Interlinking Personal Semantic Data

on the Semantic Desktop and the Web of Data

Laura Cătălina Drăgan

Submitted in fulfilment of the requirements for the degree of
Doctor of Philosophy

SUPERVISOR:
Dr. Siegfried Handschuh

INTERNAL EXAMINER:
Prof. Dr. Stefan Decker

EXTERNAL EXAMINER:
Prof. Dr. Enrico Motta

Digital Enterprise Research Institute (DERI),
National University of Ireland, Galway
September 2012

The work reported in this thesis was supported by Science Foundation Ireland (SFI) under the DERI-Líon project (SFI/02/CE1/I131) and the DERI-Líon–II project (SFI/08/CE/I1380) and by the European Union 6th Framework under the NEPOMUK IP (FP6-027705) project.

Abstract

With the goal of improved Personal Information Management, the Semantic Desktop emerged as a solution to the problem of data fragmentation and disconnection on the desktop. It provides a framework based on Semantic Web standards and technologies, for interlinking desktop information. Interlinked data mimics the user's mental model, and as a result it is easier to manage, to find and refind, thus also offering a solution to the information overload problem. The Semantic Desktop overcomes limitations of the conventional desktop by using a common representation for the data, common vocabularies to describe it, and a desktop-central place to store it. Thus, it creates a network of linked desktop data, in a standardised format, accessible to all applications. However, having a relatively stable framework for the Semantic Desktop does not end the quest for improved Personal Information Management.

To users, the Semantic Desktop framework is a transparent layer, and as such, its benefits must shine through the applications which use it. And so the challenge of designing *good* semantic applications for the Semantic Desktop emerges. Important facets of this challenge include: presenting and visualising data, handling the relations among desktop entities, and incorporating new semantic information into the existing network of connected data. A second challenge is generated by the fact that the desktop, even a semantic one, is no longer sufficient for our information needs. The ever growing amount of online data, and the increasing volume of it which is available in structured form, enables further interlinking of personal semantic information with the Web of Data, to the benefit of the user.

In order to address these challenges, we follow the theme of interlinking personal semantic data in two threads through this thesis. First locally, within the Semantic Desktop, interlinking must be supported, and even more so *encouraged* by semantic applications built on top of the framework. We describe the challenges of designing and developing a semantic application for the Semantic Desktop, and present our SemNotes as an example, detailing the design decisions made along the way. The second direction extends to bridging the gap between the Semantic Desktop and the Web of Data, through connecting matching entities from the two spaces. This bridge opens up the possibility of using the rich data from the Web to complement and augment a user's personal information in new and innovative ways.

Contents

List of Figures

List of Tables

List of Listings

Part I

Prelude

Chapter 1

Introduction

We accumulate massive amounts of information on our computers and other devices, so much that it becomes daunting and sometimes useless because it is impossible to remember it all: emails, appointments, tasks, documents, people, conversations, etc. In traditional desktop architectures this information is created and stored in various applications, each an isolated island of data, with its own storage and its own format, unaware of related and relevant data in other applications. But in our mind this information is not separate, rather everything is connected by implicit links. When we use information we do not work with just one detached slice of it, but we mentally follow connections to other pieces of information from multiple sources. Information is inherently interlinked and used in an integrated fashion.

The Semantic Desktop emerged as a solution to the data fragmentation and disconnection problem. It provides a foundation and a framework based on Semantic Web standards and technologies, for interlinking information on the desktop. Interlinked information is easier to manage and organise, and because it mimics users' mental models, it is easier to find when required, thus also offering a solution to the information overload problem. The Semantic Desktop overcomes limitations of conventional desktops by using a common representation for the data, common vocabularies to describe it, and a central place to store it. It creates a network of linked desktop data, in a standardised format, which is accessible to all applications.

Several implementations of Semantic Desktops exist, with varied function sets and capabilities. NEPOMUK is a project undertaking the task of creating a blueprint for

the Semantic Desktop. The work presented in this thesis is grounded and built upon the framework provided by Nepomuk-KDE, a branch of the NEPOMUK project.

The most common use case for the Semantic Desktop is Personal Information Management (PIM). However, having a relatively stable framework for the Semantic Desktop does not bring the search for improved PIM to an end, rather it uncovers a path to a solution. There is plenty of work ahead, exploiting and improving the framework based on experience, research and testing. This thesis focuses on providing new and improved means of creating and maintaining the network of interconnected personal semantic data, on the desktop and beyond.

1.1 Problem Statement

The Semantic Desktop provides a solid framework for applications and systems to support the use of semantics on the desktop. However, having this framework is not enough, as the success and validation of any framework is reflected in its uptake and use. A key challenge that emerges is how to design and implement good semantic applications that use the infrastructure provided by the Semantic Desktop. This is a very vague and underspecified issue, that we divide into smaller problems for which we provide solutions. Important facets include presenting and visualising data, handling the relations among desktop entities, and incorporating new semantic data into the existing network.

The Semantic Desktop connects information in a network of linked personal data. Adding the, still mostly experimental, social aspect of sharing semantic personal data, and collaborating on it, we get clusters of networks of linked personal data. However, there is already a much larger network of Linked Data on the Web. The openly available Linked Data has grown extensively in recent years, as shown by the successive visual representations of the Linking Open Data cloud[1]. With such a vast network of Linked Data available, but unconnected to the desktop, the next step is to connect these two networks. This task is made easier by the fact that they share the same representation language, yet it is complicated by the different vocabularies used to describe the data, as well as the difference between the open world assumption of the Web versus the closed world assumption of the desktop. Building the bridge between these two domains raises

[1] `http://lod-cloud.net/` Linking Open Data cloud diagram, by Richard Cyganiak and Anja Jentzsch.

several challenges, from ensuring privacy and security, to finding the best information in the large amount of available data.

1.2 Research Questions

The above challenges are materialised into more precise research questions. At a general level, two questions will be tackled in the following chapters, each of them divided into more detailed sub-questions.

We start from the premise of the existing Semantic Desktop which solves the architectural issues, and gives a stable framework to build upon. However, it does not provide a complete ecosystem able to support the user in some of the more complex tasks. As such, we focus on two complementary directions:

Starting from the desktop environment, we look into:

Q 1. *How to build semantic applications and tools for the Semantic Desktop to provide the best experience for the users, while creating reusable semantic data?*

And then we expand the scope from the desktop to the larger domain of the Web of Data:

Q 2. *How to expand the scope of the Semantic Desktop into the realm of the Web of Data, to benefit the users and enhance their experience?*

We divide these questions into more specific sub-questions, that describe the facets of the issues tackled.

