Facet-specific interface extensions

The values of a facet are typically presented in a list or a tree structure with textual labels. However, some structures are more easy to understand when presented differently. In particular, data which can be ordered linearly can be presented as points on an axis. Time, in particular, is a quantity that is often associated with objects, not only in the cultural heritage domain. It is useful to give a timeline representation of date data where this is appropriate. We have developed a timeline plug-in to visualise time-related facets (such as `dc:date` and its sub-properties).

Not only artworks have associated dates, but also related objects such as the life-span of the artist, Van Gogh, and the period associated with the `aat:post-impressionist` art style. Since the temporal information is related to the set of objects, this can be displayed together on a single timeline, as shown on the bottom of Figure 4.3.

A timeline interface could also be extended to not only show the temporal information, but also allow it to be used as part of the facet constraint mechanism. A similar facet dependent interface extension would be to relate geographical information together and display it on a two-dimensional spatial-axes interface such as a map.

4.5 Discussion and Related Work

Initial development of `/facet` has been heavily inspired by the facet interface of the MUSEUMFINLAND portal (Hyvönen et al. 2005). Where MUSEUMFINLAND is built on a strongly aligned dataset, we focus on supporting heterogeneous, loosely coupled collections with multiple thesauri from external sources. They provide mapping rules that hide the peculiarities of the underlying data model from the user interface. We have sacrificed this abstraction level and expose the underlying data model, but with the advantage that the software is independent of the data model and requires no manual configuration or mapping rules.

In comparison with `mSpace` (m.c. schraefel et al. 2005), `/facet` retains the original facet browsing principle to constrain a set of results. In a visually oriented domain such as ours, this leads to an intuitive interface where, after each step, users can see a set of images that reflect their choices: even users who do not know what “post-impressionist” paintings are, can immediately see from the results whether they like them or not. Also note that a heterogeneous dataset, such as ours, would lead to an m -dimensional space with $m > 250$, which would make the `mSpace` interface unusable. Alternatively, we could split up the data in multiple smaller `mSpaces`, but would then have no way of connecting them.

Unlike `/facet`, the Simile project’s Longwell (SIMILE 2005) facet browser requires to be configured for a specific dataset and its interface provides no solutions for dealing with large numbers of facets. An advantage of Longwell over `/facet` is

that the display of the results is fully configurable using Fresnel (Bizer et al. 2005).

While Noadster (Rutledge et al. 2005) is not specifically a facet browser, it is a generic RDF browser. Noadster applies concept lattices to cluster search results based on common properties. It clusters on any property, but ignores “stop properties” that occur too frequently. The resulting hierarchy forms a Table of Contents, with the original search results typically as leaf nodes, and common properties as branches. An advantage of Noadster is that its clustering prioritizes facets by placing those occurring more frequently in the matches higher in the tree. A disadvantage is the occasional “noisy” excess of properties in the clustering.

While we claim that /facet has some key advantages over the systems discussed above, the current prototype also suffers from some limitations. First, the algorithm for determining the facet configuration automatically needs further refinement. We now treat every RDF property as a potential facet. We then filter out many “schema level” properties from the RDF, RDFS and OWL namespace, and from our own internal namespaces (including the namespace we use for our facet specifications). It is still possible that a certain type of object will be associated with so many facets that the interface becomes hard to use. The techniques discussed in this chapter only partially address this problem, and they are highly dependent on the structure of the `rdfs:subpropertyOf` hierarchy. More research is needed to classify facets into a hierarchy that is optimised for usage in a user interface. In addition, we currently generate facets for all literal properties. On the one hand, this has the disadvantage that facets are generated for properties such as comments in RDFS, gloss entries in WORDNET and scope notes in the AAT. The values of such properties are unlikely to become useful for constraining the dataset. On the other hand, other properties with literal values, such as labels in RDFS or titles in Dublin Core, provide useful facets. More research is needed to provide heuristics for determining the type of literal values that are useful in the facet interface.

In the type hierarchy, we display all classes from the underlying domain, filtering out only the classes from the RDF(S) and OWL namespaces and the system’s internal namespaces. Still, this often leaves classes that are not helpful for most users. Examples include the abstract classes that characterise the top levels of many thesauri, such as the `vp:Subject`, `aat:Subject` and `aat:Concept` classes in our domain. The prototype’s current facet-mining algorithm is unable to deal with multiple hierarchies within a single facet. For example, many of our paintings have subject matter annotations referring to concepts from WORDNET. There are, however, several different relations that can be used to organise these into a hierarchy, such as hypernym/hyponym and holonym/meronym. For the user interface, it may be appropriate to merge the different hierarchies in a single tree or to keep them separate.

On the implementation side, the current prototype is developed directly on SWI-Prolog. The server side is a Prolog module built on top of the SWI-Prolog

Semantic Web Server (Wielemaker et al. 2003; Wielemaker 2005a). The client side is a standard Web browser that uses AJAX (Paulson 2005) for the dynamics of the suggestion interface. A drawback of this implementation is that users have to upload their data into /facet’s triple store. We are planning to make future versions of the browser using the SPARQL (W3C 2006) API, so that /facet can be used to browse any RDF repository that is served by a SPARQL compliant triple store. The large amount of RDFS and OWL-based reasoning at run time slowed down the system and gave the impression of an unresponsive interface. To address this, we reduced the amount of run-time reasoning by explicitly deriving the triples needed for calculating the result set when starting up the server so we can quickly traverse the expanded RDF-graph at run time. For example, we compute and add closures of transitive and inverse properties to the triple set at start up or when new data is added.

4.6 Conclusion and Future Work

We have discussed the requirements for a fully generic RDFS/OWL facet browser interface: automatic facet generation; support for multiple object types; cross-type selection so objects of one type can be selected using properties of another, semantically related, type; keyword search to complement hierarchical navigation; and supporting visualisation plug-ins for selected data types.

We developed the /facet Web interface to experiment with facet browsing in a highly heterogeneous semantic web environment. The current prototype meets the requirements discussed, it fulfils the described scenario in a cultural heritage domain and similar scenarios in other domains. A number of drawbacks remain, which we would like to address in future work. First, determining the facets automatically needs further refinement. Second, we are still fine tuning which classes from the class hierarchy we want to show and which facets we want to associate with the super-classes. Third, the prototype’s current facet-mining algorithm is unable to deal with multiple hierarchies within a single facet. Finally, we need to develop a version of /facet that is independent of a particular triple store implementation and runs on any SPARQL compliant triple store.

Chapter 5

Case study III: Semantic search

In this chapter we investigate, for a specific task, the use of graph structures in search functionality and result presentation to support users searching in multiple semantically-rich collections. The task this case study focusses on is finding artworks by domain experts. In a first experiment a number of use cases from domain experts are collected and the paths in the graph by which artworks can be found are analysed. A number of different types of paths are identified and their usefulness is qualitatively evaluated. In a second experiment we explore how the different path types can be used in a semantic search algorithm to support the intended search behaviour indicated by the experts. This study shows that users need highly interactive support to explore multiple search strategies and that different types of search functionality require different configurations of the graph search algorithm.

This chapter is submitted as a journal article titled “Searching in semantically-rich linked data: a case study in cultural heritage” and was co-authored by Jacco van Ossenbruggen, Lynda Hardman, Jan Wielemaker and Guus Schreiber.

5.1 Introduction

The term “semantic search” has been used to refer to a wide variety of search strategies. Some of these are based on logical inference, others on smart use of statistics, and yet others on natural language processing. Within the Semantic Web community, the focus of semantic search has been on improving the search process by explicit use of knowledge encoded in RDF/OWL. In previous work ([Hildebrand et al. 2007](#)), we surveyed several RDF and OWL-based search systems. The survey showed that different systems use different types of relations from the RDF data for different tasks. We have, however, little experimental evidence to support claims

about what types of relations in real world RDF data are relevant to which search tasks and how these relations are best deployed in a semantic search engine to support the user with a particular task. In this chapter we present a case study that investigates which relations between queries and search results are present in the data, and to what extent these relations are relevant for users with a specific search task.

Our case study makes use of the cultural heritage domain, where searching for artworks can be a time consuming task, even for domain experts. To satisfy their non-trivial information needs, they often need to formulate multiple queries and manually combine and integrate the various search results into a single coherent set of answers (Amin et al. 2008). We investigate how linked data can be used to support the user with the task of finding artworks. In particular, we focus on semantically-rich and heterogeneous linked data, where artworks from multiple collections have been annotated with terms from multiple structured and interlinked vocabularies.

To better understand how the relations in the data can be used to link queries to artworks we analyse three concrete use cases that are collected during interviews with domain experts. Our first finding is that the queries from these use cases can be successfully matched to literals in our data set, and that many of these literals are indeed directly or indirectly related to artworks. Our second finding is that, because of the heterogeneity of the data, there is a large number of different types of related terms that are potentially useful.

To deal with the heterogeneity of the data, we classify the relations into six path types. In a second round of interviews with the domain experts, we solicit their feedback on the relevance of these path types. Our key finding here is that while experts find the information resulting from all path types potentially relevant for their search process, if and how they would like to practically use it depends on many factors. This suggests that effective semantic search for experts in this domain can only be realised in a highly interactive search application.

To support different types of search strategies in an interactive search process we explore the applicability and configuration of the six path types in a graph search algorithm. For this purpose we collected the 25 queries most frequently submitted to a semantic search engine for cultural heritage. Using these queries we investigate the effect of different configurations of the graph search algorithm on the results. Based on our findings we discuss the implications for the design of an interactive semantic search application in the cultural heritage domain.

The chapter is organised as follows. In the next section we explain our study setup. In section 5.3 we describe the linked data set used in the study. In section 5.4 we explain the expert use cases and the selection of the test queries. Section 5.5 investigates how the queries in the use cases could be matched to literals in the data set and how these literals can be related to artworks. The large number of paths are abstracted to six path types. A qualitative evaluation of the relevance of these

path types for the expert use cases is presented in section 5.6. How to implement the path types in a semantic search algorithm is explored in section 5.7. We discuss the implications of our findings on the design of interactive search applications in section 5.8, here we also include references to related work. Finally, section 5.9 presents the conclusions.

5.2 Study setup

For this study we collected two sets of test queries used to find artworks in a large collection. For the first set, we collected the top 25 most popular queries from the logs of the online Europeana “ThoughtLab” search engine.¹ The queries cover a variety of different categories. In addition to this set of queries, we collected in-depth information about the use of text-based queries in expert use cases. We interviewed three domain experts from the Rijksmuseum Amsterdam about information needs they recently encountered in their own work. The experts were chosen to cover different areas of expertise, including a librarian assisting in external requests over the whole collection, a specialist in Japanese prints and an expert in middle age prints. Details of the collected test queries, the domain experts and their use cases are discussed in Section 5.4.

We use the collections of annotated artworks and controlled vocabularies used in the Europeana “ThoughtLab” as the data set for our experiments. Details about this data set are given in the next section.

We focus our study on the investigation of two research questions: (i) Which relations in linked data are useful in professional artwork search, and (ii) how can these relations be used in a semantic search application? To answer the first research question we analyse the different path types in the linked data and qualitatively evaluate these types in a user study with domain experts. Based on the findings of the first experiment, we perform a second experiment where we explore how the path types indicated useful by the domain experts can be exploited in a semantic search application.

First experiment — For the selected queries, we generate the paths of relations available in the data. We use a graph search algorithm to compute all paths up to path length 6. By analysing the paths we collect first insights on how the relations in the data can be used to relate artworks to the queries. In follow up interviews with the domain experts we collect qualitative feedback for different path types.

Second experiment — For the 25 queries from the search logs we analyse the results that can be found with the different path types. We explore how the algorithm should be tuned to approach the desired behaviour indicated by the domain experts.

¹<http://europeana.eu/portal/thought-lab.html>

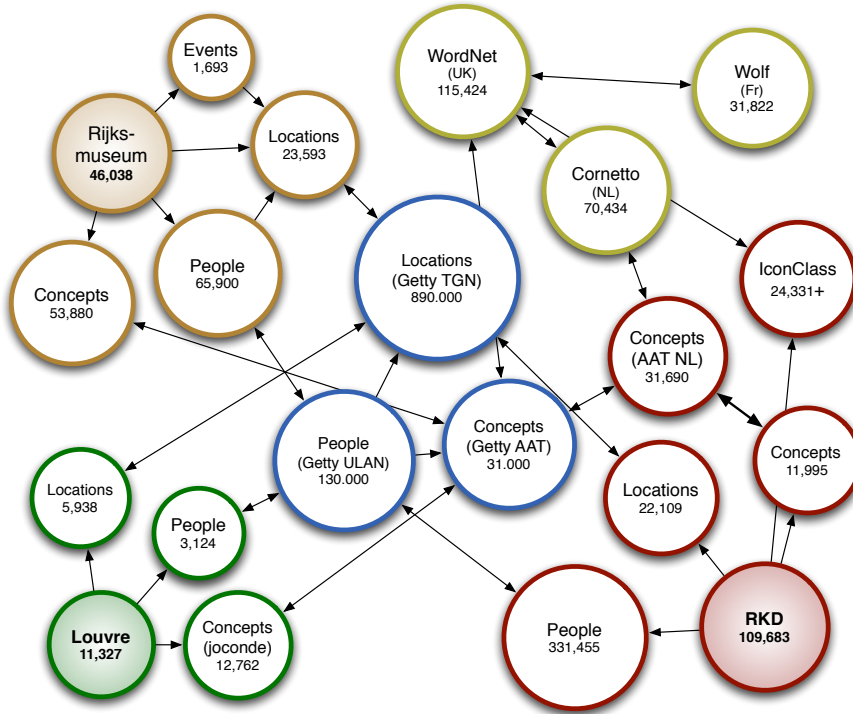


Figure 5.1: Datacloud of the Europeana “ThoughtLab” data set

5.3 Data set

We use an existing data set: the data from the Europeana “ThoughtLab”. Figure 5.1 provides an overview of the sources in the data. There are three types of data sources: *collections* describing works of art (the circles with a coloured fill the figure), *vocabularies* used to annotate the artworks (the remaining circles) and *alignments* among the vocabularies (the arrows between the vocabularies). Note that there are many different vocabularies, and only three different collections.

Collections — The artwork collections used are the internal collection database of the Rijksmuseum Amsterdam, the RKDimages database of the Netherlands Institute for Art History (RKD) and the Atlas database of the Musée du Louvre. For all three sources, only the artworks for which images are available on the Web were converted to RDF. The results are RDF descriptions of about

170,000 artworks.

The three institutions have described their artworks using different and rich metadata schemata. These original schemata have been translated directly to RDF, resulting in 469 different metadata properties on subjects of type `vra:Work`. All these properties have then been mapped to VRA, a specialisation of Dublin Core for visual resources. The properties have a wide variety of values. Some are short RDF literals, such as titles, measurements and dates. Longer RDF literals include descriptions, literature references and editorial notes. Some values point to terms defined in one of the controlled vocabularies, while others are structured (blank node) values. The latter are used to capture relations with arity > 2 , e.g. that an artwork was part of a collection, but only during a specific period. In these structured values, `rdf:value` is used to indicate the “main” value of the property, in the example above the name of the collection. In an informal context we can often ignore this use of blank nodes and simply consider the main value as a direct property of the artwork.

In some cases there is no clear choice between modelling a metadata property as a literal or as an object property. For example, some institutes use literal values for `dc:creator`, often with conventions on how to spell an artist’s name, while others fill the same field with a pointer to an entry in a predefined list of artists. In fact, any literal property value can be replaced by a term which has the same literal as an `rdfs:label`, and in the data set both modelling conventions are used. Some fields, however, have a such a wide range of values that it becomes virtually impossible to predefine in a vocabulary. Titles, descriptions and editorial notes, for example, are typically free text fields in most collections, and represented as literal RDF properties in the data set.

Vocabularies — All three institutions use and maintain their own in-house vocabularies, that typically describe people, locations, events and concepts. The Rijksmuseum and RKD also use concepts from the ICONCLASS² classification system, currently maintained by RKD. In addition, the data contains several external vocabularies. From Getty it includes the United List of Artist Names³ (ULAN), the Thesaurus of Geographic Names⁴ (TGN) and the Art and Architecture Thesaurus⁵ (AAT). Finally, three interlinked lexical sources, the W3C’s RDF version of Princeton’s WORDNET⁶, the Dutch lexical semantic database CORNETTO⁷ and the French WOLF version of WORDNET⁸ are included. All vocabularies are either directly modelled in SKOS or mapped to SKOS using `rdfs:subClassOf` and `rdfs:subPropertyOf`.

²<http://www.iconclass.nl>

³http://www.getty.edu/research/conducting_research/vocabularies/ulan

⁴http://www.getty.edu/research/conducting_research/vocabularies/tgn

⁵http://www.getty.edu/research/conducting_research/vocabularies/aat

⁶<http://www.w3.org/2006/03/wn/wn20>

⁷<http://www2.let.vu.nl/oz/cltl/cornetto>

⁸<http://alpage.inria.fr/~sagot/wolf-en.html>

Cross-vocabulary relations — The in-house vocabularies have been (partially) aligned with those from Getty, for example, vocabularies for persons are aligned with ULAN, those for locations with TGN and other concepts with AAT. ICONCLASS and AAT are also aligned with WordNet. For most alignments `skos:exactMatch` is used, some others use `owl:sameAs`. In the data set alignment relations may also occur within a single vocabulary, typically to state that two terms that were once thought as being distinct, are now to be considered equivalent. These alignments were already present in the original vocabulary data provided to the Europeana project. The majority of the alignments, however, relates terms from different vocabularies and are the result of automatic or manual vocabulary alignment efforts.

In addition to alignment relations, there are also vocabulary-specific relations that link terms from different vocabularies. For example, the locations from TGN are used as values for the birth places of persons in ULAN.

For more details on the conversion of the original data to RDF we refer to (Tordai et al. 2007; van Assem et al. 2004) and for details on the vocabularies alignment to (Tordai et al. 2009).

5.4 Collecting the query test set

Category	Query
Concept:	book, war
Location:	portugal, spain, rome, italy, greece, paris, poland, romania
Museum:	prado, louvre
Painting:	mona lisa
Style:	renaissance
Person:	klimt, van gogh, vermeer, rubens, goya, shakespeare, munch, da vinci, monet, renoir, hitler

Table 5.1: Top 25 queries from the January 2009 logs of the Europeana online “ThoughtLab” demonstrator and the inferred categories.

The 25 queries in Table 5.1 are taken from the January 2009 logs of the Europeana online “ThoughtLab” demonstrator. While we do not have exact demographic data, we assume most visitors are lay persons and not art experts. In total, the log of that month contained almost 13,500 session cookies, and 7,330 unique queries. After removal of the queries listed as examples on the website,

and those used by the project members for demonstrations, the remaining top 25 queries were selected. When we interpret these queries we infer that two refer to general concepts. All other queries refer to names: eight to location names, two to museum names, one to the name of a painting, 11 to person names and one to the name of a style/period.

These types of queries are comparable to those found in a more extensive study by Trant on the search log data from the Guggenheim Collection on-line (Trant 2006). We find the same focus on named entities as Trant, and, in particular, artists names are searched on the most. Only we find fewer references to styles and periods, and more to locations.

5.4.1 Expert use cases

The first set of interviews with the domain experts from the Rijksmuseum Amsterdam were centred around a search session within area of the expertise of the participant. We asked the participants to reproduce an information need that they had recently encountered, and perform the actions to satisfy this need with their own tools. We asked them to think aloud and explain their actions. For each interview, we recorded the search terms the experts entered and their motivation to choose these. The interviews and the search queries were performed in Dutch. In consultation with the participants we translated the topics and queries into English.

5.4.1.1 Use case: peddler

The first participant (P1) is a librarian of the Rijksmuseum's reading room. One of her tasks is to assist cultural heritage researchers in finding museum objects. In the interview she explains how she assisted a researcher with a study into the different ways "peddlers" (a kind of travelling merchant, in Dutch: "marskramers") have been depicted on historical prints and paintings. The initial task of the librarian is to collect evidence that the collection contains a variety of artworks depicting peddlers.

The librarian starts her session by searching for museum objects in the museum's collection management system, using the query **peddler**. She first searches in the title and second in the description field. These searches return only a few objects. To find more objects she tries a new query **street vendor**, which is a different type of travelling merchant. This search term also returns a few objects. She also tries **street salesman**. To get other ideas she turns to the library system to find books about the topic. Using again the query **peddler** she finds a book about the topic. She selects the book from the library and finds several prints depicting peddlers. As these prints are made by **Pieter Breughel**, she returns to the collection management system to find objects made by this artist. Although there are many artworks that do not depict peddlers, one of the artworks that

was displayed in the book is found. She concludes that there is sufficient material about the topic. She reports back to the researcher, expecting to return later for a more thorough investigation.

Because **peddler** was the query initially entered by the user, and a typical example of a vocabulary concept, we will use this query in the remainder of the article as the main query associated with this use case.

5.4.1.2 Use case: Fuji

The second participant (P2) is a cataloguer of the Rijksmuseum print room, specialised in Japanese prints. At the time of the experiment she was not performing her own research, so we together discussed a possible query topic: artworks by Ando Hiroshige that depict mountains. She initially explained that this topic was too broad to be realistic. After the search session, however, she explained that the session was typical for her search behavior.

Within the Rijksmuseum's internal database there are 163 prints from Ando Hiroshige. In the search session, the participant searches within this set. She starts by entering the query **mountain** in the title field. As there are only 3 artworks, she adds the description field. Ten more artworks are found. She states: "The chance is high that there are landscape paintings that contain mountains, but this has the disadvantage you might get artworks without mountains". She adds the search term **landscape** as a disjunctive query in the description field. Four additional artworks are found, from which one indeed contains a mountain. She recognises that this is the Japanese mountain **Fuji**, which she tries as her next search term. There are 11 artworks that depict this mountain. She also tries the related term **valley** in the description, which does not have any results. She explains that she could continue with this process for a while.

When asked for other methods to search, she explains that if you know that an artist has created artworks about a topic in a specific period, you can look for other artworks within this period. Or if you know when a volcano erupted, you can search for artworks that are created shortly after that date within the same region.

Most queries used in this use case are typical thesaurus concepts, which are represented in the data in a way similar to that of the "peddler" concept of the previous use case. The query **Fuji**, however, refers to a geographical name, and has different characteristics. Because we know from the Europeana logs that location names are frequently queried for, we will focus on this query as the main query associated with this use case.

5.4.1.3 Use case: Gregory

The third participant (P3) is also a cataloguer of the print room. She is specialised in prints from the Middle Ages. In addition to her work as a cataloguer,

she investigates a specific technique used for illustrations in Middle-Age books. The Rijksmuseum print collection also contains prints that were originally parts of books, and she tries to discover if any of these are made using this specific technique.

As the technique of her interest is very rare and not named or described yet, her main strategy is to query on topics she knows that are used in the books. In previous research, she discovered that the **Gregorian mass** is one of these topics. She uses this as a search term to search for prints in the Rijksmuseum collection. Several prints contain the search term in the title, but upon further study none of these are made using the technique. To find more prints she also tries **gregory** to find artworks depicting the pope “Gregory the Great” who was involved in this mass. This returns fewer than 20 artworks, which can be studied one by one. She also tries the search term **mass**. For this, many more artworks are found and she wants to further constrain this set by place and time. In previous research she discovered that the print technique is used in Germany between 1400 and 1500. She demonstrates how a different database supported her by allowing constraints between 1400 and 1500. She also mentions that she would like to search on all locations within Germany.

Again, most queries correspond to concepts. Only **gregory** refers to a person’s name, the other main type of query we found in the Europeana logs. We will therefore focus on this query when further discussing this use case.

5.5 Analysis of relations found

To find the relations between queries and potentially related artworks in the RDF data set, we apply a graph search algorithm (Wielemaker et al. 2008). To match queries to RDF literals, we use the algorithm’s default string matching technique based on Porter stemming (Porter 1980) and tokenization. The directional graph search traverses the graph from objects to subjects, but not the other way around: only symmetric properties and properties with an explicitly defined inverse are traversed in both directions. We set the maximum path length to 6, counted by the number of properties. In our experience path lengths above 6 become unrealistic to compute in reasonable time. For more details about the algorithm used, we refer to (Wielemaker et al. 2008).

5.5.1 Expert use cases: path analysis

We analyse the paths from query to artwork that can be found for the three queries described in the expert use cases. We define a path as a series of triples, where the object of the one is the subject of the following. The last object is the literal that matched with the query and the first subject is the artwork found. At path length 1, artworks are, thus, related by an RDF property with a literal value that matches

path length:	1		2		3		4		5		6	
peddler	46	47	2	104	1	128	5	2,106	137	14,189	260	33,909
fuji	15	15	2	3	0	0	2	48	1	1	15	1,514
gregorius	56	68	5	5	4	9	1	40	33	235	119	1,146
gregory	8	9	11	57	2	3	19	158	64	3,714	154	8,992

Table 5.2: For each query from the expert use cases the number of different paths and the artworks found these paths (#paths #artworks).

the query. At path length 2, the artworks are related to a vocabulary term that is in turn related to a literal matching the query. As we are interested in the different ways to find artworks and not how to find vocabulary resources, we consider all properties to find artworks and only one path for each unique vocabulary term.

Table 5.2 shows for each query and path length the total number of artworks and the total number of different paths needed to find these artworks (displayed as #path #artworks). At path length 1, for all queries only a small number of artworks are found and almost as many paths are required to find these. In other words, almost all artworks are found via a different literal. At path length 2, relatively few paths are required to find the artworks. In this case the different values by which the artworks are found are, thus, represented by a single vocabulary term.

At path length 6, more than 1,000 artworks are found for all queries. For the queries **peddler** and **gregory** over 1,000 results are already found for path lengths 4 and 5. For the query **Fuji**, however, only a small number of artworks is found at path lengths 3,4 and 5. For comparison we also generated the paths for the 25 queries from the search logs (not shown in Table 5.2). From the 25 queries, 16 are already related to more than 1,000 artworks at path length 4. The queries with generic concepts, e.g. “war” and “book”, well known persons, e.g. “van gogh”, and locations, e.g. “rome” and “paris”, have over 10,000 results at path length 4. The small number of results for the query **Fuji** at lengths 3 and 4 should thus be seen as an exception, caused by a limited amount of information available on the topic.

For the first expert use case (**peddler**) we describe in detail the different paths found at different path lengths and discuss our findings from the analysis of these paths. For the other two use cases we highlight the similarities and differences compared to the first use cases.

5.5.1.1 Use case: peddler

At path length 1 artworks are related by an RDF property with a literal value that matches the query. The query **peddler** matches with literals used as titles, descriptions and editorial notes of 47 artworks. The Rijksmuseum provides 15 of the artworks, while the other 32 are from RKD. With the exception of the title “The Peddler”, which occurs twice, all other literals are unique. An example path of length 1 is:

rma:SK-C-1346	dc:title	”The peddlers”
---------------	----------	----------------

At path length 2, the concept **peddler** in the RKD subject thesaurus is found via a label matching the query. It is used to describe the depicted subject of 104 artworks from the RKD collection. Note that almost all artworks found at length 1 had a different path, while the 104 artworks at length 2 are found by only two vocabulary terms (see the second column in Table 5.2). All artworks found by this path are from the RKD collection, as only these are described with the concept **peddler** from the RKD in-house thesaurus. An example path is:

rkd:68359	dc:subject	rkd:peddler
rkd:peddler	skos:prefLabel	”peddler”

One artwork is also found by a different path at length 2, because its title is modelled as a compound object with the title as the value of the `rdf:value` property.

At path length 3 there is only one path, resulting in 128 related artworks. These are described with the concept **salesman**, a more general concept of the RKD concept **peddler**. An example path is:

rkd:9429	dc:subject	rkd:salesman
rkd:salesman	skos:narrower	rkd:peddler
rkd:peddler	skos:prefLabel	”peddler”

At path length 4 there are five different paths, resulting in 2,106 artworks. Four of these paths contain vocabulary terms related to the concept **salesman** in the RKD thesaurus. One of these contains the more generic concept **professions**. The other two are more specific terms of **salesman**: **market.salesman** and **fish.salesman**. These concepts are thus siblings of **peddler**. The activity of **trade** is found by a `skos:related` property. An example path is:

rkd:60688	dc:subject	rkd:trade
rkd:trade	skos:related	rkd:salesman
rkd:salesman	skos:narrower	rkd:peddler
rkd:peddler	skos:prefLabel	"peddler"

The largest number of artefacts (1,772 out of 2,106) are found via the concept **basket** from the RKD thesaurus. This concept is found through an equivalent concept in CORNETTO WORDNET. The concept is a more generic term of a type of basket used by peddlers, in Dutch named a “mars”.

rkd:57252	dc:subject	rkd:basket
rkd:basket	skos:exactMatch	wn:basket
wn:basket	wn:hypernymOf	wn:mars
wn:mars	wn:gloss	"peddler basket"

At path length 5 the number of paths drastically increases and results in over 14,189 related artworks. All 137 paths use relations from the RKD subject thesaurus, of these, 117 lead to sibling concepts of **salesman**. In fact, we have now reached all professions in the RKD thesaurus. Eight terms are related to the activity of **trade**: six are *skos:related*, such as **scale** and **market stall**, one is the more specific concept **money trade** and another is more generic concept. Seven paths contain vocabulary terms related to the concept **basket**, including specific types of baskets, such as **fruit_basket**. The final five paths are concepts found via *skos:related* and *skos:broader* properties from **salesman** and **market_salesman**. As we start to drift off topic we do not discuss the more than 33,000 results at path length 6.

We conclude the analysis of this use case with three findings. First, some artworks can only be found via literal properties, because they have not been explicitly annotated with a vocabulary term that matches the query **peddler**. Searching with the vocabulary concept **peddler** as the value of a *dc:subject* property only finds artworks from the RKD collection, and none from the Rijksmuseum. All results from the Rijksmuseum collection are found via literal properties, such as *dc:title* and *dc:description*. This explains why expert P1, who is familiar with the collection, searched on these literal properties during the first interview.

Second, a large number of additional artworks were found at lengths 3 and above. The majority of these different paths involves some combination of thesaurus and alignment relations. The more than 14,000 artworks found at path length 5 is overwhelming. From our analysis it is unclear *a priori* which paths include results relevant to the search task.

Our third finding is that some of the paths contain concepts that are, to a large extent, similar to the alternative queries used by our expert user, but not exactly

the same, e.g. *salesman*. In addition, the paths contain many other vocabulary terms for which it is also not clear *a priori* if they are relevant to the search task.

5.5.1.2 Use case: Fuji

For the query *fuji* similar types of relations are used at path length 1 as in the first use case: titles, descriptions and editorial notes. In this case only 15 artworks are found, but again all results are found via different literals. Interpreting the literals showed that most were about Mount Fuji, and one was about the Fuji Photo Film Company.

At path length 2 there are two paths, resulting in only three artworks. Here we also find the two different interpretations of the query, represented by two vocabulary terms: *Mount Fuji* and *Fuji Photo Film Company*. At length 3 there are no artworks found.

At length 4 there are two paths, resulting in 48 artworks from the Louvre collection. Both paths include the term *Fuji-san* from the Joconde thesaurus. One path leads to two artworks depicting mountains via the concept *mountain*, which is related by two *skos:broader* relations to *Fuji-san*. Another path contains the sibling concept of *Fuji*, the volcano *Vesuvio*. At path length 5 another sibling is found, in this case an additional step is required as the path goes via the term *the Alps*.

At path length 6, 15 paths are found, resulting in 1,514 artworks. The majority of the paths (13), contain geographical concepts from the JOCONDE thesaurus. 453 artworks from the RKD collection are found. For these artworks the concept *mountain* from the RKD thesaurus is used as a value for the *dc:subject* property. This concept is found through an alignment with WORDNET, where it is related to *Fuji* by three *wn:hypernymOf* relations. Via a similar path, artworks from the RKD collection are found that depict cherry trees. In WORDNET, *fuji* is also defined as a specific type of cherry, and through *wn:hypernymOf* relations the generic concept of *cherry tree* is found.

For the query *fuji*, the number of artworks found is considerably less than for the query *peddler*. Most other findings are, however, similar. We do not find all artworks depicting Mount Fuji by looking only at artworks with concept *Mount Fuji* as a *dc:subject*. At longer path lengths we find related vocabulary terms, including some related to the queries from the expert use case, e.g. *mountain*. Only on a few of these artworks mount Fuji is depicted. An additional finding is that the query has multiple interpretations. At path length 1 these interpretations are implicit in the individual literals by which the artworks are found. At path length 2 the interpretations are explicitly represented by different vocabulary terms.

5.5.1.3 Use case: Gregory

We analyse the paths for the query in Latin (`gregorius`) and in English (`gregory`), as there are no alignments between the concepts in the different languages.

At path length 1 there are 68 artworks found for the query `gregorius`. The artworks are from RKD and the Rijksmuseum. Some of these are related to the “mass of gregorius”, others to “pope gregorius” and again others to topics unknown to us. For the query `Gregory` the matching artworks are about many different topics. For example, several artworks are found because the background literature used to describe the object is written by “J. Gregory”.

At path length 2 the query `Gregory` leads to nine paths with a vocabulary term from `ICONCLASS`, of which seven are events that include “Gregory the Great”. For the query `gregorius`, the matching vocabulary terms are other persons. Two persons are found because the query matches with their biography. Reading this biography we discover that one is a cousin of Gregory the Great.

At path lengths greater than 3, the vocabulary terms found for the query `gregorius` are linked to interpretations of the query that are not related to Gregory the Great. A large variety of relations are used in these paths, such as the collection-specific properties `granted_privilege` and `assigned_to` relations between persons `assisted_by`, `teacher_of` and `sibling_of`. Other persons are found because they are born in a place with a matching name.

For the query `Gregory` similar types of paths are found. We also find at path length 5 relations from `WORDNET`. For example, 58 artworks are found because they depict a pope. `WORDNET` contains a `wn:hypernymOf` relation between `Gregory the Great` and the concept `pope`. For a different vocabulary term, a `wn:hypernymOf` relation leads to the concept `saint`, resulting in an overwhelming 2,849 artworks.

Again we conclude that relevant artworks are found by matches on literal properties as well as via vocabulary terms. At longer path lengths there are many different paths and artworks found. Multiple interpretations of the query all lead to more related results, whereas only the paths from one or a few interpretations are useful. Another finding is that vocabulary terms are found via labels as well as descriptions, e.g. a biography, where the relation to the query is implicitly captured in the text.

5.5.2 Abstraction of the paths

From the use cases above we conclude that a large number of artworks can be found via many different paths. The large number of resulting artworks makes evaluation based on a domain expert scoring the relevance of each artwork unrealistic. Even the number of different paths is too large to be scored individually, also because the semantics of many of the longer paths and the subtle differences between them are hard to express in natural language or by some other understandable means.

We therefore look for frequently recurring types in the paths we found, and see if we can use these path types to classify all relations into a small number of abstractions.

5.5.2.1 Metadata properties

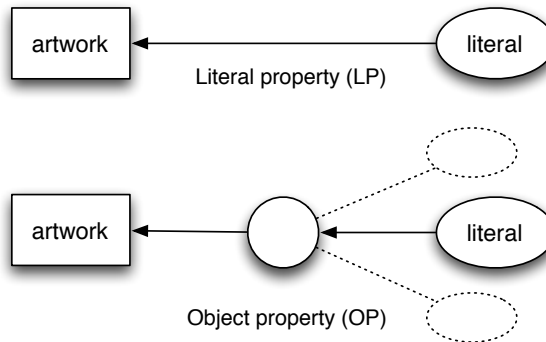


Figure 5.2: **Metadata paths:** Two types of paths between a literal and an artwork. The arrows are shown in the search direction, from object to subject.

The paths found in the three use cases show a clear distinction between those that directly use the metadata properties on the artworks, and longer paths that include relations between terms. With metadata properties we refer both to the paths of length 1, where the query is matched to the value of a literal property of an artwork and of length 2, where the query matches a label of a vocabulary term. These two types of paths are represented in Figure 5.2. The path type labelled *literal property* represents all paths with a direct relation between the matching RDF literal and an artwork via an RDF property. In the three use cases artworks were found using RDF properties for titles, descriptions and notes. Note the arrows are shown in the search direction, thus from object to subject.

The path type labelled *object property* represents all paths where the matching literal and artwork are connected through a single resource. In the three use cases, artworks were found by vocabulary terms, such as persons, locations, events, domain specific concepts and collection names. These terms were related to the artworks using high level properties, such as `dc:subject`, as well as collection specific properties, such as `granted_privilege`. Some artworks were found by an object with a matching literal from the `rdf:value` property.

5.5.2.2 Relations between vocabulary terms

The relations between the terms used in the paths longer than 2 follow from different schemata in the data set. The vocabularies are either directly modelled in SKOS or they have their own schema, which is mapped to SKOS. At an abstract level the relations between vocabulary terms can, thus, be defined in SKOS using the hierarchical relations, `skos:broader`, `skos:narrower`, the associative relation `skos:related` and alignments, such as `skos:exactMatch`. Our data set only contains equivalence alignments. We describe how these relations create different path types to find artworks.

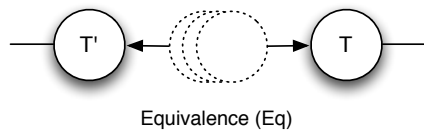


Figure 5.3: **Equivalence**: Vocabulary terms defined as equivalent.

The path type in Figure 5.3 labelled *Equivalence* represents a path that aligns two equivalent terms. This path can be used both directions, as the equivalence relation is symmetric. Equivalence between two vocabulary terms can be defined directly, for example, in the first use case the concept *basket* from the in-house thesaurus of RKD was found via an equivalence alignment with WORDNET. An equivalence alignment can also cover multiple terms.

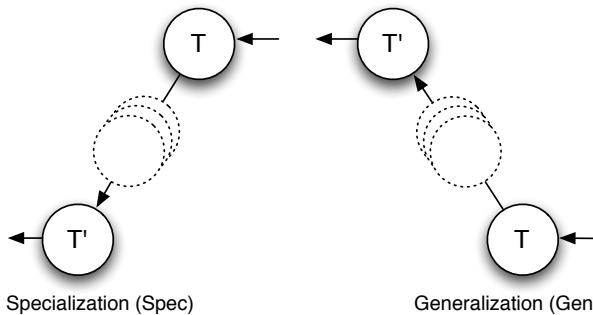


Figure 5.4: **Hierarchical**: Two types of hierarchical paths between vocabulary terms: specialisation and generalisation.

Figure 5.4 shows two types of paths to connect vocabulary terms by hierarchical relations. The path type labelled *specialisation* defines the relation by which a more specific (or narrower) term is found. The path type labelled *generalisation* defines the opposite relation by which a more generic (or broader) term is found. For example, in the “peddler” use case the concept *salesman* was found as generalisation of the concept *peddler*. In WORDNET the concept *mountain* was found as a generalisation of *Fuji* via three relations. By combining the a generalisation and a specialisation a sibling term can be found. For example, in the “peddler” use case the concept *fish_salesman* was found as a sibling of the concept *peddler*.

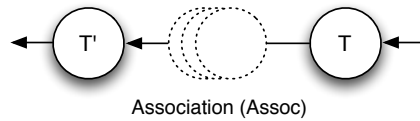


Figure 5.5: **Association**: Vocabulary terms associated by one or more relations.

The path type in Figure 5.5 labelled *Association* represents a path between two vocabulary terms connected by an associative relation. By this path type an associated term can be found. For example, in the first use case the concept the activity *trade* was found via an association to *salesman*. A single path can contain multiple association relations.

The four types of paths between vocabulary terms provide a means to extend the object property path, replacing the single node with one or more related terms. Inclusion of the different path types has different effects on the relation between the artwork and the query. In the following two sections we investigate the role of the different path types in the search process, by (i) a qualitative evaluation with the domain experts and (ii) a qualitative analysis of the results found.

5.6 Qualitative evaluation of path types

Having identified the six path types through analysis of the data when then evaluated to what extent the domain experts deem the information found by these path types relevant for their search activities. We would like to evaluate the potential relevance of the information itself, so we need to avoid that the experts’ feedback is influenced by RDF modelling flaws in the RDF data set, bugs in the search engine or confusing elements in the user interface of the Europeana “ThoughtLab”. To achieve this, we manually select web pages on the websites of the organisations that originally provided the data before it was converted to RDF, where each web

page shows some of the information directly resulting from applying one or more path types to the original query. In this way, we collect six sets of web pages for each query, where each set covers a specific type of information, and each set consists of examples from the different websites of the original content providers.

Each interview took place in the museum, using a museum computer to show the participants the various webpages. Each interview lasted one hour, was voice recorded and notes were taken by the conductor of the experiment and a second observer. After reading a short description of the goals of the study and the outline of the experiment, the participants were shown the six sets of pages. After seeing each set, they were asked to comment freely on what they had seen and were also asked five or six directed questions.

5.6.1 Searching free text fields

To get feedback on the role of the different metadata properties in the search process, we showed the experts a set of pages from the collection websites. We used pages that resulted from clicking on an individual search result associated to one of the queries used in the first interview. We selected results that were found by matches on different properties, e.g. a page showing a painting with “peddler” in the title, another page with a print with “peddler” in the description and a final one with “peddler” in the depicted subject field. We asked the experts in which fields they would search.

All experts commented that searches on a controlled field, in general, yield incomplete results. [P1]: *“When I only need one or two examples, the thesaurus-based search is best, because it will yield exactly the examples I need . . . but if I really need all depictions of a specific topic, I will also search on free text fields such as title and description, because cataloguers never add all relevant terms to the subject field.”* [P3]: *“It depends if you know how thorough a collection has been annotated. I know that the subject field has not been used for all objects of this collection.”*

Experts strongly prefer to search in fields using controlled vocabularies. In practice, they need to search on the literal properties of free text fields, because they know the annotations with terms from controlled vocabularies are often incomplete.

5.6.2 Search using controlled vocabularies

A key difference between searching object property fields and searching in literal property fields is that the vocabularies can be used to explicitly disambiguate the query if it has multiple interpretations (i.e. homonymy). We used web pages of the thesauri providers’ websites (i.e. from the Getty AAT, ULAN and TGN website, Princeton’s Wordnet site, the Joconde site of the French ministry of culture, the

RKD site for Iconclass). Each web page showed the search results for the query in the thesaurus, showing all different interpretations of the query.

All experts appreciated being able to see the different interpretations of the query explicitly. [P1]: *“Yes, if you know there are multiple meanings, you know in advance you can expect lots of noise in the search results”*. All experts also like the feature of thesaurus-based systems to search on only one specific meaning of a term. [P2]: *“If I search only on a thesaurus-controlled field, I would trust the list of search results more, and would not click every result to check why it is a match”*. Again, P3 uses this as a strategy to deal with errors in the data: [P3]: *“I would trust the results more than those of a free text field . . . but I would also search on the other interpretations, just in case the cataloguers have made a mistake”*.

We conclude that when a query has multiple interpretations, experts would not only like to be able to disambiguate and search using the intended interpretation, but also to be made aware of other possible interpretations.

5.6.3 Search using equivalences

The key feature of the equivalence path type is that it links terms from different thesauri that have the same or similar meaning. We again selected web pages from the thesauri web sites. We showed the participants that there are often multiple thesauri containing terms relevant to their query, and showed the experts that different thesauri encode different types of information related to their query. Again, we asked them how useful these different information elements would be in their search process.

All experts found the thesauri that provided name variants (e.g. Fuji versus Fujisan) and spelling variants (e.g. Fujisan versus Fuji-San) extremely useful, especially for person and location names. P2, looking at the name variants for Fuji listed in TGN: [P2]: *“Yes, this is very useful indeed. I would search on all variants listed to see if the results would yield additional results.”* Also the multilingual aspects were considered useful: [P1]: *“Having a domain-specific thesaurus in another language is very useful, as normal dictionaries do not always cover the jargon I am looking for. In addition, we are an internationally oriented museum, and the search interface software on our website is multilingual. But the content is often still in one language, which is confusing.”* [P3]: *“For the names of saints, even if you are using the Latin name, there are always subtle differences in different languages. So this is very useful.”*

We conclude that the experts consider equivalent relations across thesauri more generally useful: their applicability is not limited to specific cases. They seem most useful when the links to other thesauri bring in extra name or spelling variants, or translations to other languages.

5.6.4 Search using specialisation and generalisation

We selected web pages from the thesauri websites, with the term hierarchy fully expanded wherever possible, showing all broader terms to the top of hierarchy, and narrower terms wherever applicable.

When confronted with the full hierarchy experts responded positive [P3]: *“Being able to move down the hierarchy is useful for query refinement when you have too many results, or for broadening when you have too few”*. Most focussed on the narrower relations: [P1]: *“In general, the more specific you can be, the better.”* All three also indicated a possible use for the more generic terms, but only in cases where few results are found. P2 explained she might use more generic terms, but only in combination with other terms or restrictions. After seeing that “Fuji” was a narrower term of “mountain” in one thesaurus and of “Japan” in another: [P2]: *“I would do a new search by combining both “mountain” and “Japan””*.

The experts did not express a clear preference for the hierarchy from one thesaurus above the other. One, however, expressed the need for semantic integration of the different hierarchies. [P1]: *“In the ideal case, you should be able to use all different thesauri in a way that is fully integrated . . . but I do not know if that is possible . . . it should be done right though, otherwise I would not trust it.”*

Experts had mixed opinions about the relevance of “sibling” terms, e.g. terms with the same parent. [P3]: *“Maybe I would use these if the original query yields insufficient results . . . but even then, only if the terms are semantically close to the original query.”* [P1]: *“No. I would never use siblings in this case, as it would not give new results. Fuji is by far the most important feature in this region. If a print has been annotated with something else from this region, it would not depict Fuji . . . otherwise Fuji would have been added as an annotation as well.”*

We conclude that experts see potential in using hierarchical relations for search, but only if the other terms are semantically close, and even if this is the case, they would only use them in specific cases.

5.6.5 Search using associations

As this path type is also thesaurus related, we again showed similar thesaurus web pages, this time drawing the participant’s attention to the section on the page dedicated to horizontal relations. These relations might differ widely, ranging from general `skos:related` in the RKD thesaurus to specific `ulan:brotherof` in ULAN.

Experts were all positive about this type of relations. After seeing Fuji being related to “volcano” in WordNet: [P2]: *“People always refer to Fuji as a mountain, never as a volcano . . . but now I see this, I would also search for volcanoes in Japan . . . I would not have thought of this myself.”* One expert even uses these relations as part of a strategy to deal with errors in the data. [P3]: *“For artists, knowing family or apprenticeship relations is very important. If I know an artist has a brother, for example, I would always search on the brother too, because works*

are sometimes attributed to the brother by mistake as the names are very similar. Wrong attributions to students or teachers of an artist are also common.”

Again we conclude that associative relations can potentially yield relevant search results, but *how* to use them varies from case to case.

We conclude that experts prefer searching on a field that clearly indicates the relation to the artwork and for which the annotation terms are taken from a controlled vocabulary: this gives high precision, with results that can be quickly assessed on relevance. They mainly use literal properties when striving for completeness: searching in free text fields will yield additional results, but with lower precision and the results typically require time consuming inspection to assess their relevance. Equivalence paths seem useful, but mostly for name, spelling and language variants. In addition, experts consider the information provided by the hierarchical and association path types potentially useful in an interactive search application. When and how they like to use this information depends, however, on the context of the search task.

5.7 Exploring path type configurations

The analysis of the query types showed that a large number of paths relate artworks to queries. In the previous section, we described how experts value the information related to different types of paths, where we manually collected the related information. The question remains how these path types should be applied in a search application. From the analysis in Section 5.5 we conclude that using all path types for all queries results in too low precision. This is confirmed by the experts who indicated that they would use specific paths only in specific contexts.

To better understand how the different types of paths can be used to effectively support artwork search, more analysis is required. For this purpose we use the 25 queries that were most frequently submitted to the Europeana “ThoughtLab” search engine. For each query we compute the number of artworks found for the different path types. In addition, we compute the number of vocabulary terms found. You find an overview of this recall data in Table 5.3 (at the end of the chapter).

In the following sections we discuss the different columns of the table. We provide a qualitative analysis for a number of topics, and based on observations, discuss the precision of the retrieved artworks. To cover the full spectrum of relevant topics further research is required.

5.7.1 Alternative matches for free text fields

The experts prefer to search in fields with terms from controlled vocabularies. For the 25 search log queries we investigate if the vocabulary terms and the relations

between them provide an alternative to finding artworks by free text fields. For this purpose we first compute, for each query, the artworks found via a literal property (LP). Accordingly we compute how many of these artworks are also found by a matching vocabulary term, either used directly as an object property (OP) or indirectly by a series of relations up to path length 5 (P5).

The column in Table 5.3 labelled *LP* shows for each query the total number of artworks found via a matching literal property. The $\cap OP$ column shows how many of these artworks are found via a vocabulary term with a label matching the query. For the **concept** queries none of these artworks are found by an object property. For most of the other types of queries the object property provides an alternative. For example, the painting with the matching title “Van Gogh’s bedroom in Arles” is also found via the vocabulary term *Vincent van Gogh*, which is used as the value of the `dc:creator` property.

On average 42% of the artworks found via a free text field can also be found via an object property. Of the 58% of artworks without an object property many are found via a match on an editorial note. For example, several artworks have a literal property that describes the background literature used for cataloguing, e.g. “the tulip book”. These editorial notes match with the queries, e.g. `book`, but the artworks found are in most cases not relevant. For some queries there are, however, also relevant artworks that can only be found via a literal property. For example, in the peddler use case the relevant artworks from the Rijksmuseum collection could only be found via a matching title or description. In general, the artworks found via literal properties cannot be excluded from the search results, but some properties should be excluded to increase precision.

By using longer paths in the graph even more alternative paths become available to find artworks. The column in Table 5.3 labelled $\cap P5$ shows the number of artworks that can also be found by paths up to length 5. At path length 5, 79% of the artworks found via a free text field can be found via a vocabulary term. For example, a painting created by Giovanni Antonio Boltraffio was found for the query `da vinci` because it matched with the artwork’s text of a description property. The same artwork is also found via the vocabulary term for *Boltraffio*, which is the value of the `dc:creator` property, and this person is related to *Leonardo Da Vinci* by the associative relation `ulan:student_of`. We discuss in the following paragraphs how to exploit these types of paths.

5.7.2 Using vocabulary terms for query disambiguation

The experts considered the vocabulary terms useful to disambiguate the query. We investigate which vocabulary terms are good candidates for this type of query disambiguation. As a baseline we collect all vocabulary terms with a literal matching the query. Next, we compute the different literal properties by which these vocabulary terms are found and the paths that relate them to artworks.

The *All* column in Table 5.3 shows the total number of vocabulary terms with a literal matching the query. For the **concept** and some **location** queries many different terms are found. Also for some **person** queries more than 100 different matching vocabulary terms are found. On further analysis we observe that a large part of these vocabulary terms are found via descriptions, such as a biography. The precision of the results found via these terms will be low. For example, the vocabulary term for the Dutch politician Hendrikus Colijn matches the query **war**, as this query occurs in his biography "...He served as minister of **war**...". However, most of the artworks where Hendrikus Colijn is the **dc:subject** do not depict war.

To increase precision we can consider using only the terms for which the query matches with a "label property", **rdfs:label** or one of its sub-properties. The *Label* column in Table 5.3 shows the number of vocabulary terms with a matching label. Only 31% of terms matching the query are via a label property. For some queries the number of different vocabulary terms is still large. However, for disambiguation not all the terms are required. The next column, labelled *in OP*, shows that only 14% of the terms with a matching label are directly related to an artwork. When we are only interested in artworks found via object properties, these terms are sufficient for the query disambiguation.

Where longer paths are used to find artworks, more vocabulary terms are related to the artworks and thus more interpretations of the query become available. The column in Table 5.3 labelled *in P5*, shows the number of vocabulary terms matching the query and related to artworks at path length 5. For the 25 queries 51% of the vocabulary terms with a matching label are related to an artwork at path length 5. In other words when longer paths are considered more interpretations of the query lead to artworks.

5.7.3 Including equivalence

The external vocabularies provide information that the experts deem useful in the search process. We investigate the effect of automatically including information from external sources in the search process. As a baseline we collected all artworks found via vocabulary terms with a matching label. The column labelled *OP* in Table 5.3 shows the results. The next column, labelled *+Eq*, shows the number of artworks found when equivalence relations are also included.

The effect of the equivalence relations is in general small. Only for the *concept* queries significantly more artworks are found. This increase is caused by the external sources that provide English labels that can be matched with the query, whereas the in-house vocabularies only provide Dutch or French labels. Including the equivalence alignment relations, thus, provides support for multilingual search.

5.7.4 Integrating specialisation and generalisation

The experts indicated that hierarchically related terms could be useful query suggestions. We investigate how specialisation, generalisation and siblings can be integrated into the search process. As a baseline we use the artworks found via object properties and equivalence alignments (shown in the column labelled *+Eq* in Table 5.3). The equivalence alignments are included to also make the hierarchical relations from the external sources available to find artworks. Compared to this baseline we investigate the additional artworks found by including specialisations, generalisations and siblings.

The column labelled *+Spec* in Table 5.3 shows for each query the total number of artworks found via one or more `skos:narrower` relations. For the `concept` and `location` queries we see several large increases compared to the number of artworks found via the equivalence path type alone. Automatically including these specialisations can, however, result in many irrelevant artworks for a number of queries. For example, the query `rome` matches with a vocabulary terms for the city in Italy, but also with several mythological events from `ICONCLASS`. Including specialisations for all these terms will reduce precision, as typically only one or a few interpretations of the query are intended. In these case it is better to apply specialisation after the query is disambiguated.

The `location` and `concept` queries can also be generalised. However, these generalisations lead to overly generic concepts, such as the continent Europe. We do not further investigate generalisations of the queries here, but only note that for more specific queries, such as `peddler` in the first expert use case, generalisations were more useful. Instead we take a closer look at sibling terms, the combination of generalisation and specialisation. As shown in the *+Sib* column in Table 5.3, the inclusion of sibling terms has dramatically increased the number of results. For example, for the `location` queries artworks related to all other countries in Europe are found. Again for the specific concept `peddler`, located deep in the hierarchical structure, we observe that the sibling terms are closer related to the query. The use of sibling terms should, thus, be in control of the user.

5.7.5 Integrating associations

The experts indicated that associations could be useful query suggestions. We investigate the additional artworks found via association relations and the vocabulary terms by which these artworks are found. The column labelled *+Assoc* in Table 5.3 shows for each query the total number of artworks found via a path with one or two association relations. For all types of queries we see an increase in the number of artworks compared to the artworks found via object properties alone. For all queries, on average 11 times as many artworks are found. In particular, large increases are shown for `person` queries. These are predominantly found via the associative relations in `ULAN`. Large numbers of artworks are also found for

the queries **rome**, **italy** and **paris**. The locations matching these queries are related to persons, for example by properties such as `birthPlace`, and these persons are themselves associated to other persons.

The column labelled *Term* in Table 5.3 shows the number of associated vocabulary terms per query. A large number of vocabulary terms are associated with a number of queries, with a maximum of 2,768 for the query **paris**. To get more insight into the specific types of associations we compute the different types of properties by which these terms are found, the column labelled *Rel*. In particular, for the queries with a large number of associated vocabulary terms we observe that these are found by a relatively small number of relation types. We also compute the different types of associated terms e.g. person, location, event, collection and concept, the column labelled *Type*. We observe that most queries are associated to more than one type of term, but on average the queries have 2 different types of associated terms. We conclude that the types of the terms and the different relations, provide some categorisation of the large number of associations.

5.8 Implications for design

Based on the findings from the experiments we discuss the requirements to effectively support domain experts in artwork search. We observed that the search process typically consists of multiple iterations:

- The user starts with a basic keyword search to get an idea of the artworks that are available in the collection,
- if insufficient or irrelevant results are found the user reformulates the query,
- if the result set is too large the user adds additional filters.

In this section we describe a number of implications on the search functionality to support basic search, query reformulation and faceted result filtering. For each of these we describe how the search algorithm should be configured and discuss the implications on the presentation of the search results and navigation paths. Where applicable we discuss related work.