While building on top of the Semantic Desktop, there are additional challenges beside the ones raised by normal application design and development.

Q 1.1. *How to create semantic data that is correct, and maximises the reuse of existing Semantic Desktop data through interlinking?*

Due to the blackboard type of architecture of the Semantic Desktop, applications should not be developed in a vacuum, but rather, they should be aware of other data that is available to them, and also be aware that the data they produce will be accessible to other tools. This raises the dual challenge of making sure the data is represented correctly so that other applications can use it, if they choose to, as well as ensuring the privacy and security of the data created. While is is impossible to say that information

created is ever complete, applications must ensure that the data they create contains at least the minimum amount of information as to be usable by others. This requirement depends on the type of data created, but for new resources it includes basic properties such as a label to be used for display. Regarding the correctness of the data created, this requirement refers mainly to the way information is described according to the ontologies used, rather than being correct factually, which is a harder task and depends on factors outside the control of the application developers. Correctly described data contributes also to the last point of the question, enabling better discovery and reuse.

The possibility of reusing vast amounts of existing data raises other challenges, in selecting the right subset of necessary information for maximum benefit, while not overloading the users with it. The selection is complemented by the way the information is shown:

Q 1.2. *How to design the human-computer interaction in an application for the Semantic Desktop?*

The question refers to more than just displaying the information in an interface. Collecting input from the users, and generally supporting the creation of new semantic data is also a challenge. An extra difficulty is added by the heterogeneity of interlinked information available on the desktop, combined with the reduced control of developers over what resources are linked from other tools.

While it has been demonstrated by several studies that semantic tools are better than non-semantic counterparts [Sauermann, 2008, Franz et al., 2009] in supporting users in PIM tasks, evaluating semantic applications for the Semantic Desktop is still a difficult task to undertake. Therefore, the next question is:

Q 1.3. *How to correctly evaluate a semantic application?*

The challenge comes from the lack of a standard dataset on which to perform the evaluation. Even if such a dataset existed, due to the fact that the applications on the Semantic Desktop are mainly related to personal information, it is difficult for test participants to use the data provided, as they are not familiar with it. Evaluating PIM tools has been shown to yield more accurate results when the test users are asked to perform their own tasks in their own set-up, rather than attempting to simulate it with artificial tasks and data [Kelly, 2006]. The reduced number of semantic applications in each area make it difficult to evaluate a new tool against an existing established one, thus leaving the alternative of evaluating against an existing non-semantic application.

While possibly re-demonstrating that indeed linked data is more useful, comparing a semantic and non-semantic application requires well-thought metrics, as the semantic features that need to be evaluated have no direct counterparts against which to check them.

The second research question refers to connecting the Semantic Desktop to the Web of Data. The large amount of data available on the Web makes it difficult to find relevant information. An important part of connecting the Semantic Desktop to the Web of Data is finding common entities, like people or events, which might appear in both datasets. But entity resolution is hard when the input dataset is very large, and when only limited information is available for discriminating between similar entities. Therefore:

Q 2.1. *How to find instances representing the same real-world thing described by a Semantic Desktop resource?*

Examples include persons, organisations, projects, events, and documents. Since both networks of Linked Data (on the desktop and on the Web) use the same representation, improvements in the graph matching algorithms help, but the two sides are unbalanced and most often use different vocabularies to describe the data. Reconciling the desktop ontologies with the vocabularies used on the Web of Data requires mapping the "few" ontologies in the former category to the seemingly infinite number of vocabularies in the latter.

Q 2.2. *How to use the Web information which is related to a desktop resource?*

The answer to the question depends of course on the application that uses the information and the desktop resource, but some points are general: retaining the provenance of each piece of information imported; finding and handling conflicting information and judging trust; deciding if and when should local information be deprecated and replaced with new Web information.

Furthermore, connecting the Semantic Desktop to the Web of Data does not necessarily imply a unidirectional relation. A reverse link from the Web of Data to the Semantic Desktop should be considered as well. As always when making personal information available online, privacy concerns must be taken into account.

Q 2.3. *How to make desktop data available online safely?*

While accessing desktop information from the Web is technically possible, in principle it goes against the idea of a closed personal Semantic Desktop. The suitable way of

making desktop data available online is by publishing it, thus making it part of the Web of Data. Publishing information from the desktop is a simple enough task, but privacy issues arise when the data in question is highly interlinked with other private information.

Following the directions given by the two general questions, the approach we use focuses first on interlinking Semantic Desktop data through new semantic applications, and continues with linking the desktop to the Web of Data, through establishing new connections to the outside.

1.3 Approaches and Main Contributions

In order to answer the two research questions detailed above, the thesis describes two complementary directions for expanding the network of personal semantic data of the Semantic Desktop: creating connections within the desktop, and externally to the Web.

The first direction, creating new semantic data inside the Semantic Desktop, including new connections between entities, focuses on enabling the users to do so, through semantic applications. We recognise the need for semantic applications designed specifically for the Semantic Desktop, as the framework provides capabilities above what the conventional tools can support. To demonstrate solutions to the challenges of developing semantic applications mentioned above, we describe the design process and implementation of SemNotes, a semantic note-taking tool. Although a relatively small application, it covers all the phases of the semantic data life-cycle as described in [Möller, 2009], and does so in an unrestricted domain of use. We have written about developing SemNotes in [Dragan and Handschuh, 2009] and in more detail in [Dragan et al., 2011b].

The second, external direction of bridging the gap between the Semantic Desktop data and the Web of Data looks at capitalising on the common representation and structure of the data in the two networks. Because the two worlds are rarely disconnected, and a large part of the entities from the desktop appear online as well, we define and implement an algorithm which finds Web aliases for desktop resources. This part of the work initiated from linking publication information [Groza et al., 2009a] on the desktop and on the Web, and continued with a use case focused on publishing semantic notes as semantic blog posts without losing link annotations [Dragan et al., 2010]. The finalised algorithm and its evaluation are described in [Dragan et al., 2011a].

Chapter	Research questions
4 Creating and Interlinking Semantic Data on the Desktop with SemNotes	**Q 1.1. Q 1.2. Q 1.3.**
5 Bridging the Gap between the Semantic Desktop and the Web of Data	**Q 2.1. Q 2.2.**
6 Transforming Semantic Notes into Semantic Blog Posts	**Q 2.2. Q 2.3.**
7 Additional Extensions and Applications	**Q 1.1. Q 1.2. Q 2.1. Q 2.2.**

Table 1.1: The research questions tackled in each chapter.

To establish the background for the following work, we present a survey of Semantic Desktop systems and applications, which is an extended version of the one published as [Dragan and Decker, 2012].

1.4 Thesis Outline

The remainder of the thesis is structured as follows:

Part II — Foundations

We begin by setting the foundations for our work. This part is comprised of two chapters.

Chapter 2 describes the terminology and introduces some of the related work about the Semantic Web and its representative technologies, Linked Data, and Personal Information Management.

Chapter 3 contains an extensive survey of existing Semantic Desktops, from the visionaries that inspired them, to discussing their architectures, and their applications, the common points and what they do differently. This chapter also describes in more detail the NEPOMUK Semantic Desktop, the framework on which this thesis is built on.

Part III — Core

The core of the thesis presents our work, divided in four chapters: the first two reflect the approaches used to solve our two main research questions, and the last two contain respectively a combination of the approaches into one unified use case, plus a set of extensions complementing the work.

Chapter 4 describes our solution to the first research question **Q 1.** — how to build semantic applications for the Semantic Desktop, and all of its sub-questions. To illustrate the solution, we present the design and implementation of SemNotes, a semantic note-taking tool for the Semantic Desktop, as an example application. We use note-taking as an example because it is a desktop activity that is not restricted to a specific domain, as the notes can vary widely in topic. It is also a common activity that plays an important role in Personal Information Management and one that we believe would benefit from the use of semantic technologies.

Chapter 5 describes our solution to the second research question **Q 2.** — how to expand the scope of the Semantic Desktop into the Web of Data, by providing an algorithm, and its implementation, for finding Web aliases of desktop resources. The work is motivated by many of the resources from the Semantic Desktop, also appear on the Web, and the information available about them online could be used to augment the local data. By creating the connection between the local resources and their Web counterparts, we make the first step towards truly personalised desktop services using Web data, and also enabling the publication of desktop data safely online.

Chapter 6 describes a system that uses of the work from the previous two chapters, publishing online the semantic notes taken with SemNotes, as semantic blog posts. Locally, the notes are well integrated and interlinked with the rest of the semantic network of desktop data. However, as the connections are to local resources, publishing the note as a blog post directly would lead to broken links and the context of the note being lost. On the other hand, including the related desktop information as metadata with the blog post might lead to privacy issues. However, because the relevant desktop resources have Web aliases, we can replace the local links with Web links and preserve the context without the risk of exposing any private information.

Chapter 7 contains short descriptions of several applications and extensions for the Semantic Desktop, which complement the main work presented before. We begin with extensions to SemNotes which were built to incorporate and test existing

research in the area of Natural Language Processing, and which could be applied to note-taking, enhancing the user experience. We also describe the process of linking publication data with Sclippy, the tool that initiated our work on connecting the desktop with the Web of Data. Another application for the Semantic Desktop that could connect to the Web is Konduit. It was envisioned as a simple user friendly tool that allows users to create their own applications with semantic data, by visually defining workflows.

Part IV — Conclusion

Chapter 8 contains the conclusion of the work, reiterating the contributions and how they answer our research questions. We discuss some of our results and the insights we gathered from the work. We also outline ideas for future research.

Part II

Foundations

Chapter 2

Background

This chapter presents the terminology used in the rest of the thesis. We discuss the evolution of the Web into the Semantic Web, and how it became popular as Linked Data and through the Linking Open Data project. We describe some of the core concepts and technologies of the Semantic Web, like ontologies and RDF. We also present a list of Personal Information Management studies, and we define the meaning with which the term "personal information" is used in this thesis.

2.1 The Semantic Web

2.1.1 A Quick History of the Web

The idea of hypertext was at the basis of many of the developments of the Web. It was invented simultaneously in the 1960s by Engelbart and Nelson (see Section 3.1 for more details) [Engelbart, 1962, Engelbart and English, 1968, Nelson, 1965]. "Enquire" was created by [Berners-Lee, 2000] at CERN twenty years later, in the early 1980s, and it was the first PIM system to use hypertext .

So far, the systems following the idea of hypertext, including Enquire, were limited to work with information from a single computer. Starting from 1984, Berners-Lee started to extend the system, to enable it to point to documents from other machines, using the existing Internet infrastructure, developed in the 1970s partly by the old hypertext community. Based on this extension, he proposed a system called "Mesh" [Berners-Lee, 1989], which incorporated technologies that have developed into today's

Web: *HyperText Markup Language* (HTML), *HyperText Transfer Protocol* (HTTP) and *Universal Resource Identifiers* (URIs). It was at CERN than Berners-Lee developed the first applications to use these technologies: the first Web browser and the first Web server. The "Mesh" was not a success when it was first created, but it was an open and distributed architecture, features that allowed others to develop their own browsers and publish their own documents.

In Berners-Lee's implementation, Web documents were both readable and editable, but the editing functionality was ignored by the other browser implementations. Editing was both a very difficult task, and one that was not considered as important. Mosaic (1993) was the first widely available browser, which made the "Web" gain more and more popularity. The number of Web servers also grew, and so did the number of pages published.

2.1.2 Semantic Web Technologies

The World Wide Web [Berners-Lee et al., 1994] allowed people to publish information easily. It provided read-only access to an immense quantity of information, which grew exponentially, as the technology evolved, and as having access to a faster and more reliable Internet connection became wide-spread. With the Semantic Web we move on from a Web of documents understood by humans to a Web of machine understandable information [Berners-Lee et al., 2001]. But this was not something new: most of the ideas we now attribute to the Semantic Web — typed links and nodes — were in fact present in the initial proposal of the Mesh, but they were omitted in favour of simplicity.

Thus, the goal of the Semantic Web is to add (machine understandable) meaning to the huge repository of connected information that is the Web. To accomplish this, the Semantic Web uses the same infrastructure and standards as the Web, along with additional technologies, which make up the "Semantic Web layer cake"[1] shown in Figure 2.1. In this section we will discuss some of the technologies that make up the layers, and which are relevant for the following chapters.