5.8.1 Basic search functionality

If the goal is to support the user in finding artworks “directly” related to the query, only literal and object property paths should be searched, in combination with the equivalence relations to cater for name, spelling and language variants. To increase precision of the obtained results, specific properties can be excluded. First, the free text fields found via editorial notes and sub properties of `rdfs:comment` could be excluded, as these “meta” properties are unlikely to contain results that are

relevant for most users. Second, only vocabulary terms with a matching literal value of a sub-property of `rdfs:label` could be included, as the vocabulary terms matching on other literal properties make the relation between query and result indirect. Additionally, assessment of the results is, in both cases, more difficult. The user should, however, have the opportunity to disable these restrictions when high recall is important.

There are tasks where the result set should also contain artworks related to specialisations of the query. For example, for a query on works made in Germany, it makes sense to also include works made in a city within Germany, as we observed in the “Gregory” use case. The `skos:narrower` relation, however, is used for different types of specialisations in our data set and yields low precision for many queries. For example, the concept `war` specialises in WORDNET to `battle`; `battle` specialises to `soldier`, but also to `horse`. It is unlikely that returning depictions of horses on a query for depictions of war is the intended behavior for most search tasks. The user should thus be able to control the in- or exclusion of specialisations.

In our data set, we observed that there are different interpretations for most queries and typically only one of these is intended by the user. The user experiment (Section 5.6) and the result analysis (Section 5.7), showed that automatically including the indirect relations for the other interpretations may dramatically reduce precision. We thus advise not to include specialisations before the query has been disambiguated. For the other types of relations we should be even more cautious. Hollink et al. showed that there are only a few combinations of hierarchical relations from WORDNET that actually yield good precision and recall (Hollink et al. 2007). We thus advise to omit relations other than specialisation in the basic search functionality. We discuss below how to use them for query reformulation.

In the presentation of the search results, the relation between the results and the query should be communicated to the user, as the domain experts assess the results found via controlled vocabulary terms differently from results found via a plain text field. In (Schreiber et al. 2008), for example, the results are clustered based on the relation between results and query. As mentioned by Hearst, such clustering has the advantage that irrelevant groups of results can be quickly eliminated (Hearst 2006).

5.8.2 Interactive query reformulation

The large number and the diversity of the relations make it difficult to effectively include them automatically in basic search functionality. Koenemann and Belkin also concluded that interactive query expansion improves effectiveness and user satisfaction over automatic expansion (Koenemann and Belkin 1996).

As most search sessions require several queries before the desired results are obtained, effective support for interactive query reformulation would thus be a useful feature of a semantic search application. The relations between vocabulary terms

are likely candidates for such functionality. The experts indicated that they want to explore multiple search strategies. We distinguish three such strategies based on expert users feedback in Section 5.6: disambiguation of the query with vocabulary terms, specialisation or generalisation of the query and recommendation of associated vocabulary terms.

Query disambiguation To disambiguate multiple interpretations of a keyword query, the vocabulary terms should be provided as suggestions. The suggestions should include at least all the vocabulary terms used in the basic search functionality, as for these it is known they are directly related to artworks. For further exploration other related vocabulary terms could be suggested to the user. Presenting them separately makes the user aware of the difference.

A selected vocabulary term provides the query for the basic search functionality. The URI of this term can be directly used to filter the results. This, however, will not find results by free text fields. It will require further research to discover if and how the labels of the vocabulary terms can also be used for disambiguation of the free text fields.

In the presentation of the suggested navigation paths, ranking could help the user choose the appropriate vocabulary terms. Meij et al. demonstrated the use of DBpedia to discover the concepts contained in text-based queries (Meij et al. 2009). They show that the corresponding concepts can be effectively re-ranked by learning the most effective features. The literature also provides suggestions for grouping similar results. For example, the terms can be grouped by different types (Amin et al. 2009). This requires vocabulary terms to have more specific types than `skos:Concept` alone. In Chapter 3 (Hildebrand et al. 2009) we concluded that in term search additional information is often required to disambiguate terms that have similar labels, for example, by showing the profession and birth date of people.

Query specialisation or generalisation The hierarchically related vocabulary terms could be presented to the user as specialisation or generalisation suggestions, including at least the narrower terms and a broader term. The equivalence alignments could also be included, as different thesauri provide their own hierarchical structures.

The hierarchical relations of similar types of thesauri may need to be integrated into a single structure. For geographical thesauri this is often straightforward. In Chapter 3 (Hildebrand et al. 2009) we demonstrated that the integration of TGN and the in-house location thesaurus of the Rijksmuseum created a useful extension for both sources. TGN providing the top level of the hierarchy, with the Rijksmuseum thesaurus contain specific details, such as street names (Hildebrand et al. 2009). The hierarchical structures of different types of thesauri are, however, often better presented as alternatives, providing different perspectives on the topic

(e.g. art specific in AAT, religious, biblical and mythological in ICONCLASS and lexical in WORDNET).

For the interface to support the navigation, we advise a design that provides interactive expansion, as this gives the user control over the path length and the direction, preventing the explosion of related terms. In addition, Joho et al. also showed that the presentation of a hierarchical structure can significantly reduce the time users need for query refinement compared to suggestions presented in a list (Joho et al. 2002).

Recommending associated terms After disambiguation of the query, vocabulary terms that are associated to the query, or otherwise related, could be made available as query suggestions. The suggestion algorithm could even include combinations of all path types. For example, in the first use case the concept of **trade** was associated to the concept **salesman**, which was a generalisation of the query **peddler**.

In the presentation of suggested navigation paths it is important that the relation to the query is communicated. The experts indicated that the type of relation helps to determine if a suggestion should be explored. Magennis and Rijsbergen showed that it is often difficult for end users to determine which suggestions are more useful (Magennis and van Rijsbergen 1997) and Ruthven concluded that the identification of relationships among related information can help the user make such a decision (Ruthven 2003).

As the number of associated vocabulary terms become large, additional organisation needs to be provided. In Chapter 4 (Hildebrand et al. 2006) we demonstrated the use of sub-property relations to hierarchically organise the properties in the interface. To be helpful to the user, however, the sub-property hierarchy needs to be well designed.

5.8.3 Result filtering

In addition to reformulation of the query, the user also needs to be able to filter the result set on other dimensions. For example, P3 wanted to search for artworks related to the query **gregory**, but only when they were made in Germany at a particular time. In addition, the user should be able to combine query reformulation with result filtering. For example, P2 wanted to generalise the **Fuji** in combination with a constraint on the location, e.g. querying for volcanos (a generalisation of **Fuji**) but constrained to results made or depicting scenes in Japan.

A popular method to interactively add these types of constraints is faceted browsing (Yee et al. 2003). In Chapter 4 (Hildebrand et al. 2006) we showed that this functionality can be effectively applied to RDF data. The precise integration of facet browsing with basic search functionality and query reformulation requires further research.

The main implications for design are that basic search functionality should only include literal and object properties, combined with equivalence. Hierarchical and associative relations are best used after query disambiguation. In some cases, specialisation of the query can be directly included after disambiguation, whereas the inclusion of generalisation and associations always needs to be under control of the user. The interactive search functionality for query reformulation needs to be combined with methods for result filtering.

5.9 Conclusion

We conclude that there is no one-size-fits-all solution for semantic search. Instead effective end-user support requires the user to explore different search strategies, such as direct search on the artworks metadata, query disambiguation, query specialisation and generalisation, suggestions of associated terms and result filtering. The search functionality to support these strategies require different configurations of the path types. A graph search algorithm should be able to support these configurations. We analysed the potential paths and their configurations for a specific cultural heritage data set. In addition, the different types of search results and the large number of candidates for query reformulation require different types of organisation and presentation methods.

We consider this study a first exploration to better understand how to search in semantically-rich and heterogeneous linked data. On the one hand, the qualitative analysis confirms the results already known in Information Retrieval, such as the need for interactive solutions to word sense disambiguation, query expansion and result filtering. On the other hand, the study explored new aspects introduced by linked data. First, the presence of multiple (partially) aligned vocabularies introduces both new opportunities as well as new problems. Second, the annotations from controlled vocabularies and the relations between the terms from these vocabularies provide semantically-rich background knowledge. The explicit types of the terms and relations within this background knowledge can be exploited in the search functionality and result presentation.

We are currently working on implementations of specific types of search functionality. In future work we plan to perform quantitative evaluations of these individual solutions by conducting user experiments with a larger number of participants performing a specific search task.

	#artworks			#terms			#artworks			#artworks			#terms		
	<i>LP</i>	<i>NOP</i>	<i>NP5</i>	<i>All</i>	<i>Label</i>	<i>in OP</i>	<i>in P5</i>	<i>OP</i>	<i>+Eq</i>	<i>+Spec</i>	<i>+Sib</i>	<i>+Assoc</i>	<i>Terms</i>	<i>Rel</i>	<i>Type</i>
book	114	0	106	2,247	598	135	306	1,810	4,194	6,336	73,771	7,014	95	32	2
war	17	0	3	2,080	291	21	139	885	1,123	4,414	73,660	6,504	51	15	2
portugal	57	18	33	155	56	8	33	56	59	73	20,519	335	28	12	3
spam	5	0	3	572	20	0	8	0	71	171	21,930	4,762	51	22	3
rome	1,012	439	859	1,454	695	62	401	795	822	4,129	4,716	28,195	1,752	61	4
italy	326	25	262	1,142	390	55	265	902	1,059	2,280	19,322	26,784	617	51	4
greece	2	0	1	263	11	1	8	10	20	79	18,790	26	5	6	2
paris	646	241	403	1,283	561	100	275	2,089	2,207	2,238	5,448	28,557	2,768	60	4
poland	4	0	0	182	12	2	5	2	7	1	19,026	410	16	13	3
romania	0	0	0	60	7	0	6	0	0	10	19,024	0	0	0	0
prado	243	180	183	48	46	1	5	254	254	254	262	4,064	8	8	2
louvre	435	117	315	104	85	34	48	254	254	254	427	3,390	10	8	2
mona lisa	2	0	1	2	1	0	1	0	0	0	0	14	2	4	2
renaissance	316	23	291	278	77	3	31	143	143	549	8,364	2,304	27	20	3
klint	18	0	0	10	9	0	8	0	0	0	345	728	13	8	2
van gogh	331	286	329	63	48	4	12	374	374	374	715	2,384	26	14	2
vermeer	28	12	17	108	100	12	21	172	172	172	172	4,923	28	18	3
rubens	572	420	489	203	160	8	68	2,046	2,046	2,046	2,046	9,339	105	31	3
goya	40	7	8	45	41	5	12	139	139	139	484	455	13	11	2
shakespeare	12	1	2	109	13	0	4	0	0	0	16	0	2	2	1
munich	2	0	0	33	32	0	9	0	0	0	345	7,211	12	8	2
da vinci	17	9	10	35	22	9	14	18	18	18	25	956	22	15	2
monet	5	3	4	33	29	3	10	6	6	6	351	1,049	13	7	2
renoir	2	1	1	13	9	3	6	12	12	12	2,614	2,614	15	9	2
hitler	5	3	3	44	10	1	3	12	12	12	12	16	2	2	2
	4,211	1,785	3,323	10,566	3,323	467	1,698	9,979	12,992	23,567	289,782	142,034	5,681	437	59
		42%	79%		31%	14%	51%		1.3x	1.8x	22x	11x			

Table 5.3: Results from the analysis of the 25 search log queries. The first three columns show the number of artworks found via literal properties and the subset that are found via alternative paths. The next four columns show the total number of vocabulary terms matching the query and the subset that are found via a label. The columns labelled *in OP* and *in P5* show the subset directly or indirectly related to artworks. The columns labelled *OP* and *+Eq* show the number of artworks found via an object property, with or without equivalence relations included. The columns labelled *+Spec*, *+Sib* and *+Ass* show the number of artworks found via paths including equivalence and specialisation, siblings or 2 association relations. The final three columns show the vocabulary terms found by 2 association relations, the number of different relations by which they are found and their different types.

Chapter 6

ClioPatria: Semantic search and annotation framework

In this chapter we describe the architectural support required for the three case studies Chapter 3, Chapter 4 and Chapter 5. We focus on the support required for the graph search functionality described in Chapter 5. In addition, we discuss how the configurations of the result selection and presentation, as identified in the case studies on annotation Chapter 3 and semantic search Chapter 5, can be supported by parameterized web services. The implementation of the ClioPatria framework proves the feasibility of generic support for term and graph search on RDF data sets and their parameterization to provide task-specific support.

This chapter was published as “Thesaurus-based search in large heterogeneous collections” in the Proceedings of the International Semantic Web Conference 2008 (Wielemaker et al. 2008), where it received an honorary mention. It also appeared in the PhD thesis of Jan Wielemaker (Wielemaker 2009b). It was authored together with Jan Wielemaker and co-authored by Jacco van Ossenbruggen and Guus Schreiber.

6.1 Introduction

Traditionally, cultural heritage, image and video collections use proprietary database systems and often their own thesauri and controlled vocabularies to index their collection. Many institutions have made or are making (parts of) their collections available online. Once on the web, each institution, typically, provides access to their own collection. The cultural heritage community now has the ambition to integrate these isolated collections and create a potential source for many new

inter-collection relationships. New relations may emerge between objects from different collections, through shared metadata or through relations between the thesauri.

The MultimediaN E-culture project¹ explores the usability of semantic web technology to integrate and access museum data in a way that is comparable to the MuseumFinland project (Hyvönen et al. 2005). We focus on providing two types of end-user functionality on top of heterogeneous data with weak domain semantics. First, keyword search, as it has become the de-facto standard to access data on the web. Secondly, thesaurus-based annotation for professionals as well as amateurs.

This chapter is organised as follows. In Sect. 6.2 we first take a closer look at our data and describe our requirements by means of a use case. In section Sect. 6.3 we take a closer look a search and what components are required to realise keyword search in a large RDF graph. The ClíoPatria infrastructure is described in section Sect. 6.4, together with some illustrations on how ClíoPatria can be used. We conclude the chapter with a discussion where we position our work in the Semantic Web community.

6.2 Materials and use cases

Metadata and vocabularies In our case study we collected descriptions of 200,000 objects from six collections annotated with six established thesauri and several proprietary controlled keyword lists, which adds up to 20 million triples. <http://e-culture.multimedian.nl/demo/>) We assume this material is representative for the described domain. Using semantic web technology, it is possible to unify the data while preserving its richness. The procedure is described elsewhere (Tordai et al. 2007) and summarised here.²

The MultimediaN E-Culture demonstrator harvests metadata and vocabularies, but assumes the collection owner provides a link to the actual data object, typically an image of a work such as a painting, a sculpture or a book. When integrating a new collection into the demonstrator we typically receive one or more XML/database dumps containing the metadata and vocabularies of the collection. Thesauri are translated into RDF/OWL, where appropriate with the help of the W3C SKOS format for publishing vocabularies (Miles and Bechhofer 2009). The metadata is transformed in a merely syntactic fashion to RDF/OWL triples, thus preserving the original structure and terminology. Next, the metadata schema is mapped to VRA³, a specialisation of Dublin Core for visual resources. This mapping is realised using the ‘dumb-down’ principle by means of `rdfs:subPropertyOf`

¹<http://e-culture.multimedian.nl>

²The software can be found at <http://sourceforge.net/projects/annocultor>

³Visual Resource Association, <http://www.vraweb.org/projects/vracore4/>

and `rdfs:subClassOf` relations. Subsequently, the metadata goes through an enrichment process in which we process plain-text metadata fields to find matching concepts from thesauri already in the demonstrator. For example, if the `dc:creator` field contains the string *Pablo Picasso*, then we will add the concept `ulan:500009666` from ULAN⁴ to the metadata. Most enrichment concerns named entities (people, places) and materials. Finally, the thesauri are aligned using `owl:sameAs` and `skos:exactMatch` relations. For example, the art style *Edo* from a local ethnographic collection was mapped to the same art style in AAT⁵ (see the use cases for an example why such mappings are useful). Our current database (April 2008) contains 38,508 `owl:sameAs` and 9,635 `skos:exactMatch` triples and these numbers are growing rapidly.

After this harvesting process we have a graph representing a connected network of works and thesaurus lemmas that provide background knowledge. VRA and SKOS provide —weak— structure and semantics. Underneath, the richness of the original data is still preserved. The data contains many relations that are not covered by VRA or SKOS, such as relations between artists (e.g. ULAN `teacherOf` relations) and between artists and art styles (e.g. relations between AAT art styles and ULAN artists (de Boer et al. 2007)). These relations are covered by their original schema. Their diversity and lack of defined semantics make it hard to map them to existing ontologies and provide reasoning based on this mapping.



Use cases Assume a user is typing in the query “picasso”. Despite the name *Picasso* is a reasonably unique in the art world, the user may still have many different intentions with this simple query: a painting by Picasso, a painting of Picasso, the styles Picasso has worked in? Without an elaborate disambiguation process it is impossible to tell in advance.

Figure 6.1 show part of the results of this query in the MultimediaN demonstrator. We see several clusters of search results. The first cluster contains works from the Picasso Museum, the second cluster contains works by Pablo Picasso (only first five hits shown; clicking on the arrow allows the user to inspect all results); clusters of surrealist and cubist paintings (styles that Picasso worked in; not shown for space reasons), and works by George Braque (a prominent fellow Cubist painter, but the works shown are not necessarily cubist). Other clusters include works made from *picasso marble* and works with *Picasso* in the title (includes two self portraits). The basic idea is that we are aiming to create clusters of related objects such that the user can afterwards choose herself what she is interested in. We have found that even in relatively small collections of 100K objects, users discover interesting results they did not expect. We have termed this type of search tentatively ‘post-query disambiguation’: in response to a simple keyword query the user gets (in contrast to, for example, Google image search) semantically-grouped






⁴Union List of Artist Names is a thesaurus of the Getty foundation

⁵Art & Architecture Thesaurus, another Getty thesaurus

▼ Works in museum (2)

	
Self-Portrait Miró, Joan	Portrait of a Spanish Miró, Joan

▼ Works created by (92)

				
Rembrandtesque Picasso, Pablo	Reservoir at Horta Picasso, Pablo	Still Life with Picasso, Pablo	Glass, Dice, and Picasso, Pablo	Seated Old Man Picasso, Pablo

▶ Works with style/period **Surrealist** also used by artist (1)

▶ Works with style/period **Cubist** also used by artist (1)

▼ Works by professionally related artist (31)






				
Fruit Dish, Ace of Clubs Braque, Georges	Man with a Violin Braque, Georges	Bottle, Newspaper, Braque, Georges	Still Life BACH Braque, Georges	Black Fish Braque, Georges

Figure 6.1: Clustered result presentation when searching for “picasso”. Images of paintings courtesy of Mark Harden, used with permission.

results that enable further detailing of the query. It should be pointed out that the knowledge richness of the cultural heritage domain allows this approach to work. In less rich domains such an approach is less likely to provide added value. Notably typed resources and relations give meaning to the path linking a literal to a target object.

Another typical use case for search concerns the exploitation of vocabulary alignments. The Holy Grail of the unified cultural-heritage thesaurus does not exist and many collection owners have their own home-grown variants. Consider the situation in Figure 6.2, which is based on real-life data. A user is searching for “tokugawa”. This Japanese term has actually two major meanings in the heritage domain: it is the name of a 19th century shogun and it is a synonym for the Edo style period. Assume for a moment that the user is interested in finding works of the latter type. The Dutch ethnographic museum in Leiden actually has works in this style in its digital collection, such as the work shown in the top-right corner. However, the Dutch ethnographic thesaurus SVCN, which is being

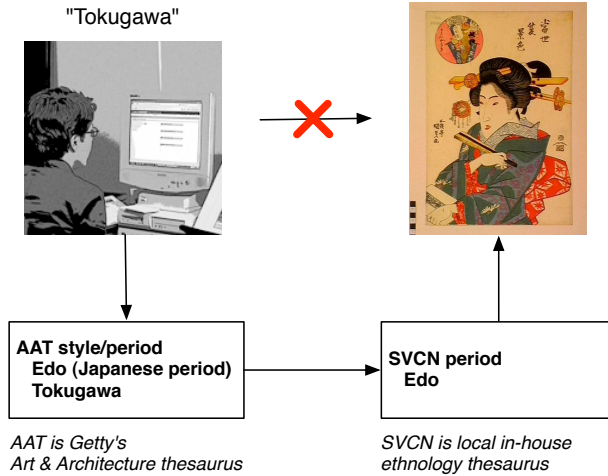


Figure 6.2: Explore alignment to find Edo painting from “tokugawa”. Image of print courtesy of Museum Volkenkunde.

used by the museum for indexing purposes, only contains the label “Edo” for this style. Fortunately, another thesaurus in our collection, the aforementioned AAT, does contain the same concept with the alternative label “Tokugawa”. In the harvesting process we learned this equivalence link (quite straightforward: both are Japanese styles with matching preferred labels). The objective of our graph search is to enable to make such matches.

Although this is actually an almost trivial alignment, it is still extremely useful. The cultural-heritage world (like any knowledge rich domain) is full of such small local terminology differences. Multilingual differences should also be taken into consideration here. If semantic-web technologies can help making such matches, there is a definite added value for users.

6.3 Required methods and components

In this section we study the methods and components we need to realise the keyword search described above. Our experiments indicate that meaningful matches between keyword and target often involve chains of up to about five relations. At this distance there is a potentially huge set of possible targets. The targets can be organised by rating based on semantics or statistics and by clustering based on the graph pattern linking a literal to the target. We discuss three possible ap-

proaches: querying using a fixed set of graph patterns, completely unconstrained graph search and best-first exploration of the graph.

6.3.1 Using a set of fixed queries

A cluster as shown in Figure 6.1 is naturally represented as a graph pattern as found in many Semantic Web query languages. If we can enumerate all possible meaningful patterns of properties that link literals to targets we reduce the search process to finding instances of all these graph patterns. This would be a typical approach in Semantic Web applications such as DBin (Tummarello et al. 2006). This approach is, however, not feasible for highly heterogenous data sets. Our current data contains over 600 properties, most of which do not have a well defined meaning (e.g., `detailOf`, `cooperatedWith`, `usesStyle`). If we combine this with our observation that it is quite common to find valuable results at 4 or even 5 steps from the initial keywords, we have to evaluate a very large number of possible patterns. To a domain expert, it is obvious that the combination of `cooperatedWith` and `hasStyle` can be meaningful while the combination `bornIn` and `diedIn` (i.e., A is related to B because A died in P , where B was born) is generally meaningless, but the set of possible combinations to consider is too large for a human. Automatic rating of this type of relation pattern is, as far as we know, not feasible. Even if the above is possible, new collections and vocabularies often come with new properties, which must all be considered in combination to the already created patterns.

6.3.2 Using graph exploration

Another approach is to explore the graph, looking for targets that have, often indirectly, a property with matching literal. This implies we search the graph from *Object* to *Subject* over arbitrary properties, including triples entailed by `owl:inverseOf` and `owl:SymmetricProperty`. We examine the scalability issues using unconstrained graph patterns, after which we examine an iterative approach.

Considering a triple store that provides reasoning over `owl:inverseOf` and `owl:SymmetricProperty` it is easy to express an arbitrary path from a literal to a target object with a fixed length. The total result set can be expressed as a union of all patterns of fixed length up to (say) distance 5. Table 6.1 provides the statistics for some typical keywords at distances 3 and 5. The table shows total visited and unique results for both visited nodes and targets found which indicates that the graph contains a large number of alternative paths and the implementation must deal with these during the graph exploration to reduce the amount of work. Even without considering the required post-processing to rank and cluster the results it is clear that we cannot obtain interactive response times for many queries using blind graph exploration.

Fortunately, a query system that aims at human users only needs to produce the most promising results. This can be achieved by introducing a distance measure

Keyword	Dist.	Literals	Nodes		Targets		Time (sec.)
			Visited	Unique	Visited	Unique	
tokugawa	3	21	1,346	1,228	913	898	0.02
steen	3	1,070	21,974	7,897	11,305	3,658	0.59
picasso	3	85	9,703	2,399	2,626	464	0.26
rembrandt	3	720	189,611	9,501	141,929	4,292	3.83
impressionism	3	45	7,142	2,573	3,003	1,047	0.13
amsterdam	3	6,853	1,327,797	421,304	681,055	142,723	39.77
tokugawa	5	21	11,382	2,432	7,407	995	0.42
steen	5	1,070	1,068,045	54,355	645,779	32,418	19.42
picasso	5	85	919,231	34,060	228,019	6,911	18.76
rembrandt	5	720	16,644,356	65,508	12,433,448	34,941	261.39
impressionism	5	45	868,941	50,208	256,587	11,668	18.50
amsterdam	5	6,853	37,578,731	512,027	23,817,630	164,763	620.82

Table 6.1: Statistics for exploring the search graph for exactly *Distance* steps (triples) from a set of literals matching *Keyword*. *Literals* is the number of literals holding a word with the same stem as *Keyword*; *Nodes* is the number of nodes explored and *Targets* is the number of target objects found. *Time* is measured on an Intel Core duo X6800.

and doing *best-first* search until our resources are exhausted (*anytime algorithm*) or we have a sufficient number of results. The details of the distance measure are still subject of research (Rocha et al. 2004), but not considered vital to the architectural arguments in this article. The complete search and clustering algorithm is given in Figure 6.3. In our experience, the main loop requires about 1,000 iterations to obtain a reasonable set of results, which leads to acceptable performance when the loop is pushed down to the triple store layer.

6.3.3 Term search

The combination of best-first graph exploration with semantic clustering, as described in Figure 6.3, works well for ‘post-query’ disambiguation of results in exploratory search tasks. It is, however, less suited for quickly selecting a known thesaurus term. The latter is often needed in semantic annotation (Chapter 3) (Hildebrand et al. 2009) and ‘pre-query’ disambiguation search tasks. For such tasks we rely on the proven *autocompletion* technique, which allows us to quickly find terms related to the prefix of a label or a word inside a label, organise the results (e.g., organise cities by country) and provide sufficient context (e.g., date of birth and

-
1. Find literals that contain the same stem as the keywords, rate them on minimal edit distance (short literal) or frequency (long literal) and sort them on the rating to form the initial *agenda*
 2. Until satisfied or empty *agenda*, do
 - (a) Take highest ranked value from *agenda* as *O*. Find $\mathbf{rdf}(S,P,O)$ terms. Rank the found *S* on the ranking of *O*, depending on *P*. If *P* is a subProperty of `owl:sameAs`, the ranking of *S* is the same as *O*. If *S* is already in the result set, combine their values using $R = 1 - ((1 - R_1) \times (1 - R_2))$. If *S* is new, insert it into *agenda*, else reschedule it in the agenda.
 - (b) If *S* is a target, add it to the *targets*. Note that we must consider $\mathbf{rdf}(O,IP,S)$ if there is an `inverseOf(P,IP)` or *P* is symmetric.
 3. Prune resulting graph from branches that do not end in a target.
 4. Smush resources linked by `owl:sameAs`, keeping the resource with the highest number of links.
 5. Cluster the results
 - (a) Abstract all properties to their VRA or SKOS root property (if possible).
 - (b) Abstract resources to their class, except for instances of `skos:Concept` and the top-10 ranked instances.
 - (c) Place all triples in the abstract graph. Form (RDF) Bags of resources that match to an abstracted resource and use the lowest common ancestor for multiple properties linking two bags of resources.
 6. Complete the nodes in the graph with label information for proper presentation.
-

Figure 6.3: Best first graph search and clustering algorithm

death of a person). Often results can be limited to a sub-hierarchy of a thesaurus, expressed as an extra constraint using the transitive `skos:broader` property. Although the exact technique differs, the technical requirements to realise this type of search are similar to the keyword search described above.

6.3.4 Literal matching

Similar to document retrieval, we start our search from a rated list of literals that contain words with the same stem as the searched keyword. Unlike document retrieval systems such as Swoogle (Ding et al. 2004) or Sindice (Tummarello et al. 2007), we are not primarily interested in which RDF documents the matching literals occur, but which semantically related target concepts are connected to them. Note that term search (Sect. 6.3.3) requires finding literals from the prefix of a contained word that is sufficiently fast to be usable in autocompletion interfaces (see also Bast and Weber 2007).