[1] `http://www.w3.org/2007/03/layerCake.svg`

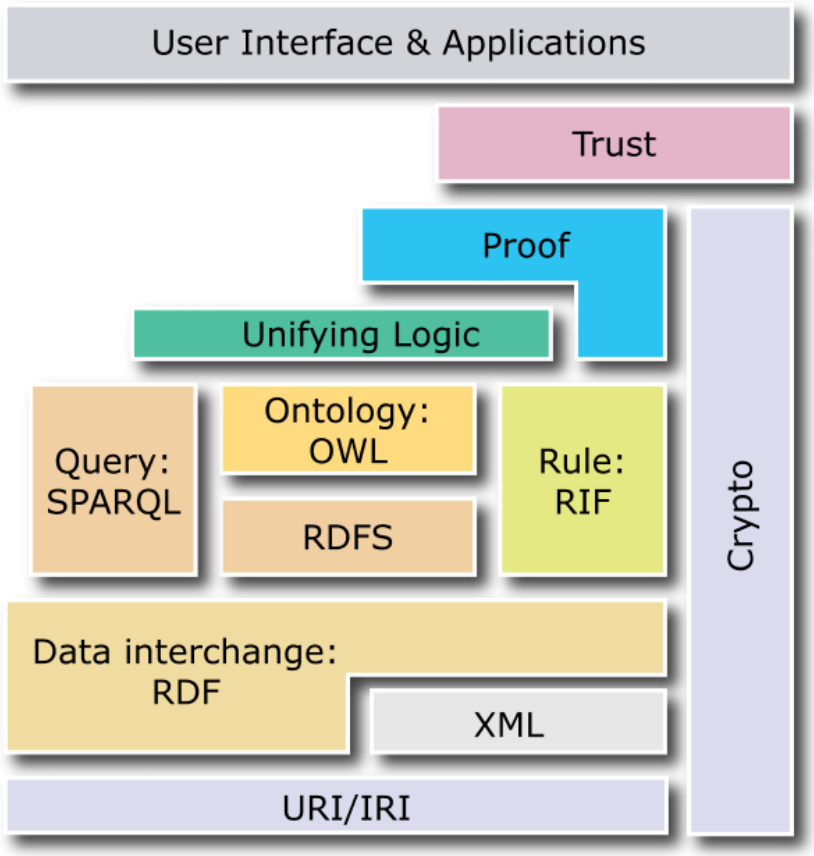

Figure 2.1: The Semantic Web layer cake.

The Resource Description Framework (RDF)

"The Resource Description Framework (RDF) is a framework for representing information in the Web." [Klyne and Carroll, 2004] It defines a standard model for representing and exchanging information on the Semantic Web. The RDF Specification has been a W3C Recommendation since 1999 [Lassila and Swick, 1999], with the latest version being a suite of six W3C Recommendations[2], published in 2004.

[2] `http://www.w3.org/standards/techs/rdf`

```
@prefix ex: <http://www.example.org/RDFexample#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix bibo: <http://purl.org/ontology/bibo/> .
@prefix dcterms: <http://purl.org/dc/terms/> .

ex:doctorow a foaf:Person ;
   foaf:name "Cory Doctorow" ;
   foaf:firstName "Cory" ;
   foaf:surname "Doctorow" ;
   foaf:homepage <http://craphound.org> ;
   foaf:made ex:ftw .
ex:ftw a bibo:Book ;
   dcterms:title "For The Win" ;
   foaf:maker ex:doctorow ;
   bibo:issn13 "9780765322166" .
```

Listing 2.1: RDF example represented in Turtle.

RDF is designed to allow flexible representation of information. The underlying structure of any data represented with RDF is a graph, which is a collection of triples. Each triple represents an RDF statement, and is made up of a subject, a predicate (or property), and an object. An example is shown in Listing 2.1. A triple can be seen as a graph made of two nodes, the subject and the object, connected through an arc, represented by the predicate. Thus, a set of triples can be represented as a graph by the corresponding set of nodes and arcs (see Figure 2.2 for an example). Any information about a resource can be expressed through triples, including information about a triple, through a process called reification. While a powerful feature, reification adds complexity, and is usually used sparsely. An extension to RDF allows statements to be grouped in *named graphs* which helps avoid the use of reification to store meta-information about triples, like provenance for example.

The RDF specification defines three types of elements:

- identified resources, which are represented by a URI. They can appear on any position in a triple.
- unidentified resources, called blank nodes, which cannot appear as predicates in triples. They are unnamed nodes within a graph, and using them can pose problems, thus it is discouraged [Bizer et al., 2007].
- literals, which can only appear as objects in a triple. Literals represent values of properties. They can be typed, using XML Schema datatypes, or can have a language tag.

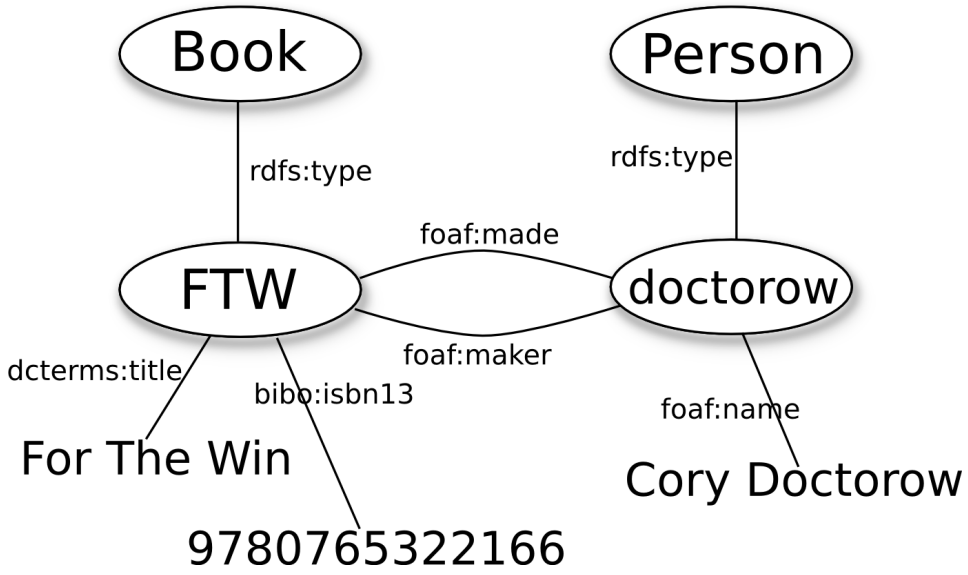

Figure 2.2: RDF example represented as a graph.