6.3.5 Using SPARQL

If possible, we would like our search software to connect to an arbitrary SPARQL endpoint. Considering the *fixed query* approach, each pattern is naturally mapped onto a SPARQL graph pattern. *Unconstrained graph search* is easily expressed too. Expressed as a CONSTRUCT query, the query engine can return a minimal graph without duplicate paths.

Unfortunately, both approaches proved to be unfeasible implementation strategies. The best-first graph exploration requires one (trivial) SPARQL query to find the neighbours of the next node in the *agenda* for each iteration to update the agenda and to decide on the next node to explore. Latency and volume of data transfer make this unfeasible when using a remote triple store.

The reasoning for clustering based on the property hierarchy cannot be expressed in SPARQL, but given the size and stability of the property hierarchy we can transfer the entire hierarchy to the client and use local reasoning. After obtaining the clustered results, the results need to be enriched with domain specific key information (title and creator) before they can be presented to the user. Re-requesting the same information from a large collection of resources can be realised using a rather inelegant query as illustrated in Figure 6.4.

```
SELECT ?i1 ?i2 ...
WHERE { { ulan:artists1 rdfs:label ?i1 } UNION
        { ulan:artists2 rdfs:label ?i2 } UNION
        ...
}
```

Figure 6.4: Query the labels of many resources

We conclude that SPARQL is inadequate for adaptive graph exploration algorithms, incapable of expressing lowest common parent problems and impractical for enriching computed result sets⁶. Finally, regular expression literal matching cannot support match on stem. Prefix and case insensitive search for contained word can be expressed. Ignoring diacritic marks during matching is generally required when dealing with text from multiple languages using multiple scripts, but is not supported by the SPARQL regular expression syntax.⁷

6.3.6 Summary of requirements for search

- Obtain rated list of literals from stem and prefix of contained words.

⁶Some of these issues are being addressed in the changes that will be made to the SPARQL query language to form SPARQL 1.1 <http://www.w3.org/TR/2009/WD-sparql11-query-20091022/>

⁷Some regex variations support diacritic mark matching. For example CQP <http://www.ims.uni-stuttgart.de/projekte/CorpusWorkbench/>

- The OWL primitives `owl:inverseOf` and `owl:SymmetricProperty` are used to specify which relations are searched in both directions.
- Entailment over `owl:TransitiveProperty` is used to limit results to a particular hierarchy in a SKOS thesaurus.
- Entailment over `owl:sameAs` for term search.
- The best-first graph exploration must be tightly connected to the triple store to enable fast exploration of the graph.
- Reasoning with types as well as the class, concept and property hierarchy. This includes finding the lowest common parent of a set of resources in these hierarchies. Note that none of these form strict trees (i.e., the relations form cycles and nodes have multiple parents).

6.4 The ClioPatria search and annotation toolkit

We have realised the functionality described in the previous section on top of the SWI-Prolog⁸ web and Semantic Web libraries (Wielemaker et al. 2008; Wielemaker et al. 2007) that are distributed as standard packages of SWI-Prolog. This platform provides a scalable in-core RDF triple store (Wielemaker et al. 2003) and a multi-threaded HTTP server library ((Wielemaker et al. 2008)). ClioPatria is the name of the reusable core of the E-culture demonstrator, the architecture of which is illustrated in Figure 6.5. First, we summarise some of the main features of ClioPatria.

- Running on a Intel core duo X6800@2.93GHz, 8Gb, 64-bit Linux it takes 120 seconds elapsed time to load the 20 million triples. The server requires 4.3Gb memory for 20 million triples (2.3Gb in 32-bit mode). Time and space requirements grow practically linear in the amount of triples.
- The store provides safe persistency and maintenance of provenance and change history based on a (documented) proprietary file format.
- Different operations require a different amount of entailment reasoning. Notably deleting and modifying triples complicates maintenance of the pre-computed entailment. Therefore, reasoning is as much as possible based on backward chaining, a paradigm that fits naturally with Prolog's search driven programming.

⁸<http://www.swi-prolog.org>

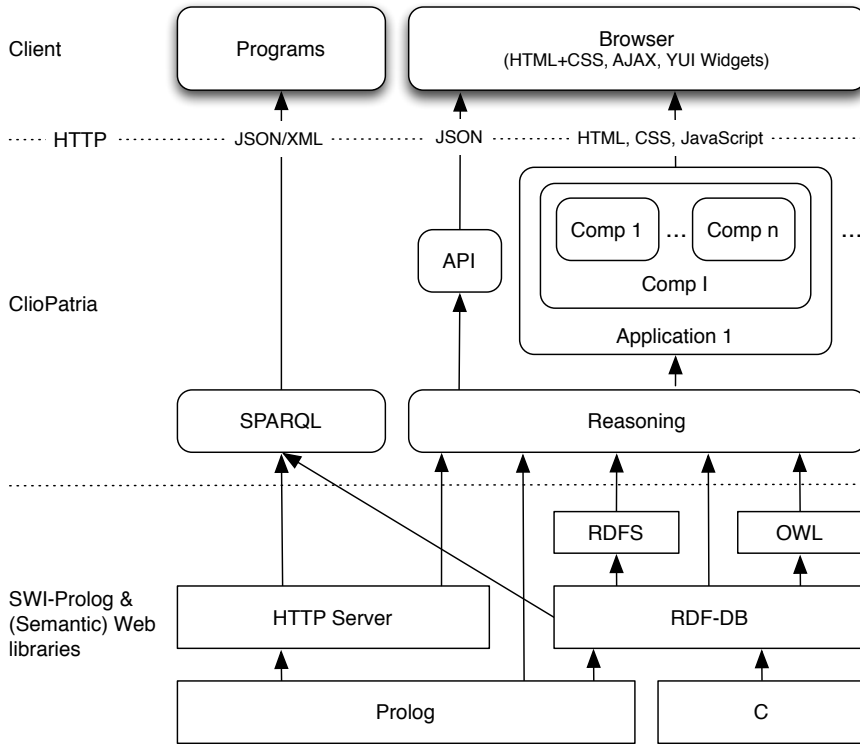


Figure 6.5: Overall architecture of the ClioPatria server

6.4.1 Client-server architecture

In contrast to client-only architectures such as Simile’s Exhibit (Huynh et al. 2007), ClioPatria has a client-server architecture. The core functionality is provided as HTTP APIs by the server. The results are served as presentation neutral data objects and can thus be combined with different presentation and interaction strategies. Within ClioPatria, the APIs are used by its web applications. In addition, the APIs can be used by third party applications to create mash-ups.

The ClioPatria toolkit contains web applications for search and annotation. The end-user applications are a combination of server side generated HTML and client side JavaScript interface widgets. The generated HTML contains the layout of the application page including place markers for the interface widgets. The in-

terface widgets are loaded into the page on the client side and populate themselves by requesting data through one of the APIs.

The reusability of the design is demonstrated by a variety of applications that use ClioPatria, either as central server or as service within a larger application. Besides for the MultimediaN E-Culture demonstrator⁹ for which ClioPatria was developed, it is currently in use by the following projects. The K-Space European Network of Excellence is using ClioPatria to search news.¹⁰ At the time of writing Europeana¹¹ is setting up ClioPatria as a demonstrator to provide multilingual access to a large collection of very diverse cultural heritage data. The ClioPatria API provided by the E-Culture Project is also used by the CATCH/CHIP project Tour Wizard that won the 3rd prize at the Semantic Web Challenge of 2007. For the semantic search functionality CHIP uses the web services provided by the ClioPatria API.

6.4.2 Output formats

The server provides two types of presentation oriented output routines. *Components* are Prolog grammar rules that define reusable parts of a page. A component produces HTML and optionally posts requirements for CSS and JavaScript. For example, the component `localview` emits an HTML `div` and requests the JavaScript code that realises the detailed view of a single resource using an AJAX widget. Components can embed each other. *Applications* produce an entire HTML page that largely consists of configured components. HTML pages, and therefore applications, cannot be nested. The HTML libraries define a resource infrastructure that tracks requests for CSS and JavaScript resources and uses this together with declarations on CSS and JavaScript dependencies to complete the HTML head information, turning components into clean modular entities.

Client side presentation and interaction is realised by JavaScript interface widgets. The widgets are built on top of the YAHOO! User Interface (YUI) library.¹² ClioPatria contains widgets for autocompletion, a search result viewer, a detailed view on a single resource, and widgets for semantic annotation fields. The result viewer can visualise data in thumbnail clusters, a geographical map, Simile Exhibit, Simile Timeline and a Graphviz¹³ graph visualisation.

The traditional language of choice for exchanging data over the network is XML. However, for web applications based on AJAX interaction an obvious alternative is provided by JSON (*JavaScript Object Notation*¹⁴), as this is processed natively by JavaScript capable browsers. JSON targets at object serialisation

⁹<http://e-culture.multimedien.nl/demo/search>

¹⁰<http://newsml.cwi.nl/explore/search>

¹¹<http://www.europeana.eu/>

¹²<http://developer.yahoo.com/yui/>

¹³<http://www.graphviz.org/>

¹⁴<http://www.json.org/>

rather than document serialisation and is fully based on UNICODE. James Clark, author of the SP SGML parser and involved in many SGML and XML related developments acknowledges the value of JSON.¹⁵ JSON is easy to generate and parse, which is illustrated by the fact that the Prolog JSON library, providing bi-directional translation between Prolog terms and JSON text counts only 792 lines. In addition the community is developing standard representations, such as the SPARQL result format (Clark et al. 2007).

6.4.3 Web services provided by ClioPatria (API)

ClioPatria provides programmatic access to the RDF data via several web services¹⁶. The query API provides standardised access to the data via the SPARQL. As we have shown in Sect. 6.3 such a standard query API is not sufficient to provide the intended keyword search functionality. Therefore, ClioPatria provides an additional search API for keyword-based access to the RDF data. In addition, ClioPatria provides APIs to get resource-specific information, update the triple store and cache media items. In this chapter we only discuss the query and search API in more detail.

6.4.3.1 Query API

The SPARQL library provides a Semantic Web query interface that is compatible with Sesame (Broekstra et al. 2002) and provides open and standardised access to the RDF data stored in ClioPatria.

Both SPARQL are translated into a Prolog query that relies on the `rdf(S,P,O)` predicate provided by SWI-Prolog's RDF library and on auxiliary predicates that realise functions and filters defined by SPARQL. Conjunctions of `rdf/3` statements and filter expressions are optimised through reordering based on statistical information provided by the RDF library (Wielemaker 2005b). Finally, the query is executed and the result is handed to an output routine that emits tables and graphs in various formats specified by SPARQL.

6.4.3.2 Search API

The search API provides services for graph search (Figure 6.3) and term search (Sect. 6.3.3). Both services return their result as a JSON object (using the serialisation for SPARQL SELECT queries, Clark et al. 2007). Both services can be configured with several parameters. General search API parameters are:

- `query(string|URI)`: the search query.

¹⁵<http://blog.jclark.com/2007/04/xml-and-json.html>

¹⁶<http://e-culture.multimedien.nl/demo/doc/>

- **filter**(*false* | *Filter*): constrains the results to match a combination of *Filter* primitives, typically OWL class descriptions that limit the results to instances that satisfy these descriptions. Additional syntax restricts results to resources used as values of properties of instances of a specific class.
- **groupBy**(*false* | *path* | *Property*): if *path*, cluster results by the abstracted path linking query to target. If a property is given, group the result by the value on the given property.
- **sort**(*path.length* | *score* | *Property*): Sort the results on path-length, semantic distance or the value of *Property*.
- **info**(*false* | *PropertyList*): augment the result with the given properties and their values. Examples are `skos:prefLabel`, `foaf:depicts` and `dc:creator`.
- **sameas**(*Boolean*): smushes equivalent resources, as defined by `owl:sameAs` or `skos:exactMatch` into a single resource.

Results that are semantically related to a keyword, e.g. “picasso” can be retrieved through the graph search API with the HTTP request below. The `vra:Work` filter limits the results to museum objects. The expression `view=thumbnail` is a shorthand for `info = ["image":"thumbnail", "title":"vra:creator", "subtitle":"dc:creator"]`.

```
/api/search?query=picasso&filter=vra:Work&groupBy=path&view=thumbnail
```

Parameters specific to the graph search API are:

- **view**(*thumbnail* | *map* | *timeline* | *graph* | *exhibit*): shorthands for specific property lists of the `info` parameter.
- **abstract**(*Boolean*): enables the abstraction of the graph search paths over `rdfs:subClassOf` and `rdfs:subPropertyOf`, reducing the number of clusters.
- **bagify**(*Boolean*): puts (abstracted) resources of the same class with the same (abstracted) relations to the rest of the graph in an RDF bag. For example, convert a set of triples linking a painter over various sub properties of `dc:creator` to multiple instances of `vra:Work` into an RDF bag of works and a single triple linking the painter as `dc:creator` to this bag.
- **steps**(*Integer*): limits the graph exploration to expand no more than *Integer* nodes.
- **threshold**(*0.0..1.0*): cuts off the graph exploration at the given semantic distance (1.0: close; 0.0 infinitely far).

For annotation we can use the term search API to suggest terms for a particular annotation field. For example, suppose a user has typed the prefix “pari” in a location annotation field that only allows European locations. We can request matching suggestions by using the URI below, filtering the results to resources that can be reached from `tgn:Europe` using `skos:broader` transitively:

```
/api/autocomplete?query=pari&match=prefix&sort=rdfs:label&
filter={"reachable":{"relation":"skos:broader","value":"tgn:Europe"}}
```

Parameters specific to the term search API are:

- **match**(`prefix` | `stem` | `exact`): defines how the syntactic matching of literals is performed. Autocompletion, for example, requires `prefix` match.
- **property**(*Property*, *0.0.1.0*): is a list of RDF property-score pairs which define the values that are used for literal matching. The score indicates preference of the used literal in case a URI is found by multiple labels. Typically preferred labels are chosen before alternative labels.
- **preferred**(`skos:inScheme`, *URI*): in case URIs are smushed the information of the URI from the preferred thesaurus is used for augmentation and organisation.
- **compound**(*Boolean*): if `true`, filter results to those where the query matches the information returned by the `info` parameter. For example, a compound query *paris, texas* can be matched in two parts against a) the label of the place *Paris* and b) the label of the state in which *Paris* is located.

6.5 Discussion

In this chapter we analysed the requirements for searching in large, heterogeneous collections with many relations, many of which have no formal semantics. We presented the ClioPatria software architecture we used to explore this topic. Three characteristics of ClioPatria have proved to be a frequent source of discussion: the non-standard API, the central main memory store model and the lack of full OWL/DL support.

6.5.1 API standardisation

First, ClioPatria’s architecture is based on various client-side JavaScript Web applications around a server-side Prolog-based reasoning engine and triple store. As discussed in this chapter, the server functionality required by the Web clients can not be provided by an off-the-shelf SPARQL endpoint. This makes it hard for Semantic Web developers of other projects to deploy our Web applications on top of

their own SPARQL-based triple stores. We acknowledge the need for standardised APIs in this area. We hope that the requirements discussed in this chapter provide a good starting point to develop the next generation Semantic Web APIs that go beyond the traditional database-like query functionality currently supported by SPARQL.

6.5.2 Central, main memory storage model

From a data-storage perspective, the current ClioPatria architecture assumes images and other annotated resources to reside on the Web. All metadata being searched, however, is assumed to reside in main memory in a central, server-side triple store. We are currently using this setup with a 20M triples dataset, and are confident that our current approach will easily scale up to 300M triples on modern hardware (64Gb main memory). Our central main memory model will not scale, however, to the multi-billion triple sets supported by other state-of-the-art triple stores. For future work, we are planning to investigate to what extent we can move to disk-based or, given the distributed nature of the organisations in our domain, distributed storage strategies without giving up the key search functionalities of our current implementation. Distribution of the entire RDF graph is non-trivial. For example, in the keyword search, the paths in the RDF graph from the matching literals to the target resources tend to be unpredictable, varying highly with the types of the resources associated with the matching literals and the type of the target resources. Implementing a fast, semi-random graph walk in a distributed fashion will likely be a significant challenge. As another example, interface components such as a Web-based autocompletion Widget are based on the assumption that a client Web-application may request autocompletion suggestions from a single server, with response times in the 200ms range. realising sufficiently fast responses from this server without the server having a local index of all literals that are potential suggestion candidates will also be challenging. Distributing carefully selected parts of the RDF graph, however, could be a more promising option. In our current data sets for example, the sub-graphs with geographical information are both huge and connected to the rest of the graph in a limited and predictable fashion. Shipping such graphs to dedicated servers might be doable with only minor modifications to the search algorithms performed by the main server. This is a topic we need to address in future work.

6.5.3 Partial OWL reasoning

From a reasoning perspective, ClioPatria does not provide traditional OWL-DL support. First of all, the heterogeneous and open nature of our metadata repositories ensures that even when the individual data files loaded are in OWL-DL, their combination will most likely not be. Typical DL violations in this domain are properties being used as a data property with name strings in one collection,

and as an object property with URIs pointing to a biographical thesaurus such as ULAN in the other; or `rdfs:label` properties being used as an annotation property in the schema of one collection and as a data property on the instances of another collection. We believe that OWL-DL is a powerful and expressive subset of OWL for closed domains where all data is controlled by a single organisation. It has proved, however, to be unrealistic to use OWL DL for our open, heterogeneous Semantic Web application where multiple organisations can independently contribute to the data set.

Secondly, our application requires the triple store to be able to flexibly turn on and off certain types of OWL reasoning on a per-query basis. For example, there are multiple URIs in our dataset, from different data sources, representing the Dutch painter *Rembrandt van Rijn*. Ideally, our vocabulary mapping tools have detected this and have all these URIs mapped to one another using `owl:sameAs`. For an end-user interested in viewing all information available on Rembrandt, it is likely beneficial to have the system perform `owl:sameAs` reasoning and present all information related to Rembrandt in a single interface, smushing all different URIs onto one. For an expert end-user annotating an artwork being painted by Rembrandt the situation is different. When selecting the corresponding entry from a biographical thesaurus, the expert is probably interested into which vocabulary source the URI is pointing, and how entries in other vocabularies differ from the selected one. This requires the system to largely ignore the traditional `owl:sameAs` semantics, present all triples associated with the different URIs separately, along with the associated provenance information. This type of ad-hoc turning on and off of specific OWL reasoning is, to our knowledge, not supported by any off-the-shelf SPARQL endpoint, but crucial in all realistic multi-thesauri Semantic Web applications.

Thirdly, we found that our application requirements rarely rely on extensive subsumption or other typical OWL reasoning. In the weighted graph exploration we basically only consider the graph structure and ignore most of the underlying semantics, with only a few notable exceptions. Results are improved by assigning equivalence relations such as `owl:sameAs` and `skos:exactMatch` the highest weight of 1.0. We search the graph in only one direction, the exception being properties being declared as an `owl:SymmetricProperty`. In case of properties having an `owl:inverseOf`, we traverse the graph as we would have if all “virtual” inverse triples were materialised. Finally, we use a simple form of subsumption reasoning over the property and class hierarchy when presenting results to abstract from the many small differences in the schemata underlying the different search results.

Chapter 7

Configuring Semantic Web interfaces by data mapping

From the three case studies (Chapters 3, 4 and 5) it became clear that different types of user interaction are required and the search functionality and presentation methods of these need to be configured for the specific domain and task. In this chapter we propose a method to configure interface components for Semantic Web applications. We describe how the underlying data model of interface components can be formally defined, allowing Semantic Web application developers to configure a component using familiar RDF constructs. This chapter demonstrates how the search functionality and presentation methods of interface components can be configured for different domains.

This chapter was published as “Configuring semantic web interfaces by data mapping” in the Proceedings of the Workshop on Visual Interfaces for the Social Semantic Web (Hildebrand and van Ossenbruggen 2009) and was co-authored by Jacco van Ossenbruggen.

7.1 Introduction

Semantic Web data is typically rich in interconnections and highly heterogeneous. Designing user interfaces for applications that use this type of data is intrinsically hard. Designing interfaces for highly diverse data tends to lead to overly generic interfaces that do not communicate the richness of the data to the end user. On the other hand, interfaces that communicate this richness effectively tend to work well for only a set of fixed schemata and not for the entire dataset. Finding a sweet spot that balances these two forces is not trivial, especially if one takes into account that most Semantic Web developers are not UI specialists, and often have even little affinity with UI design. The problem is even deepened because

many Semantic Web developers tend to build UIs from scratch, as the fixed data model that is assumed by many off-the-shelf UI toolkits seems to conflict with the heterogeneity of their data.

In this chapter, we argue that for a wide range of applications, such a sweetspot can be identified and formally modelled in RDFS or OWL. By building standard interface components on top of this model, building an initial interface can be as easy as mapping the application’s data model to this interface model.

This approach has the advantage that it leverages the significant amount of design, implementation and testing effort already invested in today’s Web UI toolkits, and we believe that reuse of these commonly available toolkits will, in general, lead to better interfaces than interfaces that are designed and implemented from scratch by a (small) Semantic Web research team.

In addition, it replaces the traditional configuration and tailoring that is needed to adapt a generic interface to a local dataset by a straight forward RDF data mapping task, a skill most Semantic Web developers will have. By providing default mappings, it is even possible to provide a no-configuration, first crude version of a user interface, very early in the application development life cycle. This will give RDF data developers the “instant gratification” that has made many Web 1.0 and 2.0 applications so popular. It also encourages them to, during their data modelling and data development tasks, think about how their data will be used in the end-user applications, and how their modelling decisions may impact the interface.

Finally, while our approach provides pre-packaged solutions for common tasks, it does not prohibit application developers to go beyond those solutions in order to add more advanced or more application specific interface components. It is built to be extended or to build other layers on top of it.

This chapter is structured as follows. In the next section, we discuss related work and compare it to the approach proposed in this chapter. As an example user interface model and its binding to a Web UI toolkit, we discuss the interface model upon which the ClioPatria (Wielemaker et al. 2008) interface components are built, and how these components are implemented on top of the Yahoo! User Interface (YUI) library. We show how this model can be used to easily create two interfaces, one in the cultural heritage domain and one in the news domain. In the last section, we discuss the pros and cons of our approach.

7.2 Related Work

modelling part of the user interface in RDF is in itself not new. Fresnel (Pietriga et al. 2006) is a good example of an RDF vocabulary that can be used to specify the presentation details of the RDF data as they appear in the user interface. For interface widgets Fresnel forms a good solution to define the visualisation of individual data items within a widget. The Fresnel vocabulary can’t be applied

to for the configuration of widget specific properties as this requires a vocabulary specific for this widget.

Simile's faceted browser Longwell supports Fresnel for the visualisation of the results (SIMILE 2005). In addition, the set of facets displayed in the interface can be defined in a configuration file in RDF. The individual facets are, however, not be configurable.

Web interface widgets have become a standard in web development. The choice among JavaScript libraries is numerous and all provide a convenient abstraction of low-level issues, such as cross-browser support. Interface widgets for semantic content are also available. Eetu Mäkelä et. al. provide several interface widgets that work on top of their ontology service infrastructure ONKI (Mäkelä et al. 2007). Example widgets are autocompletion and faceted navigation. They also seem to strive for easy configuration of the widgets, but have not presented a clear model for this.

The approach of semantic widgets is also used by the Semantic Web company Mondeca¹.

7.3 Combining the Yahoo! User Interface Library with the ClioPatria Interface Model

In many domains there is a central role for persons, locations, times and artefacts. Sometimes these are modelled together as an event. For example, in the cultural heritage domain works of art are created by persons at a specific time and location. In a figurative art it may also be important to know which persons, times and locations are depicted. News images are also made on a specific time and place by a specific photographer and depict an event often involving persons, times and locations. Persons, locations and times are thus good candidates for a central model which is sufficiently generic, while sufficiently specific to answer the classic who, where and when questions to the end user. Man-made artefacts also play a central role in many domains and their specific properties can often be abstracted from by using, for example, a general vocabulary such as Dublin Core. In addition, different domains often deploy their own set of specific thesaurus concepts that describe the properties of events. We found that SKOS provides a sufficiently rich and abstract model to describe these concepts and their relations.

Within the semantic search and annotation framework ClioPatria (Wielemaker et al. 2008) we have developed several interface widgets. When deploying ClioPatria in a specific application domain we use persons, locations, times, artefacts and thesaurus concepts as an intermediate model between the interface model and the domain specific details of the underlying RDF data. In the following paragraphs

¹http://www.mondeca.com/index.php/en/intelligent_topic_manager/applications/semantic_portal_semantic_widgets

we explain the configuration dimensions of two of these interface widgets, auto-completion and faceted navigation, and show how this can be captured in an RDF interface model. In the next section we show how the intermediate model sketched above can be mapped to these widget's interface models.

7.3.1 Example 1: Autocompletion

Autocompletion is an interface feature that allows users to type only a few characters instead of a full word or phrase. After the user has entered the first characters, the system responds by completing the word or phrase. If the characters typed in so far can be completed in more than one way, most interfaces present a list of multiple options. The user can then either select one of the options from the list, or continue typing to narrow down the number of options.



Figure 7.1: Autocompletion suggestions of historical persons. Underneath the name a short biography is displayed. This contains the nationality, role/profession and birth/death date. Note that for the first person listed, only the profession is available in the data. The abbreviation RMA, shown to the right, indicates the thesaurus source.

In context of the Semantic Web autocompletion is useful to quickly find a vocabulary term by one of its labels². In Chapter 3 (Hildebrand et al. 2009) we argued that for different tasks and data sets autocompletion widgets require a different configuration. The screenshots in Figure 7.1 and Figure 7.2 show autocompletion suggestions of historical persons and thesauri concepts. In the next section we discuss the configurations of these two widgets, here, we focus on the main differences between these two widgets. First, the widgets suggest a different type of term (e.g. persons and concepts), thus, requiring a different *selection* of the right RDF resources. Second, the persons are *organised* in an alphabetically ordered list, while the concepts are grouped by different thesauri and ranked

²For sake of simplicity we do not consider finding terms by a label of a related term.

The image shows a user interface for an autocomplete widget. At the top, a search bar contains the text "siege". Below the search bar, there are two main sections of suggestions:

- Iconclass**: This section is titled "Iconclass" and has a "view all 40 results" link. It lists three suggestions:
 - [45K21] **siege** (Society, Civilization, Culture)
 - [45K] **siege, position war** (Society, Civilization, Culture)
 - [94H] **last months of the siege and the fall of Troy** (Classical Mythology and Ancient History)
- WordNet**: This section is titled "WordNet" and has a "view all 6 results" link. It lists three suggestions:
 - siege** (beleaguering, besieging) (blockade)
 - siege of Orleans** (Orleans) (beleaguering)
 - Siege Perilous** (seat)

To the right of the WordNet section, a secondary panel titled "siege, position war (more info)" is open. It shows two small images of historical prints. Below the images, a hierarchical structure is displayed:

- Society, Civilization, Culture
- warfare; military affairs
- siege, position war**
 - fortifications, military engineering
 - attack ~ siege
 - defence ~ siege
 - capture of city (after the siege)

Figure 7.2: Autocompletion suggestions of thesauri concepts. Results from both ICONCLASS and WORDNET are shown, each presented in a separate group. A secondary panel shows more information for the highlighted term (“[45K] siege, position war”), including the hierarchical structure the term is part of. The hierarchy contains the term itself in bold, its ancestors and the direct children. Images of the prints used with permission, courtesy of the Rijksmuseum Amsterdam.

according to popularity. Finally, the individual suggestions are *visualised* differently in each widget. The suggested persons are shown with extra biographical information, whereas, the concepts are shown within their hierarchy.

The two examples are built on top of the YUI autocomplete widget. The YUI widget contains several client side configuration parameters, it supports custom functions for result formatting, construction of remote data requests and it provides many event handlers. Although this is sufficient to configure the widget for an RDF data source, we experienced that it required extensive JavaScript programming to obtain the appropriate configurations. For example, visualising different types of information requires the configuration of the server request as well as the client side JavaScript formatting functions.