RDF is a data model which can be serialised in several ways. The most popular serialisation of RDF was first RDF/XML [Beckett, 2004], due to the popularity and familiarity of XML. More recently however, other serialisations, more suitable for human consumption, have become preferred: N-Triples [Grant and Beckett, 2004], Turtle [Beckett, 2007], and N3 [Berners-Lee, 2006b]. RDFa [Adida et al., 2008] is another syntax for RDF, which allows embedding of RDF statements in HTML pages. In the following chapters of the thesis, we will use the Turtle notation in listings.

SPARQL [Prud'hommeaux and Seaborne, 2008] is the recommended query language for RDF. Other query languages for RDF exist, including:

- RDF Query Language (RQL) [Karvounarakis et al., 2002],
- Sesame RDF Query Language (SeRQL) [Broekstra and Kampman, 2003],
- RDF Data Query Language (RDQL) [Seaborne, 2004].

Ontologies and Vocabularies

RDF gives us the means to represent information in a unified way. However, this is not enough to make the information understandable to machines. Something more is needed, and that is a common, agreed-upon, and shared vocabulary to refer to things. This is represented by the "Ontology" layer in the SW layer cake. According to Gruber,

an ontology is "an explicit specification of" [...] "an abstract, simplified view of the world that we wish to represent for some purpose." [Gruber, 1993]

RDF Schema (RDFS) [Brickley and Guha, 2004] was created to support the use of RDF to define ontologies. The Web Ontology Language (OWL) is another formal language to define ontologies, which offers more expressive constructs than RDFS.

Both languages allow the definition of classes, hierarchies of classes, properties with domain and range, and hierarchies of properties.

Ontologies can be: *upper-level* (or top-level, or foundation) ontologies, defining high level, generic concepts, or *domain* ontologies, which define concepts related to a specific, restricted domain.

Numerous vocabularies have emerged for the Semantic Web. Among them, a limited number have gained wide-spread adoption and recognition. The ones we use and refer to in this thesis are listed below:

- Dublin Core Metadata Initiative (DCMI[3]) [Nilsson et al., 2008] — for "core metadata for simple and generic resource descriptions";
- Simple Knowledge Organization System (SKOS[4]) [Miles and Bechhofer, 2009] — for sharing and linking data about "knowledge organisation systems, such as thesauri, taxonomies, classification schemes and subject heading systems";
- Friend Of A Friend (FOAF[5]) [Brickley and Miller, 2005] — for describing people and organisations;
- Semantically-Interlinked Online Communities (SIOC[6]) [Breslin et al., 2005] — for describing online communities;
- Description Of A Project (DOAP[7]) — for information about software projects;
- the Music Ontology (MO[8]) — for "describing music (i.e. artists, albums, tracks, but also performances, arrangements, etc.)";
- GeoNames[9] — for geographic information.

[3] `http://dublincore.org/`
[4] `http://www.w3.org/2004/02/skos/`
[5] `http://foaf-project.org`
[6] `http://sioc-project.org`
[7] `https://github.com/edumbill/doap`
[8] `http://musicontology.com/`
[9] `http://www.geonames.org/ontology/`

2.1.3 Linked Data

The term Linked Data was first introduced by Berners-Lee in 2006 to define a set of best practices for publishing data on the Web [Berners-Lee, 2006a]. They became known as the "Linked Data Principles" and describe the recommended way to publish and connect data on the Web, so that "it is machine readable, its meaning is explicitly described, it is linked to other external datasets and can in turn be linked from external datasets".

In addition to these principles, the more recent Linking Open Data[10] project enables the creation of a huge amount of interlinked RDF data on the Web, from various open datasets, ranging from Health Care and Life Sciences (HCLS) information to the BBC programmes.

2.2 Personal Information Management

Personal Information Management has been a concern and research topic from the early days of computing, and even before. Bush's Memex (see Section 3.1.2) was the first envisioned automatised PIM system. It acted as an extended personal memory, an archive of all the personal information, and a holder of explicit links between this information. The vision of the Memex was carried over through the evolution of the personal computer (and recently many other personal devices as well).

2.2.1 Personal Information

To better understand what Personal Information Management is, we first need to define *personal information* in this context. There is a large body of research done in the area of PIM, and as such, there are several definitions of what personal information is, some more restrictive and some more broad. However, an important distinction to make is that personal information is not *Personal Identifiable Information (PII)* which is a piece of data that uniquely identifies a person. Personal information is also not restricted to private information. It is true that some of our personal information is private, but not all of it. As [Lansdale, 1988] described: "personal information is information not in a sense that it is private, but that we have it for our own use. We own it and would feel deprived if it would be taken away." In his Ph.D. thesis, Boardman defines Personal

[10] `http://linkeddata.org`

information as "information owned by an individual and over which this individual has a direct control" [Boardman, 2004].

Jones [Jones and Teevan, 2007, Jones, 2008] defines personal information from the point of view of its relation to a person, through six overlapping categories:

- *controlled by (owned by) me* — the information the person keeps, directly or indirectly, for personal use;
- *about me* — the information about a person, but available to, and possibly under the control of others;
- *directed toward me* — email received, or advertisements seen, this information itself may not be relevant to the person, but its impact is personal;
- *sent (posted, provided) by me* — this information is no longer under the control of the user after it has been sent;
- *(already) experienced by me* — like a radio show, a book read, or a web page seen, this information may or may not be under the control of the person;
- *relevant (useful) to me* — this category includes information from the previous categories, but also new information, which has not been seen before, but is relevant or useful.

In this thesis, when we refer to personal information, we mean it in the broadest sense of the term, as in this definition by Jones.

2.2.2 Personal Information Management

The goal of PIM is to help the user find the right information at the right time for the task at hand, with as little effort as possible. The means to achieve this is to enable the user to organise the information better, and over time, organisation has transformed from the means to a goal in itself.