An interface model for an autocomplete widget provides a single focal point for the configuration of a widget and only requires Semantic Web modelling skills. Note, we do not claim that this is a complete model for autocomplete, we merely want to illustrate that it is possible to define an interface model for an autocomplete widget in RDF. The model we present is for an extended version of the YUI autocomplete widget (Hildebrand et al. 2007). We added support for clustered presentation of search results, a secondary display that is shown when the user hovers over a suggestion, a single configurable result formatting function and support for easy configuration of the server side search algorithm.

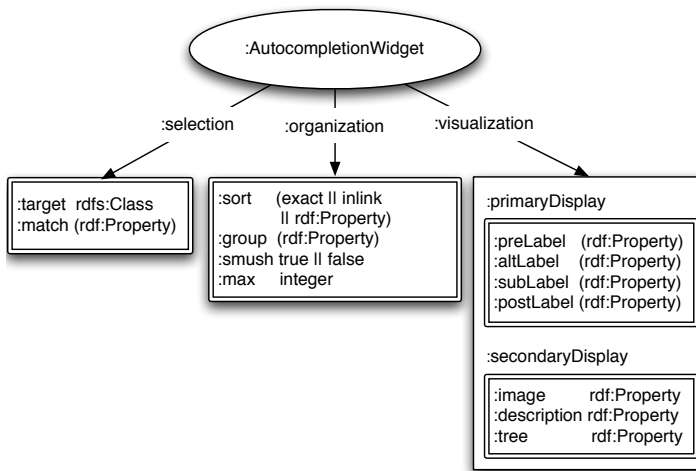


Figure 7.3: Interface model for ClioPatria's thesaurus concept autocomplete widget. All organisation and visualisation properties are optional.

An interface model for the *RDF concept autocomplete* widget is shown in Figure 7.3. The widget contains three main configuration properties that correspond with the three phases of the search process: selection, organisation and visualisation. For the selection of the appropriate term it should be known what type of terms should be selected and which literals should be used to find these terms? The first is configured by providing an **rdfs:Class** for **:target**³. The second requires the definition of a collection of RDF label properties for **:match**. The order

³The properties and classes used in the interface models are contained in our own namespace <http://e-culture.multimedial.nl/ns/interface/>. In this chapter we omit this namespace and simply write a colon.

of the properties indicates which property has preferences in case the same term is found by multiple labels. The selected terms can then be organised in a list or in groups of different lists. The grouping is performed on the values of the RDF properties provided for `:group`. The terms in the list can be ordered according to several criteria and are defined in a collection as a value of `:sort`. The built in constant, *exact* puts all terms with an exact matching label before terms with partial matching labels. Another built in constant is *inlink* that sorts the terms by the number of incoming links they have in the graph. Further sorting criteria are the display labels (explained in the next paragraph) or any RDF Property. The number of results that are returned can be limited by defining `:max`. In a grouped organisation the maximum applies to the number of items within a group. Finally, terms that are defined as equivalent (`owl:sameAs` or `skos:exactMatch`) are shown as a single suggestion when `:smush` is set to true.

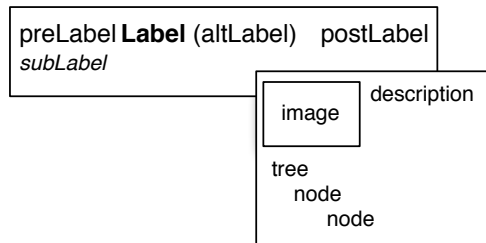


Figure 7.4: Layout of an autocomplete result. Primary display with a `preLabel`, the label itself, an alternative label between brackets, a `postLabel` aligned on the right side and a `subLabel` on a second line. The secondary panel provides additional space for larger content, such as images, descriptions and trees.

The results are visualised in a primary display panel. Besides the matching label itself the formatting function can print four additional labels: `:preLabel`, `:altLabel`, `:postLabel` and `:subLabel`. Figure 7.4 shows the skeleton of the primary display and an additional secondary result display. The secondary display, shown when the user hovers over a suggestion, provides place markers for an image, a longer piece of text and a tree containing the result.

7.3.2 Example 2: Faceted browsing

Facet browser interfaces provide a convenient way to navigate through a wide range of data collections. Originally demonstrated in the Flamenco system (Yee et al. 2003), facet browsing has also become popular in the Semantic Web community thanks to MUSEUMFINLAND (Hyvönen et al. 2005) and other systems (Huynh et al. 2007; m.c. schraefel et al. 2005; SIMILE 2005). An individual facet highlights one dimension of the underlying data. Often, the values of this dimension are hierarchically structured. By visualising and navigating this hierarchy in the user interface, the user is able to specify constraints on the items selected from the repository.

The facet browser we developed within ClioPatria, /facet (Chapter 4) (Hildebrand et al. 2006), can be applied to any RDF dataset. By considering the Class and Property hierarchy as special facets the user could configure the interface to her needs. In the Class facet the user selects the target objects (e.g. documents or persons) and from the Property facet she selects the facets she wants to navigate (e.g. creator and subject for documents or birth place and birth date for persons). This approach provides an instant interface for Semantic Web engineers. Presenting the raw data is, however, not suited for end user applications. In the project HealthFinland it was demonstrated that through careful user studies a more user-friendly configuration of the facets can be achieved (Suominen et al. 2007b).

Consider a faceted interface on a collection of documents. Each individual facet contains the values within one dimension. For example, one facet might display all the creators, whereas, another might display the subject categories. On an RDF data source this type of value *selection* corresponds to the values of a particular RDF property (e.g. dc:creator and dc:subject). Other selection criteria are also possible, such as all instances of a particular Class. In a similar fashion as the autocompletion suggestions, different types of facet values require different methods of *organisation*. The creators might be best organised in an alphabetically ordered list, while the hierarchical structure is important for the subject categories. Also the *visualisation* of facet values shows similarities with the autocompletion widget. Adding extra information in the display may help to disambiguate similar values. In addition, specific types of values (e.g. geographical locations and dates) are suited for alternative visualisation (geographical map and timeline).

When the number of defined facets is too large to be shown in the interface, it has to be defined which facets are shown. In Longwell a *facet view* can be defined as a collection of facets for a particular target. The facets defined in this view are shown, while all other facets are collapsed and available on requested. An alternative method is to allow multiple views and allow the user to select the most appropriate view. For example, the creation view contains all facets that cover the creation of a document, whereas, the content view contains the facets about the topic. In either solution, a view defines which facets are selected for display.

organised hierarchically, meaning that initially only the root values of the hierarchy are shown and after selection of one of these its children become available.

7.4 Configuring interface widgets: a mapping task

Given an interface model the configuration of a Semantic Web application becomes a task of mapping the right properties and classes to this model. In practice, this often means first finding a suited intermediate model for the domain. For example, in terms of persons, locations and times. Second mapping this intermediate model to the widget's interface model. We illustrate the mapping task with two use cases: configuring autocompletion components for an annotation application and configuring faceted navigation for a news application.

7.4.1 Use case 1: Rijksmuseum annotation user experiment

In Chapter 3 (Hildebrand et al. 2009) we developed a prototype interface for the subject annotation performed at the Rijksmuseum in Amsterdam, the Netherlands. The professional annotators of the Rijksmuseum describe thousands of artworks a year by assigning terms from controlled vocabularies. Finding the right term is complicated because the vocabularies used are large, very detailed, contain similar terms (or even duplicates) and often the annotator does not know exactly how to spell a term. We experienced that autocompletion helps professional annotators to find the right terms, but only when the widget is properly configured.

In an extensive study with these professionals we gathered the requirements for term search from multiple thesauri. During an iterative process of prototyping and discussion we tested several configurations of autocompletion widgets. A screenshot of the interface of the final prototype is shown in Figure 7.6. On the right side the interface contains three autocompletion fields to look-up terms from thesauri and a free text field to input dates. One of the results of the study is that the three autocompletion fields all required a different configuration.

The interface model for the autocompletion widget, as described in the previous section, is based on our findings at the Rijksmuseum. We acknowledge that a single study might not be sufficient to determine a complete interface model that applies to other domains. On the other hand, all three autocompletion fields required different features and configurations. Furthermore, the three fields cover generic types of terms (persons, thesaurus concepts and locations) that are very likely to be used in other domains as well.

We first introduce the vocabularies used in the annotation interface, before describing the configuration of the person and concept autocompletion fields. We used three thesauri with persons: Getty's United List of Artist Names⁴ (ULAN),

⁴http://www.getty.edu/research/conducting_research/thesauri/ulan/

The execution of Johan van Oldenbarnevelt

RP-P-OB-77.320

Print with a scene

update cancel

Who Historical persons
person
Oldenbarnevelt, Johan van x

What Iconclass (en), WordNet (en), events (nl)
(mythological) concept, object or event
beheading x

Where Name of place or region
geographical place
Den Haag x

When Date, year or period
enter date

done | cancel

Figure 7.6: Interface of the Rijksmuseum subject annotation interface. The four annotation fields in the right column are configured to support effective search in different thesauri. Image of the print used with permission, courtesy of the Rijksmuseum Amsterdam.

DBpedia’s RDF version of person data⁵ from Wikipedia (WP) and the Rijksmuseum’s own people thesaurus. All three thesauri were mapped to the generic “Person” scheme of ULAN. For places we also used, Getty’s Thesaurus of Geographic Names⁶ (TGN) and aligned it with the Rijksmuseum’s place thesaurus. We used SKOS for the geographical containment relations in combination with location-specific properties from TGN. The concepts used in this domain were also modelled or mapped to SKOS. In addition to the Rijksmuseum’s events thesaurus we added the RKD ICONCLASS⁷ thesaurus and, as a source for more general terms, W3C’s RDF version of Princeton’s WORDNET⁸.

⁵<http://dbpedia.org/>

⁶http://www.getty.edu/research/conducting_research/thesauri/tgn/

⁷<http://www.iconclass.nl/>

⁸<http://www.w3.org/2006/03/wn/wn20/>

7.4.1.1 Autocompletion on persons

```

:PersonAutocomplete
  a :Autocomplete ;
  :label "search person"@en;
  :label "zoek persoon"@nl ;
  :selection [
    :target ulan:Person ;
    :match (skos:prefLabel rdfs:label)
  ] ;
  :organization [
    :sort ("exact" :matchLabel);
    :smushing "true"
  ] ;
  :primaryDisplay [
    :subLabel (
      ulan:role
      ulan:nationality
      ulan:birthDate
      ulan:deathDate
    ) ;
    :altLabel skos:prefLabel ;
    :postLabel skos:inScheme
  ] ;
  :secondaryDisplay [
    :description ulan:biography ;
    :image vra:subject
  ] .

```

Figure 7.7: Person autocompletion allows autocompletion on instances of `ulan:Person`. The results are sorted first on exact matches and then alphabetically on the matching label. Results that are defined as equivalent (`skos:exactMatch` or `owl:sameAs`) are smushed. Each result is displayed with extra information. The primary display contains a short biography composed out of the values different properties and it contains the thesaurus source. The secondary display contains a full description and an artwork that depicts the person.

Figure 7.7 shows the configuration of the autocompletion widget in the `Who` field. The selection is restricted to terms of type `ulan:Person`. Note, the class of persons in the Rijksmuseum thesaurus and DBPedia people are subclasses of `ulan:Person`. We only consider literal values of `skos:prefLabel` and `rdfs:label`, where preference is given to the `skos:prefLabel` as this is first in the list. The results are organised alphabetically on the label and first showing all terms with an exactly matching label. The professionals at the Rijksmuseum explicitly indicated that they expect alphabetical ordering for a list of person names. As the autocompletion field gives

access to the terms from different overlapping vocabularies it turned out essential to smush equivalent results to a single suggestion.

The primary display contains three labels in addition to the matching label. The `:altLabel` is only shown in case the match was not found an a `skos:prefLabel`. Thus, when a hit is found by a `skos:altLabel` it's `skos:prefLabel` is also shown. The `:endLabel` contains the value of the `skos:inScheme` property. Thus indicating the thesaurus the term comes from. The professional annotators requested this information as terms are suggested from their own as well as other thesauri. The `:subLabel` shown beneath the main label is composed out of the values of four properties. Together these compose a short biography of the person. The annotators use this information to disambiguate similar persons from one another. The secondary display contains an image depicting the person and a longer biography.

```

:ConceptAutocomplete
  a :Autocomplete ;
  :label "search concept"@en;
  :label "zoek concept"@nl ;
  :selection [
    :target [
      owl:unionOf (
        ic:Concept ;
        wn:Synset ;
        rma:Event
      )
    ] ;
    :matchLabel (skos:prefLabel rdfs:label) ;
  ] ;
  :organization [
    :sort ("exact" "inlink") ;
    :group skos:inScheme
  ] ;
  :primaryDisplay [
    :subLabel skos:broader
  ] ;
  :secondaryDisplay [
    :description skos:note ;
    :image vra:subject
  ] .

```

Figure 7.8: Concept autocompletion allows autocompletion on instances of `skos:Concept`. The results are sorted first on exact matches and then on the number of in-links. The suggestions from the same thesaurus are grouped together. In the secondary display a tree is shown with the all ancestors and direct children of the result.

7.4.1.2 Autocompletion on thesaurus concepts

Figure 7.8 shows the configuration of the autocompletion widget in the **What** field. We only describe the configurations that are different from the **Who** field. The target is defined as an owl:union of three classes, **ICONCLASS** and **WORDNET** terms and the events from the Rijksmuseum thesaurus. The terms from the three thesauri are each shown in a separate group. The Rijksmuseum wanted to give preference to terms from **ICONCLASS** and only use **WORDNET** as a backup. organising the results in different groups allowed the annotators to easily compare terms from the different thesauri to one and other. Within each group the results are ordered by the number of links that are pointing to the term. Intuitively, this means that the popular terms are shown first.

7.4.2 Use case 2: K-Space Semantic News Browser

ClioPatria is used to support search and browsing of news items (Troncy 2008). These news items are described with multimedia standards, news codes from the IPTC standard and additional metadata from various thesauri modelled (mapped) to SKOS. The additional metadata is acquired through extraction of named entities such as persons, organisations and locations, from the textual stories. The extracted named entities are mapped to existing resources available on the Web, such as locations from Geonames, and persons from DBpedia. The data set in this use case consists of news items from 2006, including the World Cup football.

A screenshot of the faceted interface from ClioPatria is shown in Figure 7.9. The top part contains four facets: **document type**, **creation site**, **event** and **person**. The result viewer, visible below the facets, contains news items related to the keyword “zidane”. The current query is shown in the header of the result viewer. The user can extend the query by selecting values from the facets. In this case the value “photo” is selected from the **document type** facet. The other facets only contain values that correspond with the current result set. Note, this prevents the user from constructing queries that lead to an empty result set.

Figure 7.10 shows an excerpt of the facet and facet view configuration for a news demonstrator. The **creationSite** facet applies to instances of the class **NewsItem**, as indicated by the value of the **:facetTarget** property. This facet will display values from the **newsml:locCreated** property. As the values are part of a geographical containment hierarchy, this is used for the organisation. In the screenshot of Figure 7.9 the **creationSite** facet it is visible that initially only the children of the hierarchy root (e.g. World) are shown (e.g. Europe, Africa and Asia). When one of these values is selected, the children available through the hierarchical relation, **geo:parentFeature**, become available. Four facets are grouped into a facet view that covers the content of news items. In a similar fashion other facets can be grouped into views on the production and document characteristics. The facet view menu shown in the screenshot on the top left allows the user to

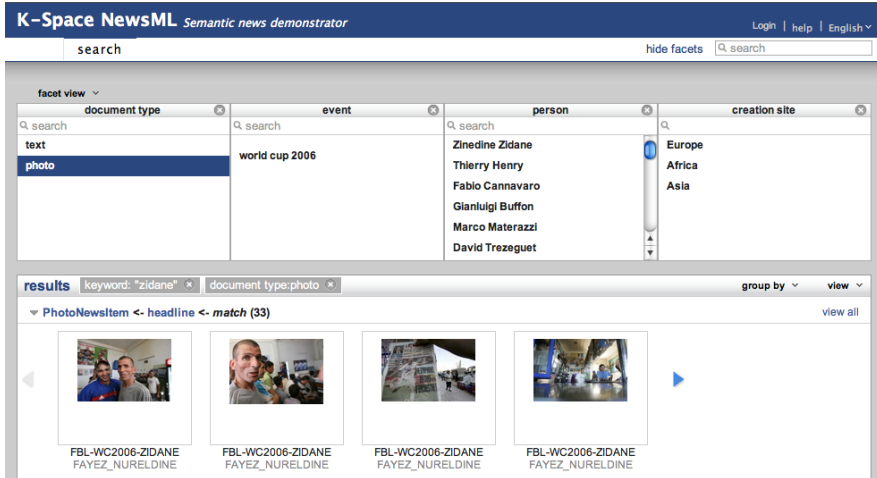


Figure 7.9: Faceted interface of the NewsML demonstrator. Four facets are active: **document type**, **creation site**, **event** and **person**. The value “photo” is selected from the **document type** facet. The full query also contains the keyword “Zidane”, as is visible in the header above the results. Images used with permission, courtesy of INA.

select one of these facet views.

7.5 Conclusion and Future Work

We have shown how we can use RDF to model the interface widgets of a specific Web application, an abstract intermediate data model, and the mapping between these two models. We argue that this approach can provide developers with an interface early in the development cycle of a Semantic Web application. As long as the chosen widgets, associated interface model and intermediate model prove to be sufficiently rich, all the developer needs to do is to provide the mappings (in RDF) between his own data model and the intermediate model, using skills that Semantic Web developers can be safely assumed to possess. This approach also allows Semantic Web UIs to be built on top of existing Web tool kits, without sacrificing the heterogeneity and semantic richness of the underlying data.

A first drawback of our approach is that our interface models are typically specific for a given interface widget or toolkit. If the same RDF data needs to be displayed the same way in multiple interfaces, a vocabulary such as Fresnel, that


```

:CreationSite
  a :Facet ;
  :label "creation site"@en ;
  :label "maak lokatie"@nl ;
  :target (newsml:newsItem) ;
  :property newsml:locCreated ;
  :hierarchy [
    a :Hierarchy ;
    :label "Geonames location hierarchy"@en ;
    :label "Geonames locatie hiërarchie"@nl ;
    :relation geo:parentFeature ;
    :root geo:World
  ] .

:DepictedPerson
  a :Facet ;
  ...

...

:ContentView
  a :FacetView ;
  :label "content"@en ;
  :label "content"@nl ;
  :target (newsml:newsItem ;
  :facets
    (
      :DocumentType
      :DepictedEvent
      :DepictedPerson
      :CreationSite
    ) .

:ProductionView
  a :FacetView ;
  ...

```

Figure 7.10: Excerpt of the facet and facet view configuration for a news demonstrator

abstracts from the interface technology used, might be a better alternative. In our applications, we have aimed to fully exploit the functionality of the interface widgets, and have traded the advantages of extra functionality against generality. Other developers might make a different trade off.

A second drawback surfaces when a given set of widgets and the associated interface model provides insufficient functionality. Then, extensions will require traditional Web scripting skills to develop extensions to widget set, typically involving a mix of HTML, CSS and JavaScript. But it also requires skills to be able to model these extensions in RDF or OWL, and this combination of skills might

be hard to find.

For future work, we would improve upon our current interface model and its implementation. The current implementation is realised as an integral part of the ClioPatria server framework, and we are investigating ways to be able to apply the same approach to create interfaces on top of arbitrary SPARQL endpoints.

Chapter 8

Conclusions

The goal of this thesis is to explore end-user access to semantically rich and heterogeneous linked data. To achieve this goal we investigated:

RQ 1. *How semantically-rich graph structures can be used in search functionality to support the user in finding objects in heterogeneous linked data?*

and,

RQ 2. *How semantically-rich graph structures can be used in the presentation of the results found in heterogeneous linked data?*

The research questions were first addressed in a literature study (Chapter 2). In a survey of search and browsing applications for Semantic Web data different types of search functionality and presentation methods are analysed. The survey provides an overview of the semantic relations, algorithms and interface designs used in existing applications. Without a common evaluation method¹ it, however, remains unclear how well these technologies improve support for end-users. Given a specific domain and search task it is, therefore, difficult to determine which semantic relations, algorithms and interface designs should be used to provide effective access.

This thesis studied the search functionality and result presentation in three case studies using the linked cultural heritage data: annotation, faceted browsing and semantic search. Chapter 3 studied the required support for artwork annotation in a user study with professional cataloguers from the Rijksmuseum Amsterdam. The final prototype was successfully used for professional annotation in an experimental setting and was judged positively by the cataloguers. The Rijksmuseum Amsterdam is currently investigating integration of such a system in their workflow.

¹At the time of writing the 3rd Semantic Search workshop has announced the organisation of a small-scale evaluation campaign for the first time. <http://km.aifb.uni-karlsruhe.de/ws/semsearch10/>

In Chapter 4, a generic solution for faceted browsing of heterogeneous linked data was explored by the implementation of a prototype. The prototype provides a completely data-driven solution that can be applied to any small to medium sized RDFS repository. In addition to the cultural heritage domain it has been used to give access to news images and articles (Troncy 2008), music songs and artists (Raimond and Sandler 2008) and historical events (Shaw et al. 2009).

In Chapter 5, support for semantic search was investigated in two experiments. First, the usefulness of different paths of relations in linked data is investigated in a user study with a small number of domain experts. A number of path types are identified and their usefulness is qualitatively evaluated. In the second experiment, the implementation of the path types in a semantic search application is investigated with the most frequently used queries from a search log. Implications for the design of an interactive search application for cultural heritage are derived.

In Chapter 6, the architectural support for the required search algorithms is investigated and their implementation as web services is discussed. The web service of the MultimediaN E-Culture project is used by the CHIP project (Aroyo et al. 2007) to provide text-based search functionality within their recommendation system. In Chapter 7 the required interface designs, in the form of several JavaScript Widgets, are discussed. To tailor the behavior of these widgets to a specific domain and task a method was presented to support configuration by data mapping.

Below, we revisit the two research questions posed in Chapter 1. We then reflect upon our work and discuss future research.

8.1 The research questions revisited

We discuss the high level conclusions for the two research questions based on the findings of the literature study, the three cases studies and our experiences in providing architectural support.

8.1.1 Search functionality for semantically-rich linked data

Semantically rich graph structures can be used in search functionality to provide effective end-user support when the search algorithm is configured for the specific data and task. In addition, interactive support should be provided for different phases of the search process. We discuss the need for configuration of term search and graph search algorithms, and the support for different types of end-user interaction. Finally, we discuss how the semantic relations provided by RDF(S), OWL and SKOS are used in the explored solutions.

Configuring search algorithms The literature study indicates that generic text-based search functionality can be defined directly on top of the RDF data model (Chapter 2). In the user studies on annotation (Chapter 3) and semantic search (Chapter 5) we showed that this does not, however, provide sufficient support for end-user applications. To provide effective task-specific support search algorithms need to be configured in several dimensions.

- *Effective support for annotation requires different configurations of a term search algorithm for different search fields.* The interface described in the case study on annotation (Chapter 3) required search fields to describe the who, what, where and when depicted on artworks. The fields required different configurations of the search algorithm, for example, different filters had to be provided to constrain the candidate results.
- *Effective support for semantic search requires configurations of a graph search algorithm at different phases of the search process.* In the case study on semantic search (Chapter 5) we derived that the process of searching for artworks contains different phases. Different types of relations in the data are useful in these phases. For example, the initial search results need to include artworks related by literal and object properties as well as equivalence alignments. For query reformulation, however, different types of hierarchical and associative relations between vocabulary terms need to be included.

To support the required search behavior for annotation (Chapter 3) and semantic search (Chapter 5) we implemented configurable algorithms for term and graph search. The algorithms are made available in ClioPatria as parameterized web services (Chapter 6).

Supporting end-user interaction A search process often consists of multiple cycles in which the user tries different queries and explores different search strategies. Interactive solutions can effectively support the user in this process.

- *Faceted browsing provides a generic solution to interactively navigate linked data repositories.* In the literature it is shown that faceted browsing can effectively support the user with the exploration of annotated objects (Chapter 2). In the case study on faceted browsing (Chapter 4) we showed that this interface paradigm can be applied to linked data. In addition, the traditional functionality, which only allows direct constraints on the results, can be extended to support navigation of the graph structure to create indirect constraints.
- *Autocompletion provides effective end-user interaction when searching for specific vocabulary terms.* In the case study on annotation (Chapter 3) autocompletion was successfully used by professional cataloguers to find annotation terms from multiple vocabularies. Autocompletion enabled users

to quickly try multiple queries, as it allowed them to easily switch between scanning the list of results, reducing or extending the query or creating a new query.

- *Effective support for access to semantically-rich linked data requires different types of end-user interaction.* In the case study on annotation (Chapter 3) text-based search alone showed to be not always sufficient to find vocabulary terms. In some cases the cataloguers wanted to navigate the hierarchical structures in which the search results were contained. In the case study on semantic search (Chapter 5), we observed that domain experts require support for multiple search strategies. To support these strategies we identified different types of end-user interaction, for example, query disambiguation by selecting vocabulary terms and query reformulation by navigating the hierarchical structure.

Many reusable interface components that support user interaction are already available on the Web. To apply these components, such as JavaScript widgets, to heterogeneous linked data they have to be configured for the specific domain and task. Typically this requires a developer for the programming and a domain expert that knows how the domain specific data should be used in the component. In Chapter 7 we proposed a method to capture the functionality of interface components in a model. Once such a model is implemented there is no more need for a programmer and the component can be configured by mapping domain specific data to this model.

Using semantics for search functionality We explored the use of three² types of semantic relations available in the data: (i) thesaurus specific relations in the original vocabularies, (ii) alignments between concepts from different vocabularies, (iii) lightweight schema mappings and (iv) ontological descriptions of properties. We discuss how these semantic relations were used in the three case studies.

- *SKOS provides a useful abstraction for vocabularies on which specific functionality can be defined.* As all vocabularies in the data set were modelled or mapped to SKOS, we could use the SKOS relations to provide search functionality for all vocabularies. For example, in the annotation prototype (Chapter 3) the `skos:prefLabel` and `skos:altLabel` provided a generic means to define the values for string matching with the query and define the preferred label in case multiple matches are found. The `skos:broader` relation defines the hierarchical structure of a thesaurus, providing a generic means to support hierarchical navigation of different thesauri. This was used in the prototypes for annotation (Chapter 3), faceted browsing (Chapter 4) and we

²[http://en.wikipedia.org/wiki/The_Spanish_Inquisition_\(Monty_Python\)](http://en.wikipedia.org/wiki/The_Spanish_Inquisition_(Monty_Python))

proposed it as a method for query reformulation in an interactive semantic search application (Chapter 5).

- *Alignment relations allow the inclusion of data from external sources, increasing recall in text-based search.* In our data, equivalent terms from different vocabularies are aligned using the `skos:exactMatch` and `owl:sameAs` relations. In the search functionality these equivalence relations allow information from external sources to be included. For example, in the case study on annotation (Chapter 3), the alignment with external vocabularies increased the available information by which the terms from in-house thesauri could be found. They provided useful spelling variations, synonyms, nicknames and multiple languages. In the user study on semantic search (Chapter 5), the experts also indicated a need for the integration of information from external vocabularies. Here they provided additional literals as well as useful relations between terms for query reformulation.

In our work we only considered equivalence alignments. Other alignment relations, such as `skos:closeMatch` and `skos:broaderMatch`, also allow the inclusion of external information, but functionality should consider their associative and hierarchical nature.

- *Schema mappings enable integrated access to heterogeneous data while preserving the richness of the individual collections and vocabularies.* Instead of a single unifying data model, the different schemata of the collections and vocabularies are aligned by lightweight mappings with `rdfs:subPropertyOf` and `rdfs:subClassOf` relations. In the search functionality these mapping relations enable integrated access. For example, in the faceted browsing prototype (Chapter 4) the mappings from the collection specific properties to a common super property enabled integrated access. For example, a facet corresponding to the `dc:creator` property provides integrated access to the large number of specific creator properties from the different collections. At the same time the rich information was maintained by allowing the user to still select a facet corresponding to a collection specific property.
- *Meta properties of the relations in the data enable integrated search functionality over heterogeneous linked data.* The schemata in our data set contains “meta” properties, such as `owl:inverseOf` and `owl:symmetricProperty`. These are useful for data integration purposes. For example, the hierarchical relations are defined by some organisations using `skos:broader`, while others use `skos:narrower`. SKOS defines that these two are each other inverse, using the “meta” property `owl:inverseOf`. Applications only need to support the semantics of such “meta” properties to provide integrated search functionality.