As with the definition of personal information, there are several definitions for PIM. [Lansdale, 1988] describes it as "the methods and procedures by which we handle, categorise, and retrieve personal information on a day-to-day basis." [Barreau, 1995] defines a PIM system as "an information system developed by or created for an individual for personal use in a work environment." According to [Boardman and Sasse, 2004], PIM is "the collection, storage, organisation and retrieval of items of digital information (e.g. email, files, appointments, reminders, contacts, bookmarks) by an individual in their per-

sonal computing environment." [Jones, 2008] formally defines PIM as referring to "both the practice and the study of the activities a person performs in order to acquire or create, store, organise, maintain, retrieve, use and distribute the information needed to meet life's many goals (everyday and long-term, work-related and not) and to fulfil life's many roles and responsibilities." [Lansdale, 1988] describes PIM in relation to psychology — showing how classification is a difficult activity due in part to ambiguous terms, and how memory affects recall. The term "mental model" was coined by [Craik, 1943] and has been used and studied [Norman, 1983, Paivio, 1986, Johnson-Laird, 1989] since then. Research in psychology and cognitive sciences has shown how mental models are created, how they evolve and how they support learning, collaboration and information retrieval [Järvelin and Wilson, 2003, Payne, 2003, Jones et al., 2011]. An overview of the field and its connection to PIM is presented in [Nadeen and Sauermann, 2007]. Conceptual models have been described and used by both Bush and Engelbart as foundations for their work.

According to [Barreau, 1995], PIM systems can be categorised using the following criteria, which are still valid today:

- *acquisition* – how and what information enters the system;
- *classification and organisation* — how the information is grouped and labelled;
- *storage* — how the information is stored for later retrieval;
- *maintenance* — how the information is updated and migrated if needed;
- *retrieval* — how the system enables retrieval of information at the appropriate time, this being one of the most important functions of PIM, the majority of the other characteristics revolving around and working towards the eventual retrieval of information;
- *output* — how the system can answer queries related to the user's information and context.

Many user studies have been conducted to determine the requirements for different PIM tasks, how people tend to solve problems that occur frequently, how they generally organise their information — be it on their physical workspaces or on their virtual ones. One of the most notable studies is that of [Barreau and Nardi, 1995], which "investigated information organisation practices of users". They categorise the information handled by the participants into three categories: ephemeral, working, and archived — with the very interesting observation that one of the concerns raised by the study was the large amount of ephemeral information that the users had to cope with, which accumulated

and cluttered the workspace. The management of this ephemeral information is the focus of the study by [Bernstein et al., 2008] on "information scraps", defined as "is information items that fall outside all PIM tools designed to manage them".

Another finding of the Barreau and Nardi study was the "preference for location-based search for finding files" and "the critical reminding function of file placement". [Fertig et al., 1996a] argue the need for better systems for organising the information than those offered by the desktop metaphor at that time. They suggest that although people make the best of the tools they are given, that does not mean that the tools are suitable for the task, and that better alternatives should be researched. Their examples include the Lifestreams [Freeman and Fertig, 1995, Fertig et al., 1996c] system which proposes a chronological organisation of documents, and the MIT Semantic File System [Gifford et al., 1991] which offers an associative organisation similar to the vision of the Memex. [Civan et al., 2008] describes another user study focused on comparing location-based organisation in folders with category-based organisation with the use of tags, in the context of email management. The study by [Blanc-Brude and Scapin, 2007] determines which attributes enable the best re-finding of information in a PIM system, and how tools for information retrieval could be improved by supporting the combination of attributes most likely to lead the user to the desired result.

Many applications have been created for each of the supporting activities of PIM: document management, contact management, email, calendaring, task management. These applications serve their chosen domain well, offering rich sets of features and capabilities. However, the information that each of them handles is rarely used on its own, or separate from the rest, thus integrated information is a requirement of PIM. Additionally, a prevalent problem is the locking of information in application specific repositories, and in application specific formats, making integration difficult. Another problem is information fragmentation in multiple places and applications. [Boardman and Sasse, 2004] describes a cross-tool study to determine how users manage their personal information within applications and across PIM tools. The findings show that most of the time the information is classified in similar hierarchies, and as a result there is duplication of the work required to organise information in each tool. From another longitudinal study Boardman classifies users into categories, depending on their behaviours, and presents insight into how these behaviours evolve over time. [Kaptelinin and Boardman, 2007] describe two different approaches to information integration: through a single "mega-application" which supports multiple PIM activities, or through the coordination of several applica-

tions in a unified workspace. The problem of information fragmentation and solutions for unification or integration are also found in [Karger and Jones, 2006, Karger, 2007].

Another challenge relevant to building and improving PIM tools is how to evaluate them. The problem has been discussed by [Whittaker et al., 2000] and also by [Kelly, 2006, Elsweiler and Ruthven, 2007, Bernstein et al., 2007]. The main difficulty originates from the personal nature of the information handled by the systems, and the familiarity of the users with this information, familiarity which cannot be replicated in artificial conditions. It extends to the fact that the tasks that users work towards solving with the PIM systems are also highly personal, thus artificial tasks for the purpose of evaluation would not yield clear results.

2.3 Conclusion

In this chapter we presented the terminology that is used in the thesis, both from the area of the Semantic Web and that of Personal Information Management. We gave an overview of the Semantic Web, its evolution from the document Web, and its core technologies. We have also described what "personal information" refers to in the context of Personal Information Management, and what are the research areas, and challenges of the domain.

We continue to show in the next chapter how the semantic technologies have been applied on the desktop and to PIM, to create Semantic Desktop systems. These systems are apply the most basic elements of semantic technologies, the categorisation of entities into classes and the explicit specification of links, to representing the mental models with the help of computers.

Chapter 3

The Semantic Desktop

Based on "Knowledge Management on the Desktop" [Dragan and Decker, 2012] published at the 18th International Conference on Knowledge Engineering and Knowledge Management (EKAW2012)

Having defined the semantic technologies used on the Web, we continue in this chapter to show how they have been applied on the desktop and to Personal Information Management, to create the Semantic Desktop. We provide an in-depth overview of existing Semantic Desktop features and architectures, since our work is grounded in the framework given by the Semantic Desktop, enhancing and complementing its features.