8.1.2 Presenting the results found in heterogeneous linked data

Semantically rich graph structures can be used in the result presentation to provide effective end-user support when the presentation information and organisation method is configured for the specific data and task. In addition, appropriate abstractions in the data are required to make the large diversity in heterogeneous data manageable. We discuss the need for configuration of result organisation and presentation methods, and the need for common abstractions in the data. Finally, we discuss how the semantic relations can be used to support such configuration and abstraction.

Configuring presentation algorithms Linked data contains various characteristics and many specific RDF properties that can be used for the organisation and presentation of the search results. To provide task-specific support different methods are required for different types of terms.

- *Unambiguous presentation of vocabulary terms requires different types of additional information for different types of terms.* In the case study on annotation (Chapter 3) it became clear that a label alone was not always sufficient to disambiguate different vocabulary terms and additional information had to be presented. The required information varied for different types of terms. For example, profession, nationality and birth/death dates for people, and the country and place type for geographical locations.
- *Search results from different vocabularies require different organisation strategies.* In the case study on annotation (Chapter 3) it became clear that professional cataloguers prefer different sorting and grouping strategies for different annotation fields. For example, they preferred to sort people alphabetically, as this is a natural and transparent ordering for names. For ICONCLASS they preferred presentation of the concepts towards the top of the hierarchy first, as they are used to navigating from these to more specific concepts. Searching in multiple thesauri, e.g in WORDNET and ICONCLASS, the presentation of the results in separate groups for each vocabulary helps to distinguish between different types of term.

To support configurable result organisation and presentation the web service for term search had to be extended (Chapter 6). In addition, the organisation and presentation dimensions had to be included in the RDF model describing the auto-completion widget. This provides a solution to configure the selection, organisation and visualisation of the widget in a single configuration file (Chapter 7).

Providing appropriate abstractions Heterogeneous data contains a large number of different types of resources and relations. To avoid overwhelming the

user, this diversity should not always be presented. For a specific domain, appropriate abstractions, which match with the conceptual model of the user, should be identified. In our studies on the cultural heritage domain artefacts, persons, locations, events and domain specific concepts were reoccurring abstractions. These abstractions should be used in the presentation of the navigation paths and search results to provide a manageable and coherent view.

- *For a specific task a small number of abstract facets should be presented to the user.* For non-trivial data sets a generic solution for faceted browsing will result in a very large number of facets. The `rdfs:subPropertyOf` relations in the data provide a means to organise the facets and provide the user with a smaller number of abstract facets (Chapter 4). To support a specific task the existing properties in the data might, however, not match with the user's conceptualisation of the domain. In this case, appropriate end-user facets should be identified and manually configured using mappings to the underlying data.
- *Effective presentation of the search results requires the identification of common abstractions in the domain that cover the richness of the data.* In the user study on semantic search (Chapter 5), the domain experts indicated that their assessment of the search results depends on the relation to the query. Results found by a literal property need manual assessment, whereas the results found by a term from controlled vocabulary are trusted by the relation to this term. The individual collections contain a large number of different types of relations by which artworks can be related to the query, e.g. many specific types of creator relations. In the presentation of the initial search results this level of detail should be hidden from the user. Instead, a number of common high-level properties in the domain should be used, e.g. creator and subject, to provide a simple unified view on the results.

In the analysis of the graph search algorithm for the search log queries (Chapter 5), a large number of related vocabulary terms were found as potential candidates for query reformulation. The association relations by which these terms are found and the classes they belong to provide a means to organise the suggestions for query reformulation. Using the right abstractions for these properties and classes a large number of suggestions for query reformulation can be organised while providing an intuitive navigation structure.

Using semantics for organisation and presentation In the result presentation we explored the use of the same four types of semantic relations available in the data: (i) thesaurus specific relations in the original vocabularies, (ii) equivalence alignments between concepts from different vocabularies (iii) lightweight schema mappings and (iv) ontological descriptions of properties.

- *SKOS provides a useful abstraction for vocabularies on which specific presentation methods can be defined, but the use of domain specific vocabularies is still required.* In the case study on annotation (Chapter 3), the presentation properties of the results were partially available as the values of SKOS properties, e.g. the preferred and alternative labels, a description and the hierarchical structure. To this extent a generic presentation solution could be provided. For the persons and locations, however, additional properties were required, such as the profession, nationality and birth/death dates for people. As different vocabularies provided their own types of properties for these values, mappings were required between these properties from different vocabularies.
- *Equivalence alignments should be used to remove duplicate search results.* In the case study on annotation (Chapter 3), similar concepts from different vocabularies need to be presented as a single search result to prevent duplicate results in the interface. The study showed that for this search task a conservative alignment method is most suited. Incorrect alignments are harmful, because they remove possible candidates from the search results whereas a few duplicates in the interface are acceptable.
- *Hierarchical relations between different schema provide a useful dimension to provide different abstractions of the result presentation.* In the case study on faceted browsing (Chapter 4), it was shown that the `rdfs:subPropertyOf` relations can be used as a dimension where along different abstractions of the available facets can be presented. This enables the user to interactively choose the facets at the appropriate level of abstraction for the specific tasks. For example, the facets capturing the common abstractions in the domain provide a high level integrated view on all collections, while the existing properties from the individual collections provide a detailed view but only applies to a single collection.

In (Chapter 6) we showed how the paths in the graph, indicating the relation to the query, can be used to group similar types of results together. Using the collection and vocabulary specific properties and classes many queries result in many different types of paths. Abstracting the path using both the `rdfs:subPropertyOf` and the `rdfs:subClassOf` relations enables groups of results at different levels of abstraction.

- *Meta properties of the relations in the data enable a unified view over heterogeneous linked data.* The “meta” properties of the relations were used in a similar fashion as in the search functionality, but in this case to generate the presentation structures. For example, in the presentation of a single SKOS concept, the use of `owl:inverseOf` guarantees that the view on the concept

properties is independent of modelling decisions with respect to `skos:broader` versus `skos:narrower`.

8.2 Discussion and future research

We identify three limitations in research described in this thesis: (I) the studies are performed only in a single domain, (II) we focussed only on support for domain experts (III) the studies only provide qualitative evaluations. We discuss each limitation in turn. In addition, we discuss the required support for representing the semantics used in this research.

Application to other domains A limitation of the research described in this thesis is that all studies are performed in the cultural heritage domain. We can not make general claims about the applicability of our solutions to other domains. We expect, however, that our solutions can be used to support end-users in other domains where objects are described with terms from structured vocabularies. Evidence for this is provided by the application of the software described in this thesis in the news domain (Troncy 2008), the music domain (Raimond and Sandler 2008) and for historical events (Shaw et al. 2009). At the time of writing, the software is primarily used to showcase linked data in these other domains. To provide effective end-user support for annotation, faceted browsing and semantic search in these domains, further research is required. We expect that it will be sufficient to model the common abstractions in the domain and configure the search algorithm and presentation methods. Finding the appropriate abstractions and configurations for a particular domain will require experimentation with the end-users in this domain.

For some of the collections shared as open linked data on the Web, the objects are not described with terms from vocabularies. In general, our solutions can still be applied to access such collections. For example, the objects can be found by their direct metadata with text-based search or by faceted browsing. The specific search functionality and result presentation methods that are enabled by the semantic relations are, however, not available. By enriching the metadata of these collections additional functionality can be enabled. For example, a literal value describing the creation site of an object can be replaced with a reference to a term from a geographical vocabulary. This makes the semantics from the external source available when accessing the collection, e.g. enabling access through the geographical containment relations.

Support for domains with stronger ontological relations and/or rules was not within the scope of this thesis. Exploiting the added value of formal reasoning in end-user tasks such as annotation, search and faceted browsing requires further research. On the other hand, we expect that in domains with strong ontological

commitments there will also be different end-user tasks than the search oriented tasks considered in this thesis, requiring altogether different support.

Broader user population A limitation of the studies performed in this research is that they focused on support for expert users. We cannot make general claims about the applicability of our solutions to other types of users. We expect, however, that some of our solutions can be applied to support other user populations, but that the configuration of the search functionality and the result presentation requires adaptation to their specific needs.

In the case study on annotation (Chapter 3), we investigated support for domain experts when creating high quality descriptions. To support annotation by the general public, future research is needed to better understand issues around quality control, the type of annotations the general public would want to make and the effort they would put into selecting vocabulary terms. We found some indication that users that are less experienced in annotation cannot be expected to carefully select the most appropriate term when multiple options are available. Two of the study participants, who only occasionally provide artwork annotations, often selected the first likely candidate without considering other options. The end-user support should take this type of behavior into consideration.

In the case studies on search (Chapter 5) and browsing (Chapter 4) we explored data-driven approaches. This assumes that users are familiar with the representation of domain-specific knowledge, as this is directly exposed in the user interface. We cannot expect all types of users to be familiar with highly specific representations. We believe that the solutions we explored can deal with this aspect. We identified the need for the use of common abstractions in the user interface. We focused on this need to handle the heterogeneity of the data. This, however, also provides a solution to adapt the information that is presented to the user at an appropriate level. We experienced that these abstractions can themselves be modelled in linked data and thus automatically appear in the data driven solutions. In addition, they can be used in the configuration of the interface components, as explained in Chapter 7.

Towards quantitative evaluation In our studies we limited the evaluations to qualitative analysis. Without commonly agreed upon evaluation methods or benchmarks we can not make any claims about the comparison to other semantic search systems. Working towards commonly agreed upon evaluation methods is, however, not straightforward. The effectiveness of semantic search systems for a specific task is affected by the quality and the modelling of the data, the behavior of the algorithms and the design of the user interface. In addition, semantic systems provide support in different stages of the search process: query formulation, search algorithm and result presentation.

Based on the conclusions of this thesis we expect that comparative studies are

best performed using a specific type of functionality and considering a specific stage of the search process. We hope that the community works in this direction, for example in initiatives, such as the evaluation track of the Semantic Search 2010³ workshop.

RDFS plus? The semantics used in our data set correspond to a subset of what Allemang and Hendler (Allemang and Hendler 2008) defined as “RDFS *plus*”: a subset of RDFS/OWL that provides sufficient utility for data integration and is computationally practical to apply. Our experiences confirm that in a domain with collections of objects annotated with terms from structured vocabularies, RDFS *plus* combined with SKOS provides sufficient utility for integrated access in the tasks we studied. Furthermore, computation with these semantic relations could be performed on the fly the performance required for interactive applications. For our purposes the following was sufficient:

- **rdf:type** to provide typing information. For vocabulary terms it is useful to have more information than **skos:Concept** alone, such as **Person**. In the search functionality the types of objects and vocabulary terms help to restrict and in the presentation to explain the results.
- **rdfs:subClassOf** and **rdfs:subPropertyOf** to define lightweight mappings between different schemata. In the search functionality this enables integrated access. In the presentation it provides a unified view on the results and enables different levels of abstraction.
- **rdf:value** to represent default values in N-ary relations. In the search functionality it indicates that the value can be considered as a direct object property. In the presentation it indicates what should be shown to the user.
- **skos:broader** and **skos:narrower** to define hierarchical relations between vocabulary terms. When more detailed hierarchical relations are already available they should be mapped to the relations in SKOS. In search functionality this enables specialisation and generalisation. In the presentation it provides a means to organise the results in an hierarchical structure.
- **skos:related** to define associations between vocabulary terms. When more detailed association relations are already available they should be mapped to the relation in SKOS. In text-based search functionality this enables suggestions for query reformulation and in the formulation of structured queries to define indirect constraints.
- **skos:exactMatch** and **owl:sameAs** to define equivalence between resources. In search functionality this enables the inclusion of information from external sources. In the result presentation it allows the removal of duplicate terms.

³<http://km.aifb.uni-karlsruhe.de/ws/semsearch10/>

- `owl:inverseOf` and `owl:symmetricProperty` to define characteristics of relations. In search functionality and result presentation this enables the unification of modelling decisions.

Although we used transitive reasoning in the back-end, for example, in the facet browser (Chapter 4) to find all objects transitively related to a selected value, we did not consider it as a characteristic of the data. In our experiences the use of transitivity varied per search functionality and task, instead for the type of data. Furthermore, in the result presentation transitivity was used in the opposite direction to find all ancestors for a given set of resources.

8.3 Looking ahead

Linked data on the Web has become reality. In 2009 both the US⁴ and UK⁵ governments have started to share information on the Web and parts of this are being published according to the principles of linked open data. Publishing this data is, however, only the start. Designing the applications so end-users can benefit from all this data is still a grand challenge. The work described in this research has explored several aspects of this challenge. Above all, it has made clear that there is no one-size-fits-all solution to access linked data. Instead, specific tasks require different types of search functionality and result presentation methods and these need to be carefully configured to provide effective end-user support.

Although much work lies ahead, we believe that with the right web services in place, and configurable interface components at our disposal end-user access to heterogeneous linked data has become within reach. Where on Web 2.0 a programmer was required to create a Mash-up, integrated access on Web 3.0 will follow from the relations in the data itself.

⁴<http://www.data.gov/>

⁵<http://data.gov.uk/>

References

- Allemang, D. and J. Hendler (2008). *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL*. Burlington, MA: Morgan Kaufmann.
- Amin, A., L. Hardman, J. van Ossenbruggen, and A. van Nispen (2008). Understanding cultural heritage experts' information seeking tasks. In *JCDL '08: Proceedings of the Joint Conference of Digital Library*, New York, NY, USA. ACM Press.
- Amin, A., M. Hildebrand, J. van Ossenbruggen, V. Evers, and L. Hardman (2009, April 6-9). Organizing suggestions in autocompletion interfaces. In *31st European Conference on Information Retrieval*, Toulouse, France. to be published, based on techreport: <http://ftp.cwi.nl/CWIreports/INS/INS-E0901.pdf>.
- Anyanwu, K., A. Maduko, and A. Sheth (2005). SemRank: Ranking Complex Relationship Search Results on the Semantic Web. In *Proceedings of the 14th international conference on World Wide Web*, Chiba, Japn, pp. 117–127. ACM Press.
- Aroyo, L., N. Stash, Y. Wang, P. Gorgels, and L. Rutledge (2007, November). Chip demonstrator: Semantics-driven recommendations and museum tour generation. In *The Semantic Web - ISWC 2007*.
- Auer, S., S. Dietzold, and T. Riechert (2006). OntoWiki -A Tool for Social, Semantic Collaboration. In *Proceedings of the Fifth International Semantic Web Conference ISWC 2006*, Athens, GA, USA.
- Baeza-Yates, R. A. and B. A. Ribeiro-Neto (1999). *Modern Information Retrieval*. ACM Press / Addison-Wesley.
- Bast, H. and I. Weber (2007). The CompleteSearch Engine: Interactive, Efficient, and towards IR&DB Integration. In *CIDR 2007, Third Biennial Conference on Innovative Data Systems Research*, Asilomar, CA, USA, pp. 88–95.

- Berlin, F. U. (2007). Search DBpedia.org. <http://dbpedia.org/search/>.
- Berners-Lee, T., Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer, and D. Sheets (2006). Tabulator: Exploring and Analyzing linked data on the Semantic Web. In *The 3rd International Semantic Web User Interaction Workshop (SWUI06)*, Athens, Georgia, USA.
- Bernstein, A., E. Kaufmann, C. Kaiser, and C. Kiefer (2006). Ginseng: A Guided Input Natural Language Search Engine for Querying Ontologies. In *Jena User Conference*, Bristol, UK.
- Binding, C. and D. Tudhope (2004a). KOS at your Service: Programmatic Access to Knowledge Organisation Systems. *Journal of Digital Information* 4(4).
- Binding, C. and D. Tudhope (2004b). KOS at your service: Programmatic access to knowledge organisation systems. *J. Digit. Inf.* 4(4).
- Bizer, C. and T. Gauß (2007). Disco - Hyperdata Browser. <http://sites.wiwiss.fu-berlin.de/suhl/bizer/ng4j/disco/>.
- Bizer, C., R. Lee, and E. Pietriga (2005, November 6–10,). Fresnel — A Browser-Independent Presentation Vocabulary for RDF. In *Proceedings of the Second International Workshop on Interaction Design and the Semantic Web*, Galway, Ireland.
- Bonino, D., F. Corno, L. Farinetti, and A. Bosca (2004, December). Ontology Driven Semantic Search. *WSEAS Transaction on Information Science and Application* 1(6), 1597–1605.
- Broekstra, J., A. Kampman, and F. van Harmelen (2002). Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In I. Horrocks and J. Hendler (Eds.), *The Semantic Web - ISWC 2002*, Number 2342 in Lecture Notes in Computer Science, Berlin Heidelberg, pp. 54–68. Springer.
- Celino, I., E. D. Valle, D. Cerizza, and A. Turati (2006). Squiggle: a Semantic Search Engine for indexing and retrieval of multimedia content. In *Proceedings of the First International Workshop on Semantic-enhanced Multimedia Presentation Systems (SEMPS 2006)*, Athens, Greece, pp. 20–34.
- Cheng, G., W. Ge, H. Wu, and Y. Qu (2008). Searching semantic web objects based on class hierarchies. In *WWW 2008 Workshop on Linked Data on the Web*.
- Clark, K. G., L. Feigenbaum, and E. Torres (2007, 18 June). Serializing sparql query results in json. W3C Working Group Note.
- Davies, J. and R. Weeks (2004). QuizRDF: Search Technology for the Semantic Web. In *Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 4 - Volume 4*.

- de Boer, V., M. van Someren, and B. J. Wielinga (2007). A redundancy-based method for the extraction of relation instances from the web. *Int. J. Hum.-Comput. Stud.* 65(9), 816–831.
- DERI (2005-2007). SWSE. <http://www.swse.org/>.
- Ding, L., T. Finin, A. Joshi, R. Pan, R. S. Cost, Y. Peng, P. Reddivari, V. C. Doshi, , and J. Sachs (2004). Swoogle: A Search and Metadata Engine for the Semantic Web. In *Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management*, Washington, D.C., USA, pp. 652–659.
- Ding, L., R. Pan, T. Finin, A. Joshi, Y. Peng, and P. Kolari (2005, November 6–10,). Finding and Ranking Knowledge on the Semantic Web. See Gil, Motta, Benjamins, and Musen (2005), pp. 156–170.
- Duke, A., T. Glover, and J. Davies (2007). Squirrel: An Advanced Semantic Search and Browse Facility. In *Proceedings of the European Semantic Web Conference ESWC 2007*, Innsbruck, Austria.
- Fluit, C., M. Sabou, and F. van Harmelen (2003a). Supporting User Tasks through Visualisation of Light-weight Ontologies. In S. Staab and R. Studer (Eds.), *Handbook on Ontologies in Information Systems*, pp. 415–434. Springer-Verlag.
- Fluit, C., M. Sabou, and F. van Harmelen (2003b). *Visualizing the Semantic Web: XML-based Internet and Information Visualization*, Chapter Ontology-based Information Visualization, pp. 36–48. Springer-Verlag.
- Geroimenko, V. and C. Chen (2003). *Visualizing the Semantic Web — XML-based Internet and Information Visualization*. Springer-Verlag.
- Gil, Y., E. Motta, V. R. Benjamins, and M. A. Musen (Eds.) (2005, November 6–10,). *4th International Semantic Web Conference, ISWC 2005*, Lecture Notes in Computer Science, Galway, Ireland. Springer.
- Group, X. (2005-2006). Falcon-S. <http://www.falcons.com.cn/>.
- Guha, R., R. McCool, and E. Miller (2003). Semantic Search. In *Proceedings of the 12th international conference on World Wide Web*, Budapest, Hungary, pp. 700–709.
- Hearst, M. A. (2006). Clustering versus faceted categories for information exploration. *Commun. ACM* 49(4), 59–61.
- Heflin, J. and J. Hendler (2000). Searching the Web with SHOE. In *Artificial Intelligence for Web Search. Papers from the AAAI Workshop*, Menlo Park, CA, pp. 35–40.
- Hendler, J. (2010). Web 3.0: The dawn of semantic search. *Computer* 43(1), 77–80.

- Hildebrand, M. and J. van Ossenbruggen (2009, February). Configuring semantic web interfaces by data mapping. In *Workshop on Visual Interfaces to the Social and the Semantic Web, IUI'09*.
- Hildebrand, M., J. van Ossenbruggen, A. Amin, L. Aroyo, J. Wielemaker, and L. Hardman (2007, November). The design space of a configurable autocompletion component. Technical Report INS-E0708, CWI.
- Hildebrand, M., J. van Ossenbruggen, and L. Hardman (2006, November). /facet: A Browser for Heterogeneous Semantic Web Repositories. In *The Semantic Web - ISWC 2006*, pp. 272–285.
- Hildebrand, M., J. van Ossenbruggen, and L. Hardman (2007, July). An analysis of search-based user interaction on the Semantic Web. Technical Report INS-E0706, CWI.
- Hildebrand, M., J. van Ossenbruggen, and L. Hardman (2010). *Multimedia Semantics: Metadata, Analysis and Interaction*, Chapter The role of explicit semantics in search and browsing. Wiley. To be published.
- Hildebrand, M., J. R. van Ossenbruggen, L. Hardman, and G. Jacobs (2009, October). Supporting Subject Matter Annotation Using Heterogeneous Theasauri, A User Study In Web Data Reuse. *International Journal of Human-Computer Studies* 67(10), 888 – 903.
- Hogan, A., A. Harth, and S. Decker (2006). ReConRank: A Scalable Ranking Method for Semantic Web Data with Context. In *2nd Workshop on Scalable Semantic Web Knowledge Base Systems*.
- Hollink, L., G. Schreiber, J. Wielemaker, and B. Wielinga (2003). Semantic annotation of image collections. In *Second International Conference for Knowledge Capture (KCAP2003) Workshop on Knowledge Markup and Semantic Annotation*, pp. 0–3.
- Hollink, L., G. Schreiber, and B. Wielinga (2007). Patterns of semantic relations to improve image content search. *Journal of Web Semantics* 5(3), 195–203.
- Huynh, D., D. Karger, and R. Miller (2007). Exhibit: Lightweight structured data publishing. In *16th International World Wide Web Conference*, Banff, Alberta, Canada. ACM.
- Huynh, D., S. Mazzocchi, and D. R. Karger (2005, November 6–10.). Piggy Bank: Experience the Semantic Web Inside Your Web Browser. See Gil, Motta, Benjamins, and Musen (2005), pp. 413–430.
- Hyvönen, E., M. Junnila, S. Kettula, E. Mäkelä, S. Saarela, M. Salminen, A. Syreeni, A. Valo, and K. Viljanen (2005, October). MuseumFinland — Finnish museums on the semantic web. *Journal of Web Semantics* 3(2-3), 224–241.

- Hyvönen, E. and E. Mäkelä (2006). Semantic Autocompletion. In *Proceedings of the 1st Asian Semantic Web Conference (ASWC 2006)*, Beijing, pp. 739–751.
- Hyvönen, E., K. Viljanen, J. Tuominen, and K. Seppälä (2008, June 1-5). Building a national semantic web ontology and ontology service infrastructure—the finnonto approach. In *Proceedings of the European Semantic Web Conference ESWC 2008*, pp. 95–109. Springer.
- Joho, H., C. Coverson, M. Sanderson, and M. Beaulieu (2002). Hierarchical presentation of expansion terms. In *SAC '02: Proceedings of the 2002 ACM symposium on Applied computing*, New York, NY, USA, pp. 645–649. ACM.
- Kiryakov, A., B. Popov, I. Terziev, D. Manov, and D. Ognyanoff (2004, December). Semantic Annotation, Indexing, and Retrieval. *Journal of Web Semantics* 2(1), 49–79.
- Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM* 46(5), 604–632.
- Kobilarov, G., T. Scott, Y. Raimond, S. Oliver, C. Sizemore, M. Smethurst, C. Bizer, and R. Lee (2009). Media meets semantic web - how the bbc uses dbpedia and linked data to make connections. In L. Aroyo, P. Traverso, F. Ciravegna, P. Cimiano, T. Heath, E. Hyvönen, R. Mizoguchi, E. Oren, M. Sabou, and E. P. B. Simperl (Eds.), *ESWC*, Volume 5554 of *Lecture Notes in Computer Science*, pp. 723–737. Springer.
- Koenemann, J. and N. J. Belkin (1996). A case for interaction: a study of interactive information retrieval behavior and effectiveness. In *CHI '96: Proceedings of the SIGCHI conference on Human factors in computing systems*, New York, NY, USA, pp. 205–212. ACM.
- Lei, Y., V. Uren, and E. Motta (2006). SemSearch: A Search Engine for the Semantic Web. In *15th International Conference on Knowledge Engineering and Knowledge Management: Managing Knowledge in a World of Networks (EKAW 2006)*, Podebrady, Czech Republic.
- Lopez, V., M. Pasin, and E. Motta (2005). AquaLog: An Ontology-portable Question Answering System for the Semantic Web. In *Proceedings of the European Semantic Web Conference ESWC 2005*, Crete, Greece, pp. 546–562.
- Magennis, M. and C. J. van Rijsbergen (1997). The potential and actual effectiveness of interactive query expansion. In *SIGIR '97: Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, New York, NY, USA, pp. 324–332. ACM.
- Mäkelä, E., K. Viljanen, O. Alm, J. Tuominen, O. Valkeapää, T. Kauppinen, J. Kurki, R. Sinkkilä, T. Käsälä, R. Lindroos, O. Suominen, T. Ruotsalo,

- and E. Hyvönen (2007, November 11). Enabling the semantic web with ready-to-use web widgets. In *Proceedings of the First Industrial Results of Semantic Technologies Workshop, ISWC2007*.
- m.c. schraefel and D. Karger (2006). The Pathetic Fallacy of RDF. In *The 3rd International Semantic Web User Interaction Workshop (SWUI06)*.
- m.c. schraefel, D. A. Smith, A. Owens, A. Russell, C. Harris, and M. L. Wilson (2005). The evolving mSpace platform: leveraging the Semantic Web on the Trail of the Memex. In *Proceedings of Hypertext 2005*, Salzburg, pp. 174–183.
- Meij, E., M. Bron, L. Hollink, B. Huurnink, and M. de Rijke (2009, October). Learning semantic query suggestions. In *8th International Semantic Web Conference (ISWC 2009)*.
- Metaweb (2007). Freebase. <http://www.freebase.com/>.
- Mika, P. (2005, October). Flink: Semantic Web Technology for the Extraction and Analysis of Social Networks. *Journal of Web Semantics* 3(2-3), 211–223.
- Mika, P. (2006). OpenAcademia. <http://www.openacademia.org/>.
- Miles, A. and S. Bechhofer (2009, August 18). Skos simple knowledge organization system reference. W3C Recommendation.
- Oren, E., R. Delbru, and S. Decker (2006). Extending Faceted Navigation for RDF Data. In *5th International Semantic Web Conference*, Athens, GA, USA, pp. 559–572.
- Page, L., S. Brin, R. Motwani, and T. Winograd (1998). The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project.
- Paulson, L. D. (2005). Building Rich Web Applications with Ajax. *IEEE Computer* 38(10), 14–17.
- Pietriga, E., C. Bizer, D. R. Karger, and R. Lee (2006). Fresnel: A browser-independent presentation vocabulary for RDF. In I. F. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo (Eds.), *International Semantic Web Conference*, Volume 4273 of *Lecture Notes in Computer Science*, pp. 158–171. Springer.
- Porter, M. F. (1980). An algorithm for suffix stripping. *Program* 14(3), 130–137.
- Quan, D. and D. R. Karger (2004, May 17-22,). How to Make a Semantic Web Browser. In *The Thirteenth International World Wide Web Conference*, New York City. IW3C2: ACM Press.
- Raimond, Y. and M. Sandler (2008). A web of musical information. In *Proceedings of the International Conference on Music Information Retrieval 2008*, pp. 263–268.

- Rocha, C., D. Schwabe, and M. de Aragao (2004). A hybrid approach for searching in the semantic web. In *Proceedings of the 13th International World Wide Web Conference*, New York, NY, USA, pp. 374–383.
- Ruthven, I. (2003). Re-examining the potential effectiveness of interactive query expansion. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, New York, NY, USA, pp. 213–220. ACM.
- Rutledge, L., J. van Ossenbruggen, and L. Hardman (2005, May). Making RDF Presentable – Integrated Global and Local Semantic Web Browsing. In *The Fourteenth International World Wide Web Conference*, Chiba, Japan, pp. 199–206. IW3C2: ACM Press.
- Schreiber, G., A. Amin, L. Aroyo, M. van Assem, V. de Boer, L. Hardman, M. Hildebrand, B. Omelayenko, J. van Ossenbruggen, A. Tordai, J. Wielemaker, and B. J. Wielinga (2008). Semantic annotation and search of cultural-heritage collections: The multimedial e-culture demonstrator. *J. Web Sem.* 6(4), 243–249.
- Schreiber, G., A. Amin, M. van Assem, V. de Boer, L. Hardman, M. Hildebrand, L. Hollink, Z. Huang, J. van Kersen, M. de Niet, B. Omelayenjkko, J. van Ossenbruggen, R. Siebes, J. Taekema, J. Wielemaker, and B. Wielinga (2006, November). MultimediaN E-Culture Demonstrator. In *The Semantic Web - ISWC 2006*, pp. 951–958.
- Shaw, R., R. Troncy, and L. Hardman (2009). Lode: Linking open descriptions of events. In A. Gómez-Pérez, Y. Yu, and Y. Ding (Eds.), *ASWC*, Volume 5926 of *Lecture Notes in Computer Science*, pp. 153–167. Springer.
- SIMILE (2003-2005). Longwell RDF Browser. <http://simile.mit.edu/longwell/>.
- Sinkkilä, R., E. Mäkelä, E. Hyvönen, and T. Kauppinen (2008). Combining context navigation with semantic autocompletion to solve problems in concept selection. In K. Belhajjame, M. d’Aquin, P. Haase, and P. Missier (Eds.), *SeMMA*, Volume 346 of *CEUR Workshop Proceedings*, pp. 61–68. CEUR-WS.org.
- Suominen, O., K. Viljanen, and E. Hyvönen (2007a). User-centric Faceted Search for Semantic Portals. In *Proceedings of the European Semantic Web Conference ESWC 2007*, Innsbruck, Austria.
- Suominen, O., K. Viljanen, and E. Hyvönen (2007b, June 4-5). User-centric faceted search for semantic portals. In *Proceedings of the European Semantic Web Conference ESWC 2007*, Innsbruck, Austria. Springer.
- Sure, Y. and V. Iosif (2000). First results of a Semantic Web Technologies Evaluation. In *Proceedings of the Common Industry Program*, University of California, Irvine.

- Tordai, A., B. Omelayenko, and G. Schreiber (2007). Thesaurus and metadata alignment for a semantic e-culture application. In *K-CAP '07: Proceedings of the 4th international conference on Knowledge capture*, New York, NY, USA, pp. 199–200. ACM.
- Tordai, A., J. R. van Ossenbruggen, and G. Schreiber (2009, September). Combining Vocabulary Alignment Techniques. In *Proceedings of The Fifth International Conference on Knowledge Capture* (first ed.). IAAA.
- Trant, J. (2006). Understanding searches of a contemporary art museum catalogue: A preliminary study. <http://tinyurl.com/yl5l1tk>.
- TRECVID organizers (2007, April 18). Guidelines for the TRECVID 2007 Evaluation. <http://www-nlpir.nist.gov/projects/tv2007/tv2007.html>.
- Troncy, R. (2008, November). Bringing the iptc news architecture into the semantic web. In A. P. Sheth, S. Staab, M. Dean, M. Paolucci, D. Maynard, T. W. Finin, and K. Thirunarayan (Eds.), *International Semantic Web Conference*, Volume 5318 of *Lecture Notes in Computer Science*, Berlin Heidelberg, pp. 483–498. Springer.
- Tummarello, G., C. Morbidoni, and M. Nucci (2006). Enabling Semantic Web communities with DBin: an overview. In *Proceedings of the Fifth International Semantic Web Conference ISWC 2006*, Athens, GA, USA.
- Tummarello, G., E. Oren, and R. Delbru (2007, November). Sindice.com: Weaving the open linked data. In *Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference (ISWC/ASWC2007)*, Busan, South Korea, Berlin, Heidelberg, pp. 547–560.
- Uschold, M. (2003, September). Where are the semantics in the semantic web? *AI Magazine* 24(3), 25–36.
- van Assem, M., A. Gangemi, and G. Schreiber (2006, May). Conversion of WordNet to a standard RDF/OWL representation. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy.
- van Assem, M., M. R. Menken, G. Schreiber, J. Wielemaker, and B. Wielinga (2004, November). A Method for Converting Thesauri to RDF/OWL. In *Proceedings of the Third International Semantic Web Conference (ISWC'04)*, Number 3298 in *Lecture Notes in Computer Science*, Hiroshima, Japan, pp. 17–31. Springer.
- W3C (1999, February 22,). Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendations are available at <http://www.w3.org/TR>.
- W3C (2004, 10 February). Web Ontology Language (OWL) - Overview. W3C Recommendation.

- W3C (2005, November 2.). SKOS Core Guide, W3C Working Draft. W3C Candidate Recommendations are available at <http://www.w3.org/TR>. Edited by Alistair Miles and Dan Brickley.
- W3C (2006, April 6.). SPARQL Query Language for RDF. W3C Candidate Recommendations are available at <http://www.w3.org/TR>. Edited by Eric Prud'hommeaux and Andy Seaborne.
- W3C (2007, August 14.). Image Annotation on the Semantic Web. Multimedia Semantics Incubator Group Report (XGR).
- Wielemaker, J. (2005a, October). An optimised Semantic Web query language implementation in Prolog. In *ICLP 2005*, pp. 128–142. LNCS 3668.
- Wielemaker, J. (2005b, October). An optimised semantic web query language implementation in prolog. In M. Baggielli and G. Gupta (Eds.), *ICLP 2005*, Berlin, Germany, pp. 128–142. Springer Verlag. LNCS 3668.
- Wielemaker, J. (2009a). *Logic programming for knowledge-intensive interactive applications*. Phd thesis, Universiteit van Amsterdam.
- Wielemaker, J. (2009b). *Logic programming for knowledge-intensive interactive applications*. Ph. D. thesis, University of Amsterdam. <http://dare.uva.nl/en/record/300739>.
- Wielemaker, J., M. Hildebrand, and J. van Ossenbruggen (2007). Prolog as the Fundament for Applications on the Semantic Web. In *Proceedings of the ICLP'07 Workshop on Applications of Logic Programming to the Web, Semantic Web and Semantic Web Services (ALPSWS2007)*, Porto, Portugal.
- Wielemaker, J., M. Hildebrand, J. van Ossenbruggen, and G. Schreiber (2008, November). Thesaurus-based search in large heterogeneous collections. In A. P. Sheth, S. Staab, M. Dean, M. Paolucci, D. Maynard, T. W. Finin, and K. Thirunarayan (Eds.), *International Semantic Web Conference*, Volume 5318 of *Lecture Notes in Computer Science*, Berlin Heidelberg, pp. 695–708. Springer.
- Wielemaker, J., Z. Huang, and L. van der Meij (2008). Swi-prolog and the web. *Theory and Practice of Logic Programming* 8(3), 363–392.
- Wielemaker, J., G. Schreiber, and B. Wielinga (2003, October 20-23.). Prolog-Based Infrastructure for RDF: Scalability and Performance. In *The Semantic Web - ISWC 2003*, Sanibel Island, Florida, USA, pp. 644–658. Springer-Verlag Heidelberg.
- Yee, K.-P., K. Swearingen, K. Li, and M. Hearst (2003). Faceted Metadata for Image Search and Browsing. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, Ft. Lauderdale, Florida, USA, pp. 401–408. ACM Press.

Summary

On the Web, organisations are opening up their data and services for re-use. This openness allows information from different sources to be combined, enabling access in ways unforeseen by the original data providers. The Web applications that combine different data sources and services, also known as mash-ups, have become a common solution to support specific end-user tasks. Creating these web applications, however, requires a developer to provide precise instructions on how to use and integrate the data from different sources.

Semantic Web technologies promise to simplify reuse of data from different sources. The Semantic Web representation languages provide a standardised way to model and share knowledge on the Web. In addition, these languages allow aspects of the semantics available in the data to be made explicit in a machine-accessible way, enabling intelligent technologies to automatically infer how data should be used and integrated.

A growing number of machine-accessible data sets is now being published and linked together on the Web. The result is a large graph of linked data, describing a variety of different types of objects. In a survey study, we analyse applications that provide access to this heterogeneous linked data. From this study we conclude that Semantic Web applications support a wide variety of different types of tasks and provide access to different types of data sets. It, however, remains unclear how well these semantic technologies improve support for end-users, as commonly agreed upon evaluation methods are lacking. Given a specific domain and search task, it is thus difficult to determine how the semantics in the data can be used to benefit the end user.

Within this thesis, cultural heritage is chosen as an application domain to study end-user support for access to heterogeneous linked data. The semantically-rich Web of culture data created in the MultimediaN E-Culture project is used as a data set. The practical results of this research are made available as applications and web services in ClioPatria, the open source framework of the project.

This thesis takes a first step in formulating, for a specific domain and a number

of tasks, the requirements to support end-user access to semantically-rich and heterogeneous linked data. Different aspects of the search problem are explored in three case studies: (i) artwork annotation to study how users can effectively find terms in multiple vocabularies, (ii) faceted browsing on multiple collections of annotated artworks to study the formulation of structured queries, and (iii) semantic artwork search to study how the different relations in linked vocabularies can be used to find objects semantically related to a query.

Annotation In professional annotation the task of the user is to find vocabulary terms to describe an artwork. In existing collection management systems the annotation fields each provide access to the terms of a single vocabulary. The scope of the internal vocabulary used for this purpose are not always sufficient. In particular, when describing the content depicted on artworks, the cataloguers need to spend valuable time on extending the vocabularies with missing terms. External sources on the Web can be used to extend the scope of annotation terms, but this requires end-user support to find terms in multiple vocabularies. How can text-based search be used to find terms in multiple vocabularies with different schemata and characteristics?

In a study with professional cataloguers at the Print room of the Rijksmuseum Amsterdam, we investigate how multiple vocabularies can be used in an annotation tool. The initial requirements on the data, algorithms and interface design are formulated based on an analysis of their current practices. In a process of iterative prototyping, the requirements are refined and different solutions are explored. The solutions in the final prototype are qualitatively evaluated with feedback from the cataloguers. We found that end-users can be effectively supported with existing technologies, such as client side interface components for autocompletion, and server side algorithms for term search and result organisation and presentation. However, the user interface and algorithms need to be carefully configured for different annotation fields and vocabularies. We identify the required parameterization of the algorithms and user interface, and demonstrate successful configuration for a specific task and domain.

Faceted browsing Faceted browsing is a popular interface paradigm to support the formulation of structured queries to explore (artwork) collections. How can the faceted interface paradigm be used to support the formulation of structured queries for linked data? Traditional faceted browsers require a homogeneous collection with a single schema, whereas heterogeneous Semantic Web repositories may contain different types of objects each with their own facets. In addition, structured queries on linked data involve indirect constraints that cannot be formulated in traditional faceted browsers.

Based on a use case, we formulate the requirements for faceted browsing on heterogeneous Semantic Web repositories. Solutions for the required search func-

tionality and presentation methods are explored by the implementation of a prototype system. We found that the required functionality for any small to medium sized RDFS repository can be supported with a completely data-driven solution. The semantic relations in the data can improve end-user support by organising the large number of facets. The formulation of indirect constraints is supported in the prototype by allowing users to browse different types of objects in the same interface, and using the constraints on one type as constraints of a semantically related type. However, we also found that to support a specific task the existing properties in the data might not match with the user's conceptualisation of the domain. In this case, appropriate facets oriented to the end-user should be identified and manually configured.

Semantic search In many professional search tasks users want to find artworks that are somehow related to a topic. To satisfy their non-trivial information needs, domain experts often need to formulate multiple queries and manually combine and integrate the various search results into a single coherent set of answers. Our hypothesis is that the artwork annotations with terms from multiple structured and interlinked vocabularies, and the relations among these terms, can help end-users in the search process. To provide such semantic search we need to understand which and how the different types of relations in semantically-rich linked data can support end-users.

Support for semantic search is investigated in two experiments. First, the usefulness of different paths of relations investigated in a user study with a small number of domain experts. A number of path types are identified and their usefulness is qualitatively evaluated. In the second experiment, the implementation of the path types in a semantic search application is investigated with the most frequently used queries from a search log. Based on the findings of these experiments the implications for the design of an interactive semantic search application in the cultural heritage domain are discussed. We observed that the process of searching for artworks contains different phases. Different types of relations in the data are useful in these phases and different types of end-user interaction are required to support the user. For example, the initial search results need to include artworks related by literal and object properties as well as equivalence alignments. Interaction should be provided for query disambiguation by selecting vocabulary terms and query reformulation using different types of hierarchical and associative relations between vocabulary terms. To support these strategies multiple interactive interface components are required using different configurations of a graph search algorithm.

From the case studies we derived several architectural requirements on the search functionality and result presentation methods. To support annotation and semantic search, configurable term and graph search are required, as well as configuration

of the result organisation and presentation methods. Interactive solutions are required to support the user in trying different queries and explore different search strategies. In the presentation of navigation paths and search results appropriate abstractions in the data are required to support the user with the large number of different types of terms and relations.

The functionality for text-based search is implemented with generic algorithms on the RDF data model, which can be configured in a number of dimensions. The algorithms are made available in ClioPatria as parameterized web services. The web services are extended with configurable algorithms for organisation and presentation of the search results. The required user interaction is supported by reusing and extending Web interface components. To support the configuration of these components and the web services used by them, we propose a method that captures the functionality in a model. Accordingly the configuration becomes a mapping task that can be performed by a domain expert, reducing the need for a programmer.

Within the explored solutions four types of semantic relations were used to improve end-user support. First, the thesaurus-specific relations in the original data, as defined in SKOS, provide a useful abstraction for vocabularies on which specific functionality and presentation methods can be defined. Second, the equivalence alignments between terms from different vocabularies allow the inclusion of data from external sources, increasing recall in text-based search. They also enable the removal of duplicate search results. Third, lightweight schema mappings enable integrated access to heterogeneous data, while preserving the richness of the individual collections and vocabularies. They also enable the use of different abstractions of the result presentation. Finally, ontological descriptions of properties enable integrated search functionality over heterogeneous data.

In this thesis we showed that a Web of culture data can be used to improve the support for domain experts with a number of tasks. To apply our results to other domains and user populations, the appropriate abstractions in this domain and the specific user needs have to be identified, and the configurations of the search functionality and presentation methods need to be investigated. Based on the conclusions from this thesis we expect that future comparison of different semantic search systems is best studied for a specific type of functionality and considering a specific stage of the search process.

Samenvatting

Ondersteuning van eindgebruikers in het ontsluiten van heterogene gelinkte data

Organisaties maken op het Web hun data en services beschikbaar voor hergebruik. Door dit openbaar beschikbaar maken kan informatie afkomstig van verschillende bronnen worden gecombineerd. Dit maakt nieuwe manieren mogelijk om de informatie te ontsluiten, die de oorspronkelijke dataleveranciers niet noodzakelijk voorzien hadden. Webapplicaties die verschillende databronnen en services hergebruiken zijn bekend als ‘Mash-ups’, en worden een gangbare oplossing om specifieke gebruikerstaken te ondersteunen. Het ontwerpen van dit soort applicaties vereist echter een ontwikkelaar die moet aangeven hoe de data van verschillende bronnen moeten worden gebruikt en geïntegreerd.

Technologieën voor het Semantische Web beloven het hergebruik van verschillende databronnen makkelijker te maken. De representatietalen voor het Semantische Web bieden een gestandaardiseerde manier om kennis te modelleren en te delen. Bovendien kan met deze talen de betekenis van verschillende aspecten in de data expliciet gemaakt worden, zodat machines automatisch kunnen bepalen hoe de data moeten worden gebruikt en geïntegreerd.

Steeds meer databronnen worden op het Semantische Web gepubliceerd en deze databronnen worden onderling met elkaar verbonden. Het resultaat is een grote graaf met gelinkte data, die een grote verscheidenheid aan objecten beschrijft. In een eerste studie analyseren wij hoe bestaande applicaties gebruikers toegang verschaffen tot deze heterogene gelinkte data. We concluderen uit deze studie dat applicaties voor het Semantische Web een ruime verscheidenheid aan taken ondersteunen en toegang verschaffen tot verschillende soorten databronnen. Het blijft echter onduidelijk hoe goed de gebruikte semantische technologieën eindgebruikers ondersteunen, omdat algemeen geaccepteerde methoden om deze applicaties te evalueren ontbreken. Gegeven een specifiek domein en taak is het dus moeilijk

om te bepalen hoe de semantiek in de data moet worden gebruikt om de beste ondersteuning te bieden aan eindgebruikers.

In dit proefschrift is ons culturele erfgoed gekozen als het applicatiedomein om de toegang tot heterogene en gelinkte data te bestuderen. Het semantisch-rijke cultuurweb, dat is geconstrueerd in het MultimediaN E-Culture project, wordt gebruikt als experimentele data. De praktische resultaten van dit onderzoek zijn beschikbaar als applicaties en web services in Cliopatria, de open source software architectuur van dit project.

Dit proefschrift zet een eerste stap om voor een specifiek domein en aantal taken de eisen te formuleren om eindgebruikers toegang te verlenen tot semantisch-rijke en heterogene gelinkte data. Verschillende aspecten van het zoekprobleem worden geëxploreerd in drie casussen: (i) annotatie van kunstwerken om te bestuderen hoe gebruikers efficiënt naar termen kunnen zoeken in meerdere vocabulaires, (ii) facet gebaseerd navigeren om het formuleren van gestructureerde zoekvragen in meerdere collecties van geannoteerde kunstwerken te bestuderen, en (iii) het semantisch zoeken naar kunstwerken om te bestuderen hoe de relaties in meerdere gelinkte vocabulaires kunnen worden gebruikt om objecten te vinden die semantisch gerelateerd zijn aan een query.

Annotatie In professionele annotatie is de taak van de gebruiker om vocabulaire termen te vinden om een kunstwerk te beschrijven. In de huidige systemen voor collectiemanagement geeft elk annotatieveld toegang tot de termen van één thesaurus. De dekking van de vocabulaires, die intern in musea worden gebruikt, zijn niet altijd toereikend. In het bijzonder voor de beschrijving van het onderwerp afgebeeld op een kunstwerk moeten de catalogiseerders daarom hun kostbare tijd ook besteden aan het toevoegen van nieuwe vocabulaire termen. Externe bronnen op het Web kunnen de dekking van beschikbare annotatietermen verhogen. Dit vereist ondersteuning van gebruikers met het zoeken in meerdere bronnen die verschillende karakteristieken en dataschema's kunnen hebben. Hoe kan tekstgebaseerd zoeken gebruikt worden om termen te vinden in meerdere vocabulaires?

In een studie met professionele catalogiseerders van het prentenkabinet van het Rijksmuseum Amsterdam onderzoeken we hoe meerdere vocabulaires in een annotatie tool kunnen worden geïntegreerd. De initiële eisen op de data, algoritmes en interface worden geformuleerd op basis van een analyse van het huidige annotatie proces. Door in meerdere iteraties verschillende prototypes te ontwikkelen worden de eisen bijgesteld en verschillende oplossingen uitgetoetst. De oplossingen in het laatste prototype worden kwalitatief geëvolueerd met de catalogiseerders. We stellen vast dat eindgebruikers effectief kunnen worden ondersteund met bestaande technologieën, zoals interface componenten voor 'autocompletion' en algoritmes voor het zoeken naar termen en het organiseren en presenteren van de geselecteerde termen. De gebruikersinterface en de algoritmes moeten echter wel zorgvuldig worden geconfigureerd voor verschillende annotatievelden en vocabulaires. We

identificeren de vereiste parameters van de algoritmes en de gebruikers-interface en demonstreren hoe deze kunnen worden geconfigureerd voor een specifieke taak en domein.

Facet gebaseerd navigeren Facet gebaseerd navigeren is een populaire strategie voor het formuleren van gestructureerde zoekvragen om (kunst) collecties te exploreren. Hoe kan het facet gebaseerde navigeren worden gebruikt om het formuleren van gestructureerde zoekvragen voor gelinkte data te ondersteunen? Traditionele zoekapplicaties die facet gebaseerd navigeren ondersteunen vereisen een homogene collectie met een vast dataschema. Heterogene databronnen op het Semantische Web kunnen verschillende soorten objecten bevatten, elk met hun eigen facetten. Bovendien kunnen gestructureerde zoekvragen voor gelinkte data indirecte restricties bevatten, die niet door traditioneel facet gebaseerd navigeren worden ondersteund.

Op basis van een ‘use case’ formuleren we de eisen voor facet gebaseerd navigeren op heterogene databronnen op het Semantische Web. Oplossingen voor de vereiste zoekfunctionaliteit en presentatie methodes worden geëxploreerd door het implementeren van een prototype. We stellen vast dat de vereiste functionaliteit voor kleine en middelgrote RDFS databronnen kan worden ondersteund met een volledige datagestuurde oplossing. De semantische relaties in de data kunnen worden gebruikt om de facetten te organiseren en daarmee de ondersteuning voor de eindgebruikers te verbeteren. Het formuleren van indirecte restricties wordt in het prototype ondersteund door de gebruiker meerdere soorten objecten te laten navigeren in dezelfde interface waarbij de restricties op objecten van één type kunnen worden gebruikt voor objecten van een semantisch gerelateerd type object. We moeten ook concluderen dat de relaties in de data voor specifieke taken niet altijd overeenkomen met het perspectief van de gebruiker. In dit geval moet worden bepaald welke facetten geschikt zijn voor eindgebruikers en deze moeten handmatig worden geconfigureerd.

Semantisch zoeken In veel professionele zoektaken willen gebruikers kunstwerken vinden die op verschillende manieren gerelateerd zijn aan een onderwerp. Om hun niet triviale informatiebehoefes te bevredigen moeten domeinexperts vaak meerdere zoekvragen formuleren en handmatig de verschillende resultaten integreren in een coherente verzameling. Onze hypothese is dat eindgebruikers ondersteund kunnen worden in het zoekproces door op de juiste manier gebruik te maken van de annotaties van kunstwerken met termen uit gestructureerde en gelinkte vocabulaires en de relaties tussen deze termen. Om dit semantische zoeken mogelijk te maken moeten we eerst beter begrijpen welke en hoe verschillende soorten relaties in de semantisch-rijke databronnen gebruikt kunnen worden.

De ondersteuning van semantisch zoeken is bestudeerd in twee experimenten. Ten eerste is het nut van verschillende paden van relaties onderzocht in een ge-

bruikersstudie met een klein aantal domeinexperts. Een aantal typen paden is geïdentificeerd en hun nuttigheid voor de experts is kwalitatief geëvalueerd. In het tweede experiment is het gebruik van de verschillende typen paden in een interactieve zoekapplicatie onderzocht. Op basis van deze experimenten worden de implicaties voor het ontwerp van een interactieve semantische zoek applicatie voor cultureel erfgoed besproken. We observeerden dat het proces van zoeken naar kunstwerken uit verschillende fases bestaat. Verschillende soorten relaties in de data zijn nuttig in die fases en verschillende soorten interactie zijn nodig om de eindgebruikers te ondersteunen in de fases. De initiële zoekresultaten moeten, bijvoorbeeld, kunstwerken bevatten die gerelateerd zijn aan de zoekvraag op basis van tekstuele eigenschappen en met annotaties van gecontroleerde termen. Ook moeten de equivalentie relaties tussen de vocabulaire termen meegenomen worden. Vervolgens moet er interactie mogelijk zijn om de zoekvraag te disambigueren door vocabulaire termen te selecteren. Ook moet het mogelijk zijn om de zoekvraag te herformuleren door gebruik te maken van verschillende hiërarchische en associatieve relaties tussen termen. Om deze zoekstrategieën te ondersteunen zijn verschillende interface componenten nodig die gebruik maken van graaf zoek algoritmes die op verschillende manieren zijn geconfigureerd.

De casussen brengen verscheidene eisen naar boven op de software architectuur die nodig is voor de zoekfunctionaliteit en de presentatiemethoden. Om annotatie en semantisch zoeken te ondersteunen zijn configureerbare zoek algoritmes noodzakelijk. Bovendien moet de organisatie en de presentatie van de resultaten configureerbaar zijn. Er zijn verschillende interactieve oplossingen nodig om de gebruiker te ondersteunen met het uitproberen van verschillende zoekvragen en het exploreren van verschillende zoekstrategieën. Voor de presentatie van de zoekresultaten en de navigatiepaden zijn er abstracties in data nodig waarmee de grote verscheidenheid aan termen en relaties behapbaar gemaakt kan worden.

De functionaliteit voor tekst gebaseerd zoeken is geïmplementeerd met generieke algoritmes voor RDF data en kan geconfigureerd worden op een aantal dimensies. De algoritmes zijn beschikbaar in ClioPatria als geparameteriseerde web services. Deze services zijn uitgebreid met algoritmes om de resultaten te organiseren en presenteren. De vereiste interactie wordt ondersteund door bestaande Web interface componenten te hergebruiken en uit te breiden. Om deze componenten en de web services waar ze gebruik van maken te configureren hebben we een methode voorgesteld waarin de functionaliteit in een model wordt omschreven. Hierdoor wordt de configuratie een taak waarin de domeinspecifieke data gelinkt worden aan dit model. Dit kan uitgevoerd worden door een domeinexpert waardoor er geen ontwikkelaar meer nodig is.

In de geëxploreerde oplossingen worden vier soorten semantische relaties gebruikt om de gebruikersondersteuning te verbeteren. Ten eerste, de thesaurus specifieke relaties in de oorspronkelijk data, zoals gedefinieerd in SKOS, bieden nuttige

abstracties over vocabulaires waarop specifieke functionaliteit en presentatiemethoden kunnen worden gebaseerd. Ten tweede de equivalentierelaties tussen termen van verschillende vocabulaires maken het mogelijk om informatie van externe bronnen te integreren. Deze relaties maken het ook mogelijk om dubbele resultaten te verwijderen. Ten derde, de simpele relaties om dataschema's te linken maken het mogelijk om geïntegreerde toegang te verschaffen tot heterogene data, terwijl de rijkheid van de individuele collecties en vocabulaires behouden blijft. Deze relaties maken het ook mogelijk om de resultaten op verschillende abstractie niveaus te presenteren. Ten vierde, de ontologische beschrijvingen van eigenschappen verbeteren de geïntegreerde zoekfunctionaliteit in heterogene data.

In dit proefschrift hebben we laten zien dat een Web van culturele data gebruikt kan worden om domeinexperts met een aantal taken te ondersteunen. Om de resultaten toepasbaar te maken op andere domeinen en gebruikersgroepen moeten de relevante abstracties voor dat domein en de specifieke gebruikerseigenschappen geïdentificeerd worden. Het vinden van de juiste configuraties voor de zoekfunctionaliteit en de presentatiemethodes vereist verder onderzoek. We verwachten dat in de toekomst de ondersteuning van semantische zoeksystemen het best kunnen worden geëvalueerd voor specifieke functionaliteit en een specifieke fase in het zoekproces.