3.1 Visionaries

The ideas behind the Semantic Desktop have been around for much longer than the actual devices (personal computers, netbooks, tablets or even smartphones) on which they run. The common problem the Semantic Desktops are trying to solve is finding better ways for knowledge workers to manage the ever-growing amount of information they need to work with and process. The most relevant historical precursors of the Semantic Desktop are, in chronological order:

Paul Otlet's Universal Bibliographic Repertory is a Belgian enterprise from the end of the 19th and the beginning of the 20th century, trying to set up a comprehensive system of all the bibliographic knowledge available, and keep it up to date as new knowledge appeared. It used a system of interlinked index cards, with reference codes pointing

to relevant information, thus creating the first network of knowledge, although not an machine readable one. Despite not being widely recognised as such, it does represent the beginning of standardised knowledge and information management.

Vannevar Bush's Memex — cited by most "modern" Semantic Desktops as inspiration, describes a first vision for a truly personal system for interlinking information. Although Bush's seminal paper "As We May Think" was published in 1945, his work on the Memex started well before that.

Douglas Engelbart's Augment — built on the vision of Memex, the Augment was the first "real" functional system. As of 2012, it is still being used by its creator, despite being created in the early 1960s. Engelbart's ideas on bootstrapping are also relevant in the evolution of Semantic Desktops.

Theodor Nelson's Xanadu — imagined at roughly the same time as Augment, the Xanadu system was never completed. However, the ideas behind it inspired many of the future developments that shaped the Web and many technologies that exist today.

3.1.1 Paul Otlet (1868 - 1944)

Although Bush, Engelbart and Nelson's works are usually cited as inspiration and background in most works about the Semantic Desktop, Paul Otlet's work is relevant through its influence on them. His work anticipated many of the features of the other historical systems described here. Otlet, a Belgian lawyer turned bibliographer, was one of the (grand-)fathers of information science. He wrote numerous essays on the need for an international system for handling, indexing, linking, and accessing information, as a way of containing and managing the vast knowledge produced by man. Otlet created the Universal Decimal Classification (UDC) as the solution to the problems he described in his essays. The UDC pioneered the use of standardised index cards which allowed for continuous interlinking and referencing of entries, and represents one of the first faceted classification systems. The UDC was used to create a huge database called the Universal Bibliographical Repertory, which had 15 million entries by the late 1930's. The UDC is still in use throughout the world, although in revised and updated forms. It can be seen as precursor to high level taxonomies of concepts for classification, with notions of "narrower" and "broader" subject terms. Otlet's activity and vision are reflected in his chef d'oeuvre "Traité de documentation" [Otlet, 1934]. It is seen as precursor and motivator to the hypertext, mark-up languages and even the Semantic Web.

3.1.2 Vannevar Bush (1890 - 1974) and the Memex

Bush was an American scientist and public figure. During the World Wars he coordinated researchers from many domains, towards applying scientific advances to warfare and defence. He was an unofficial scientific advisor to two US presidents[1], and led several institutions and committees, along with initiating and supporting the creation of the National Science Foundation in the United States. Throughout his career Bush passionately supported collaboration between researchers, which he saw as the fastest path to the progress of humanity. He went as far as to propose an exchange of scientific results between the United States and Russia during the Cold War, to promote collaboration and prevent the atomic bomb race.

"As we may think" (1945)

Bush was deeply concerned with the continuation of the collaboration between scientists after the Second World War was over. Himself a scientist, Bush was also painfully aware of how the progress in science meant that more research results are published than can be followed, and that because of old and inadequate publishing systems, as well as the sheer amount of information, much of the new research is unreachable, lost or overlooked.

In a lengthy essay titled "As We May Think" [Bush, 1945], published in The Atlantic Monthly and later in Life, Bush describes several devices, possible inventions of the future, as solutions to the problems of research publishing and as teasers of tantalising scientific advances. Some of the devices may seem amusing in the context of current developments, but looking back, they are remarkable through their accuracy and foresight. The most famous of the devices described by the article is the Memex — "... a future device for individual use, which is a sort of private file and library", "... a device in which an individual stores all his books, records and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility". Leaving aside the physical description of the device, which was envisioned in the form of a desk with screens, drawers for storage, a keyboard, buttons, and levers, Bush has more or less described the functions of a present day personal computer — a device where we store and consult personal documents and communications. Furthermore, the Memex, attempts to mimic the way associations in the brain works, by using "associative trails" of connected things. The trails can be created, modified, browsed, shared and collaborated

[1] Franklin D. Roosevelt and Harry S. Truman

on. "Associative indexing" was "the essential feature of the Memex." Bush presents a greater purpose for his Memex — creating new forms of linked (associative) encyclopaedias. They would serve as a record of the human knowledge accumulated over time and would "accelerate technical progress".

Although the article was published in 1945, Bush had been working on the idea of the Memex for many years before, and although there are no clear indications of any influences from Otlet's work, their higher end goals are similar. Some claim that Otlet's work was indeed known to Bush, indirectly through Watson Davis' visit in 1932 and there is some overlap between the works of the two to support it[2] [Veith, 2006]. However, Bush writes against indexing systems, that he sees as artificial and cumbersome, and one of the reasons why information is hard to find. But the Memex could benefit from something like Otlet's UDC, a common system for classification of the trails. The trail names used in the Memex are instead highly personal and cryptic, which can become a problem when it comes to sharing the knowledge [Buckland, 1992].

The elegant ideas behind the Memex influenced the fields of information science and information management, and the development of personal computing, hypertext and semantic technology. The Memex is the first Semantic Desktop described in the literature, and although it was never realised, it has influenced the works of both other two visionaries included in this section, Douglas Engelbart and Ted Nelson.

Bush continued to work on the project, and wrote further articles on the Memex, like "Memex II" from 1959, and "Memex Revisited" from 1967, which are included in the book [Nyce and Kahn, 1991].

3.1.3 Douglas Engelbart and Augment

From a desire to "improve the lot of the human race" [Goldberg, 1988], Douglas Engelbart has devoted his career to "augmenting the human intellect". In pursuit of this goal, he invented many devices that greatly influenced the way we work with computers today: the mouse, the window, the word processor, video conferencing, remote procedure calls, hypertext, and more. However, all these notable advances were just first steps towards his greater purpose, that of achieving an augmented human intellect.