SIKS Dissertatiereeks

1998

- 1998-1 Johan van den Akker (CWI)
DEGAS - An Active, Temporal
Database of Autonomous Objects
- 1998-2 Floris Wiesman (UM)
Information Retrieval by Graphically
Browsing Meta-Information
- 1998-3 Ans Steuten (TUD)
A Contribution to the Linguistic
Analysis of Business Conversations
within the Language/Action
Perspective
- 1998-4 Dennis Breuker (UM)
Memory versus Search in Games
- 1998-5 E.W.Oskamp (RUL)
Computerondersteuning bij
Straftoemeting

1999

- 1999-1 Mark Sloof (VU)
Physiology of Quality Change
Modelling; Automated modelling of
Quality Change of Agricultural
Products
- 1999-2 Rob Potharst (EUR)
Classification using decision trees and
neural nets
- 1999-3 Don Beal (UM)
The Nature of Minimax Search
- 1999-4 Jacques Penders (UM)
The practical Art of Moving Physical
Objects
- 1999-5 Aldo de Moor (KUB)
Empowering Communities: A Method
for the Legitimate User-Driven
Specification of Network Information
Systems
- 1999-6 Niek J.E. Wijngaards (VU)
Re-design of compositional systems
- 1999-7 David Spelt (UT)
Verification support for object database
design
- 1999-8 Jacques H.J. Lenting (UM)

Informed Gambling; Conception and
Analysis of a Multi-Agent Mechanism
for Discrete Reallocation

2000

- 2000-1 Frank Niessink (VU)
Perspectives on Improving Software
Maintenance
- 2000-2 Koen Holtman (TUE)
Prototyping of CMS Storage
Management
- 2000-3 Carolien M.T. Metselaar (UvA)
Sociaal-organisatorische gevolgen van
kennistechnologie; een
procesbenadering en actorperspectief
- 2000-4 Geert de Haan (VU)
ETAG, A Formal Model of Competence
Knowledge for User Interface Design
- 2000-5 Ruud van der Pol (UM)
Knowledge-based Query Formulation in
Information Retrieval
- 2000-6 Rogier van Eijk (UU)
Programming Languages for Agent
Communication
- 2000-7 Niels Peek (UU)
Decision-theoretic Planning of Clinical
Patient Management
- 2000-8 Veerle Coup (EUR)
Sensitivity Analysis of
Decision-Theoretic Networks
- 2000-9 Florian Waas (CWI)
Principles of Probabilistic Query
Optimization
- 2000-10 Niels Nes (CWI)
Image Database Management System
Design Considerations, Algorithms and
Architecture
- 2000-11 Jonas Karlsson (CWI)
Scalable Distributed Data Structures
for Database Management

2001

- 2001-1 Silja Renooij (UU)

- Qualitative Approaches to Quantifying Probabilistic Networks
- 2001-2 Koen Hindriks (UU)
Agent Programming Languages: Programming with Mental Models
- 2001-3 Maarten van Someren (UvA)
Learning as problem solving
- 2001-4 Evgueni Smirnov (UM)
Conjunctive and Disjunctive Version Spaces with Instance-Based Boundary Sets
- 2001-5 Jacco van Ossenbruggen (VU)
Processing Structured Hypermedia: A Matter of Style
- 2001-6 Martijn van Welie (VU)
Task-based User Interface Design
- 2001-7 Bastiaan Schonhage (VU)
Diva: Architectural Perspectives on Information Visualization
- 2001-8 Pascal van Eck (VU)
A Compositional Semantic Structure for Multi-Agent Systems Dynamics
- 2001-9 Pieter Jan 't Hoen (RUL)
Towards Distributed Development of Large Object-Oriented Models, Views of Packages as Classes
- 2001-10 Maarten Sierhuis (UvA)
Modeling and Simulating Work Practice BRAHMS: a multiagent modeling and simulation language for work practice analysis and design
- 2001-11 Tom M. van Engers (VUA)
Knowledge Management: The Role of Mental Models in Business Systems Design
- 2002**
- 2002-01 Nico Lassing (VU)
Architecture-Level Modifiability Analysis
- 2002-02 Roelof van Zwol (UT)
Modelling and searching web-based document collections
- 2002-03 Henk Ernst Blok (UT)
Database Optimization Aspects for Information Retrieval
- 2002-04 Juan Roberto Castelo Valdueza (UU)
The Discrete Acyclic Digraph Markov Model in Data Mining
- 2002-05 Radu Serban (VU)
The Private Cyberspace Modeling Electronic Environments inhabited by Privacy-concerned Agents
- 2002-06 Laurens Mommers (UL)
Applied legal epistemology; Building a knowledge-based ontology of the legal domain
- 2002-07 Peter Boncz (CWI)
Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications
- 2002-08 Jaap Gordijn (VU)
Value Based Requirements Engineering: Exploring Innovative E-Commerce Ideas
- 2002-09 Willem-Jan van den Heuvel (KUB)
Integrating Modern Business Applications with Objectified Legacy Systems
- 2002-10 Brian Sheppard (UM)
Towards Perfect Play of Scrabble
- 2002-11 Wouter C.A. Wijngaards (VU)
Agent Based Modelling of Dynamics: Biological and Organisational Applications
- 2002-12 Albrecht Schmidt (Uva)
Processing XML in Database Systems
- 2002-13 Hongjing Wu (TUE)
A Reference Architecture for Adaptive Hypermedia Applications
- 2002-14 Wieke de Vries (UU)
Agent Interaction: Abstract Approaches to Modelling, Programming and Verifying Multi-Agent Systems
- 2002-15 Rik Eshuis (UT)
Semantics and Verification of UML Activity Diagrams for Workflow Modelling
- 2002-16 Pieter van Langen (VU)
The Anatomy of Design: Foundations, Models and Applications
- 2002-17 Stefan Manegold (UvA)
Understanding, Modeling, and Improving Main-Memory Database Performance
- 2003**
- 2003-01 Heiner Stuckenschmidt (VU)
Ontology-Based Information Sharing in Weakly Structured Environments
- 2003-02 Jan Broersen (VU)
Modal Action Logics for Reasoning About Reactive Systems
- 2003-03 Martijn Schuemie (TUD)
Human-Computer Interaction and Presence in Virtual Reality Exposure Therapy
- 2003-04 Milan Petkovic (UT)
Content-Based Video Retrieval Supported by Database Technology
- 2003-05 Jos Lehmann (UvA)
Causation in Artificial Intelligence and Law - A modelling approach
- 2003-06 Boris van Schooten (UT)
Development and specification of virtual environments
- 2003-07 Machiel Jansen (UvA)
Formal Explorations of Knowledge Intensive Tasks
- 2003-08 Yongping Ran (UM)
Repair Based Scheduling
- 2003-09 Rens Kortmann (UM)
The resolution of visually guided behaviour

- 2003-10 Andreas Lincke (UvT)
Electronic Business Negotiation: Some experimental studies on the interaction between medium, innovation context and culture
- 2003-11 Simon Keizer (UT)
Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks
- 2003-12 Roeland Ordelman (UT)
Dutch speech recognition in multimedia information retrieval
- 2003-13 Jeroen Donkers (UM)
Nosce Hostem - Searching with Opponent Models
- 2003-14 Stijn Hoppenbrouwers (KUN)
Freezing Language: Conceptualisation Processes across ICT-Supported Organisations
- 2003-15 Mathijs de Weerd (TUD)
Plan Merging in Multi-Agent Systems
- 2003-16 Menzo Windhouwer (CWI)
Feature Grammar Systems - Incremental Maintenance of Indexes to Digital Media Warehouses
- 2003-17 David Jansen (UT)
Extensions of Statecharts with Probability, Time, and Stochastic Timing
- 2003-18 Levente Kocsis (UM)
Learning Search Decisions
- 2004**
- 2004-01 Virginia Dignum (UU)
A Model for Organizational Interaction: Based on Agents, Founded in Logic
- 2004-02 Lai Xu (UvT)
Monitoring Multi-party Contracts for E-business
- 2004-03 Perry Groot (VU)
A Theoretical and Empirical Analysis of Approximation in Symbolic Problem Solving
- 2004-04 Chris van Aart (UvA)
Organizational Principles for Multi-Agent Architectures
- 2004-05 Viara Popova (EUR)
Knowledge discovery and monotonicity
- 2004-06 Bart-Jan Hommes (TUD)
The Evaluation of Business Process Modeling Techniques
- 2004-07 Elise Boltjes (UM)
Voorbeeldig onderwijs; voorbeeldgestuurd onderwijs, een opstap naar abstract denken, vooral voor meisjes
- 2004-08 Joop Verbeek (UM)
Politie en de Nieuwe Internationale Informatiemarkt, Grensregionale politieële gegevensuitwisseling en digitale expertise
- 2004-09 Martin Caminada (VU)
For the Sake of the Argument; explorations into argument-based reasoning
- 2004-10 Suzanne Kabel (UvA)
Knowledge-rich indexing of learning-objects
- 2004-11 Michel Klein (VU)
Change Management for Distributed Ontologies
- 2004-12 The Duy Bui (UT)
Creating emotions and facial expressions for embodied agents
- 2004-13 Wojciech Jamroga (UT)
Using Multiple Models of Reality: On Agents who Know how to Play
- 2004-14 Paul Harrenstein (UU)
Logic in Conflict. Logical Explorations in Strategic Equilibrium
- 2004-15 Arno Knobbe (UU)
Multi-Relational Data Mining
- 2004-16 Federico Divina (VU)
Hybrid Genetic Relational Search for Inductive Learning
- 2004-17 Mark Winands (UM)
Informed Search in Complex Games
- 2004-18 Vania Bessa Machado (UvA)
Supporting the Construction of Qualitative Knowledge Models
- 2004-19 Thijs Westerveld (UT)
Using generative probabilistic models for multimedia retrieval
- 2004-20 Madelon Evers (Nyenrode)
Learning from Design: facilitating multidisciplinary design teams
- 2005**
- 2005-01 Floor Verdenius (UvA)
Methodological Aspects of Designing Induction-Based Applications
- 2005-02 Erik van der Werf (UM)
AI techniques for the game of Go
- 2005-03 Franc Grootjen (RUN)
A Pragmatic Approach to the Conceptualisation of Language
- 2005-04 Nirvana Meratnia (UT)
Towards Database Support for Moving Object data
- 2005-05 Gabriel Infante-Lopez (UvA)
Two-Level Probabilistic Grammars for Natural Language Parsing
- 2005-06 Pieter Spronck (UM)
Adaptive Game AI
- 2005-07 Flavius Frasinca (TUE)
Hypermedia Presentation Generation for Semantic Web Information Systems
- 2005-08 Richard Vdovjak (TUE)
A Model-driven Approach for Building Distributed Ontology-based Web Applications
- 2005-09 Jeen Broekstra (VU)
Storage, Querying and Inferencing for Semantic Web Languages

- 2005-10 Anders Bouwer (UvA)
Explaining Behaviour: Using Qualitative Simulation in Interactive Learning Environments
- 2005-11 Elth Ogston (VU)
Agent Based Matchmaking and Clustering - A Decentralized Approach to Search
- 2005-12 Csaba Boer (EUR)
Distributed Simulation in Industry
- 2005-13 Fred Hamburg (UL)
Een Computermodel voor het Ondersteunen van Euthanasiebeslissingen
- 2005-14 Borys Omelayenko (VU)
Web-Service configuration on the Semantic Web; Exploring how semantics meets pragmatics
- 2005-15 Tibor Bosse (VU)
Analysis of the Dynamics of Cognitive Processes
- 2005-16 Joris Graaumanns (UU)
Usability of XML Query Languages
- 2005-17 Boris Shishkov (TUD)
Software Specification Based on Re-usable Business Components
- 2005-18 Danielle Sent (UU)
Test-selection strategies for probabilistic networks
- 2005-19 Michel van Dartel (UM)
Situating Representation
- 2005-20 Cristina Coteanu (UL)
Cyber Consumer Law, State of the Art and Perspectives
- 2005-21 Wijnand Derks (UT)
Improving Concurrency and Recovery in Database Systems by Exploiting Application Semantics
- 2006**
- 2006-01 Samuil Angelov (TUE)
Foundations of B2B Electronic Contracting
- 2006-02 Cristina Chisalita (VU)
Contextual issues in the design and use of information technology in organizations
- 2006-03 Noor Christoph (UvA)
The role of metacognitive skills in learning to solve problems
- 2006-04 Marta Sabou (VU)
Building Web Service Ontologies
- 2006-05 Cees Pierik (UU)
Validation Techniques for Object-Oriented Proof Outlines
- 2006-06 Ziv Baida (VU)
Software-aided Service Bundling - Intelligent Methods & Tools for Graphical Service Modeling
- 2006-07 Marko Smiljanic (UT)
XML schema matching – balancing efficiency and effectiveness by means of clustering
- 2006-08 Eelco Herder (UT)
Forward, Back and Home Again - Analyzing User Behavior on the Web
- 2006-09 Mohamed Wahdan (UM)
Automatic Formulation of the Auditor's Opinion
- 2006-10 Ronny Siebes (VU)
Semantic Routing in Peer-to-Peer Systems
- 2006-11 Joeri van Ruth (UT)
Flattening Queries over Nested Data Types
- 2006-12 Bert Bongers (VU)
Interactivation - Towards an e-ecology of people, our technological environment, and the arts
- 2006-13 Henk-Jan Lebbink (UU)
Dialogue and Decision Games for Information Exchanging Agents
- 2006-14 Johan Hoorn (VU)
Software Requirements: Update, Upgrade, Redesign - towards a Theory of Requirements Change
- 2006-15 Rainer Malik (UU)
CONAN: Text Mining in the Biomedical Domain
- 2006-16 Carsten Riggelsen (UU)
Approximation Methods for Efficient Learning of Bayesian Networks
- 2006-17 Stacey Nagata (UU)
User Assistance for Multitasking with Interruptions on a Mobile Device
- 2006-18 Valentin Zhizhkun (UvA)
Graph transformation for Natural Language Processing
- 2006-19 Birna van Riemsdijk (UU)
Cognitive Agent Programming: A Semantic Approach
- 2006-20 Marina Velikova (UvT)
Monotone models for prediction in data mining
- 2006-21 Bas van Gils (RUN)
Aptness on the Web
- 2006-22 Paul de Vrieze (RUN)
Fundamentals of Adaptive Personalisation
- 2006-23 Ion Juvina (UU)
Development of Cognitive Model for Navigating on the Web
- 2006-24 Laura Hollink (VU)
Semantic Annotation for Retrieval of Visual Resources
- 2006-25 Madalina Drugan (UU)
Conditional log-likelihood MDL and Evolutionary MCMC
- 2006-26 Vojkan Mihajlović (UT)
Score Region Algebra: A Flexible Framework for Structured Information Retrieval
- 2006-27 Stefano Bocconi (CWI)
Vox Populi: generating video documentaries from semantically

- annotated media repositories
- 2006-28 Borkur Sigurbjornsson (UvA)
Focused Information Access using XML
Element Retrieval
- 2007**
- 2007-01 Kees Leune (UvT)
Access Control and Service-Oriented
Architectures
- 2007-02 Wouter Teepe (RUG)
Reconciling Information Exchange and
Confidentiality: A Formal Approach
- 2007-03 Peter Mika (VU)
Social Networks and the Semantic Web
- 2007-04 Jurriaan van Diggelen (UU)
Achieving Semantic Interoperability in
Multi-agent Systems: a dialogue-based
approach
- 2007-05 Bart Schermer (UL)
Software Agents, Surveillance, and the
Right to Privacy: a Legislative
Framework for Agent-enabled
Surveillance
- 2007-06 Gilad Mishne (UvA)
Applied Text Analytics for Blogs
- 2007-07 Natasa Jovanovic' (UT)
To Whom It May Concern - Addressee
Identification in Face-to-Face Meetings
- 2007-08 Mark Hoogendoorn (VU)
Modeling of Change in Multi-Agent
Organizations
- 2007-09 David Mobach (VU)
Agent-Based Mediated Service
Negotiation
- 2007-10 Huib Aldewereld (UU)
Autonomy vs. Conformity: an
Institutional Perspective on Norms and
Protocols
- 2007-11 Natalia Stash (TUE)
Incorporating Cognitive/Learning
Styles in a General-Purpose Adaptive
Hypermedia System
- 2007-12 Marcel van Gerven (RUN)
Bayesian Networks for Clinical Decision
Support: A Rational Approach to
Dynamic Decision-Making under
Uncertainty
- 2007-13 Rutger Rienks (UT)
Meetings in Smart Environments;
Implications of Progressing Technology
- 2007-14 Niek Bergboer (UM)
Context-Based Image Analysis
- 2007-15 Joyca Lacroix (UM)
NIM: a Situated Computational
Memory Model
- 2007-16 Davide Grossi (UU)
Designing Invisible Handcuffs. Formal
investigations in Institutions and
Organizations for Multi-agent Systems
- 2007-17 Theodore Charitos (UU)
Reasoning with Dynamic Networks in
Practice
- 2007-18 Bart Orriens (UvT)
On the development and management
of adaptive business collaborations
- 2007-19 David Levy (UM)
Intimate relationships with artificial
partners
- 2007-20 Slinger Jansen (UU)
Customer Configuration Updating in a
Software Supply Network
- 2007-21 Karianne Vermaas (UU)
Fast diffusion and broadening use: A
research on residential adoption and
usage of broadband internet in the
Netherlands between 2001 and 2005
- 2007-22 Zlatko Zlatev (UT)
Goal-oriented design of value and
process models from patterns
- 2007-23 Peter Barna (TUE)
Specification of Application Logic in
Web Information Systems
- 2007-24 Georgina Ramrez Camps (CWI)
Structural Features in XML Retrieval
- 2007-25 Joost Schalken (VU)
Empirical Investigations in Software
Process Improvement
- 2008**
- 2008-01 Katalin Boer-Sorbn (EUR)
Agent-Based Simulation of Financial
Markets: A modular, continuous-time
approach
- 2008-02 Alexei Sharpanskykh (VU)
On Computer-Aided Methods for
Modeling and Analysis of Organizations
- 2008-03 Vera Hollink (UvA)
Optimizing hierarchical menus: a
usage-based approach
- 2008-04 Ander de Keijzer (UT)
Management of Uncertain Data -
towards unattended integration
- 2008-05 Bela Mutschler (UT)
Modeling and simulating causal
dependencies on process-aware
information systems from a cost
perspective
- 2008-06 Arjen Hommersom (RUN)
On the Application of Formal Methods
to Clinical Guidelines, an Artificial
Intelligence Perspective
- 2008-07 Peter van Rosmalen (OU)
Supporting the tutor in the design and
support of adaptive e-learning
- 2008-08 Janneke Bolt (UU)
Bayesian Networks: Aspects of
Approximate Inference
- 2008-09 Christof van Nimwegen (UU)
The paradox of the guided user:
assistance can be counter-effective
- 2008-10 Wauter Bosma (UT)
Discourse oriented summarization
- 2008-11 Vera Kartseva (VU)
Designing Controls for Network

- Organizations: A Value-Based Approach
- 2008-12 Jozsef Farkas (RUN)
A Semiotically Oriented Cognitive Model of Knowledge Representation
- 2008-13 Caterina Carraciolo (UvA)
Topic Driven Access to Scientific Handbooks
- 2008-14 Arthur van Bunningen (UT)
Context-Aware Querying; Better Answers with Less Effort
- 2008-15 Martijn van Otterlo (UT)
The Logic of Adaptive Behavior: Knowledge Representation and Algorithms for the Markov Decision Process Framework in First-Order Domains
- 2008-16 Henriette van Vugt (VU)
Embodied agents from a user's perspective
- 2008-17 Martin Op 't Land (TUD)
Applying Architecture and Ontology to the Splitting and Allying of Enterprises
- 2008-18 Guido de Croon (UM)
Adaptive Active Vision
- 2008-19 Henning Rode (UT)
From Document to Entity Retrieval: Improving Precision and Performance of Focused Text Search
- 2008-20 Rex Arendsen (UvA)
Geen bericht, goed bericht. Een onderzoek naar de effecten van de introductie van elektronisch berichtenverkeer met de overheid op de administratieve lasten van bedrijven
- 2008-21 Krisztian Balog (UvA)
People Search in the Enterprise
- 2008-22 Henk Koning (UU)
Communication of IT-Architecture
- 2008-23 Stefan Visscher (UU)
Bayesian network models for the management of ventilator-associated pneumonia
- 2008-24 Zharko Aleksovski (VU)
Using background knowledge in ontology matching
- 2008-25 Geert Jonker (UU)
Efficient and Equitable Exchange in Air Traffic Management Plan Repair using Spender-signed Currency
- 2008-26 Marijn Huijbregts (UT)
Segmentation, Diarization and Speech Transcription: Surprise Data Unraveled
- 2008-27 Hubert Vogten (OU)
Design and Implementation Strategies for IMS Learning Design
- 2008-28 Ildiko Flesch (RUN)
On the Use of Independence Relations in Bayesian Networks
- 2008-29 Dennis Reidsma (UT)
Annotations and Subjective Machines - Of Annotators, Embodied Agents, Users, and Other Humans
- 2008-30 Wouter van Atteveldt (VU)
Semantic Network Analysis: Techniques for Extracting, Representing and Querying Media Content
- 2008-31 Loes Braun (UM)
Pro-Active Medical Information Retrieval
- 2008-32 Trung H. Bui (UT)
Toward Affective Dialogue Management using Partially Observable Markov Decision Processes
- 2008-33 Frank Terpstra (UvA)
Scientific Workflow Design; theoretical and practical issues
- 2008-34 Jeroen de Knijf (UU)
Studies in Frequent Tree Mining
- 2008-35 Ben Torben Nielsen (UvT)
Dendritic morphologies: function shapes structure
- 2009**
- 2009-01 Rasa Jurgelenaite (RUN)
Symmetric Causal Independence Models
- 2009-02 Willem Robert van Hage (VU)
Evaluating Ontology-Alignment Techniques
- 2009-03 Hans Stol (UvT)
A Framework for Evidence-based Policy Making Using IT
- 2009-04 Josephine Nabukenya (RUN)
Improving the Quality of Organisational Policy Making using Collaboration Engineering
- 2009-05 Sietse Overbeek (RUN)
Bridging Supply and Demand for Knowledge Intensive Tasks - Based on Knowledge, Cognition, and Quality
- 2009-06 Muhammad Subianto (UU)
Understanding Classification
- 2009-07 Ronald Poppe (UT)
Discriminative Vision-Based Recovery and Recognition of Human Motion
- 2009-08 Volker Nannen (VU)
Evolutionary Agent-Based Policy Analysis in Dynamic Environments
- 2009-09 Benjamin Kanagwa (RUN)
Design, Discovery and Construction of Service-oriented Systems
- 2009-10 Jan Wielemaker (UVA)
Logic programming for knowledge-intensive interactive applications
- 2009-11 Alexander Boer (UVA)
Legal Theory, Sources of Law & the Semantic Web
- 2009-12 Peter Massuthe (TUE,
Humboldt-Universitaet zu Berlin)
Operating Guidelines for Services
- 2009-13 Steven de Jong (UM)
Fairness in Multi-Agent Systems
- 2009-14 Maksym Korotkiy (VU)

- From ontology-enabled services to service-enabled ontologiesmaking ontologies work in e-science with ONTO-SOA
- 2009-15 Rinke Hoekstra (UVA)
Ontology Representation - Design Patterns and Ontologies that Make Sense
- 2009-16 Fritz Reul (UvT)
New Architectures in Computer Chess
- 2009-17 Laurens van der Maaten (UvT)
Feature Extraction from Visual Data
- 2009-18 Fabian Groffen (CWI)
Armada, An Evolving Database System
- 2009-19 Valentin Robu (CWI)
Modeling Preferences, Strategic Reasoning and Collaboration in Agent-Mediated Electronic Markets
- 2009-20 Bob van der Vecht (UU)
Adjustable Autonomy: Controlling Influences on Decision Making
- 2009-21 Stijn Vanderlooy (UM)
Ranking and Reliable Classification
- 2009-22 Pavel Serdyukov (UT)
Search For Expertise: Going beyond direct evidence
- 2009-23 Peter Hofgesang (VU)
Modelling Web Usage in a Changing Environment
- 2009-24 Annerieke Heuvelink (VUA)
Cognitive Models for Training Simulations
- 2009-25 Alex van Ballegooij (CWI)
RAM: Array Database Management through Relational Mapping
- 2009-26 Fernando Koch (UU)
An Agent-Based Model for the Development of Intelligent Mobile Services
- 2009-27 Christian Glahn (OU)
Contextual Support of social Engagement and Reflection on the Web
- 2009-28 Sander Evers (UT)
Sensor Data Management with Probabilistic Models
- 2009-29 Stanislav Pokraev (UT)
Model-Driven Semantic Integration of Service-Oriented Applications
- 2009-30 Marcin Zukowski (CWI)
Balancing vectorized query execution with bandwidth-optimized storage
- 2009-31 Sofiya Katrenko (UVA)
A Closer Look at Learning Relations from Text
- 2009-32 Rik Farenhorst and Remco de Boer (VU)
Architectural Knowledge Management: Supporting Architects and Auditors
- 2009-33 Khiet Truong (UT)
How Does Real Affect Affect Recognition In Speech?
- 2009-34 Inge van de Weerd (UU)
Advancing in Software Product Management: An Incremental Method Engineering Approach
- 2009-35 Wouter Koelwijn (UL)
Privacy en Politiegegevens; Over geautomatiseerde normatieve informatie-uitwisseling
- 2009-36 Marco Kalz (OUN)
Placement Support for Learners in Learning Networks
- 2009-37 Hendrik Drachler (OUN)
Navigation Support for Learners in Informal Learning Networks
- 2009-38 Riina Vuorikari (OU)
Tags and self-organisation: a metadata ecology for learning resources in a multilingual context
- 2009-39 Christian Stahl (TUE, Humboldt-Universitaet zu Berlin)
Service Substitution – A Behavioral Approach Based on Petri Nets
- 2009-40 Stephan Raaijmakers (UvT)
Multinomial Language Learning: Investigations into the Geometry of Language
- 2009-41 Igor Berezhnyy (UvT)
Digital Analysis of Paintings
- 2009-42 Toine Bogers (Recommender Systems for Social Bookmarking)
- 2009-43 Virginia Nunes Leal Franqueira (UT)
Finding Multi-step Attacks in Computer Networks using Heuristic Search and Mobile Ambients
- 2009-44 Roberto Santana Tapia (UT)
Assessing Business-IT Alignment in Networked Organizations
- 2009-45 Jilles Vreeken (UU)
Making Pattern Mining Useful
- 2009-46 Loredana Afanasiev (UvA)
Querying XML: Benchmarks and Recursion
- 2010**
- 2010-01 Matthijs van Leeuwen (UU)
Patterns that Matter
- 2010-02 Ingo Wassink (UT)
Work flows in Life Science
- 2010-03 Joost Geurts (CWI)
A Document Engineering Model and Processing Framework for Multimedia documents
- 2010-04 Olga Kulyk (UT)
Do You Know What I Know?
Situational Awareness of Co-located Teams in Multidisplay Environments
- 2010-05 Claudia Hauff (UT)
Predicting the Effectiveness of Queries and Retrieval Systems
- 2010-06 Sander Bakkes (UvT)
Rapid Adaptation of Video Game AI
- 2010-07 Wim Fikkert (UT)
A Gesture interaction at a Distance
- 2010-08 Krzysztof Siewicz (UL)

- Towards an Improved Regulatory Framework of Free Software. Protecting user freedoms in a world of software communities and eGovernments
- 2010-09 Hugo Kielman (UL)
A Politiele gegevensverwerking en Privacy, Naar een effectieve waarborging
- 2010-10 Rebecca Ong (UL)
Mobile Communication and Protection of Children
- 2010-11 Adriaan Ter Mors (TUD)
The world according to MARP: Multi-Agent Route Planning
- 2010-12 Susan van den Braak (UU)
Sensemaking software for crime analysis
- 2010-13 Gianluigi Folino (RUN)
High Performance Data Mining using Bio-inspired techniques
- 2010-14 Sander van Splunter (VU)
Automated Web Service Reconfiguration
- 2010-15 Lianne Bodenstaff (UT)
Managing Dependency Relations in Inter-Organizational Models
- 2010-16 Sicco Verwer (TUD)
Efficient Identification of Timed Automata, theory and practice
- 2010-17 Spyros Kotoulas (VU)
Scalable Discovery of Networked Resources: Algorithms, Infrastructure, Applications
- 2010-18 Charlotte Gerritsen (VU)
Caught in the Act: Investigating Crime by Agent-Based Simulation
- 2010-19 Henriette Cramer (UvA)
People's Responses to Autonomous and Adaptive Systems
- 2010-20 Ivo Swartjes (UT)
Whose Story Is It Anyway? How Improv Informs Agency and Authorship in Emergent Narrative
- 2010-21 Harold van Heerde (UT)
Privacy-aware data management by means of data degradation