[2] edwardvanhoutte.blogspot.com/2009/03/paul-otlet-1868-1944-and-vannevar-bush.html

"Augmenting the Human Intellect" (1962)

The 1962 report "Augmenting the Human Intellect" [Engelbart, 1962] describes a conceptual framework for his research, presenting the details of the H-LAM/T system (Human using Language, Artefacts, Methodology, in which he is Trained). It defines four classes of possible augmentation means, through intertwining artefacts, language, methodology and training. It also discusses thought process and repertoire hierarchies, and how they influence problem-solving capabilities. Furthermore, Engelbart defines mental structures, concept structures and symbol structures, and their relationship to each other. He introduces the hypothesis that "better concept structures ca be developed – structures that when mapped into a human's mental structure will significantly improve his capability to comprehend and to find solutions within his complex problem situations."

The report thoroughly quotes and discusses Bush's "As We May Think", which Engelbart believes "to make a very convincing case for the augmentation of the individual intellectual worker." He focuses on Bush's description of the Memex, and highlights how the device fits within the conceptual framework of augmentation. He also shows how the associative trails pioneered by Bush map to the concept and symbol structures from H-LAM/T, and how they can be evolved further, in the context of a note-card system. Apart from the high-level ideas that precede and build up to the ideas of the Semantic Desktop, the Augment report also describes and motivates the need for a well designed semantic model for such a system. In an exercise of forethought, Engelbart says that classification "might be the most significant part of the work." He envisions that the extra effort required "to form tags and links [...] consciously specifying and indicating categories" would be rewarded by the capacity gained by the computer to understand and perform more sophisticated tasks. He describes the ontology creation model - establishing the categories and relations between them, dealing with and removing semantic ambiguity.

"A Research Center for Augmenting Human Intellect" (1968)

The [Engelbart and English, 1968] report describes the system that was shown at the 1968 Fall Joint Computer Conference in what has become known as "the mother of all demos" because of the size, novelty and sheer number of innovations shown together at one time. The demo, and report, are outstanding for many reasons — the infrastructure required to run it, the number of people involved, the introduction of the mouse,

the collaborative work environment, the first word processing software, and many more. However, from the Semantic Desktop point of view, the spectacular part comes from its realisation of the conceptual framework detailed in the 1962 report [Engelbart, 1962] — the introduction of explicit hierarchical structuring of information — "we adopted some years ago the convention of organising all information into explicit hierarchical structures, with provisions for arbitrary cross-referencing among the elements of a hierarchy"; "the symbols one works with are supposed to represent a mapping of one' s associated concepts, and further that one's concepts exist in a network of relationships." Allowing the creation of arbitrary links between elements of the hierarchy, and providing unique names for entities for better and faster access, represents the realisation of Bush's associative trails and makes Engelbart's On-Line System (NLS) the first functional predecessor of a Semantic Desktop. The semantics are limited to the network of statements in files, but this is an important first step towards deeper interlinking.

3.1.4 Theodor Nelson and Xanadu

By his own account, Nelson was not an engineer but a philosopher and cinematographer, and "a computer fan, computer fanatic if you will" [Nelson, 1974]. However, he saw the potential that computers had beyond data processing and computation. He envisioned a way of using the computer as "an adjunct to creativity" for writing and personal information management, in the style of Bush's Memex for personal use.

Nelson foresaw the information overload crisis we are facing, and many of the developments that created it: the personal computer, enhanced communications, digital publishing, virtually infinite storage. He has imagined a solution to the problem — a "psychic architecture" of a system [Nelson, 1973], where psychic refers to "the mental conceptions and space structures among which the user moves". The idea was embodied in a system called Xanadu, the specification and requirements of which are detailed as early as 1965 in [Nelson, 1965]. Xanadu transformed over time, from an "effort not to forget" and to organise personal information better, into a vision of unrestricted linking of passages of text.

Nelson proposes for Xanadu a new file structure called ELF (evolutionary file structure) made up of entries, lists and links, as a way or realising the associative trails of the Memex. The zippered list structure described by him allows "any two entries to be connected", by "link-modes having different meanings to the user" — similar to how we use RDF in current semantic systems. Nelson shares with Engelbart the credit

for the invention of hypertext (hypermedia and hyperfilm), where "hyper" has roughly the mathematical sense of "extended, generalised, and multidimensional". Hypertext's purpose was to reflect "the real structure of the thoughts expressed" by users. The freedom to link anything with his version of hypertext was supporting what Nelson called transclusion, the process of including something by reference rather than by copying — "quoting without copying". Already in his 1973 article [Nelson, 1973] he mentions concerns for the security of hyperlinked data, and based on transclusion, Nelson invented a new type of copyright management called transcopyright. Zippered lists also have evolved into zzstructures, which are at the basis of ZigZag hyperstructure toolkit and the ZigZag personal environment [Nelson, 2004].

The Xanadu system was in development for many years, but it has never become fully functional. Despite it not being a success, the vision behind it has inspired many dedicated followers and influenced today's world of personal computing.

3.1.5 Influences and Influencers

There is a distinguishable network of influence in the domain of networks of knowledge.

It is uncertain if Otlet's work has in any way influenced the inception of the Memex across the Atlantic at the same time, but the idea is supported by the fact that he "reported enthusiastically on experiments with other bibliographical applications of technology, especially microfilm" [Goldschmidt and Otlet, 1906, Otlet and Rayward, 1990]. Furthermore, "in the early 1930's Otlet began to speculate about how a wide range of then experimental technology — radio, cinema, microfilm, and television — could be combined to achieve a new complexity and variety of functionality in information searching, analysis, re-structuring and use" [Rayward, 1991].

The influence of Bush's Memex on the works of Engelbart and Nelson is, however, undisputed, and stated by both in their writings. The directions were different though: Engelbart NLS was designed by engineers, focused on their needs, and encouraging collaboration. Xanadu on the other hand was intended for personal use, for capturing the train of thought of one user and to serve as an extended memory. Some of the features envisioned by Nelson for Xanadu were incorporated in Engelbart's NLS — the ability to link paragraphs of text, revise documents in real-time by moving pieces of text on the screen, preserve and track changes.

3.2 A Survey of Semantic Desktops

We have covered so far the futuristic systems of the past and the visionaries who imagined them, like the Memex, which since 1945 have been an inspiration for most of the work in personal computing and personal information management. We have also briefly described the Semantic Web and its technologies. We will now show through a list of modern Semantic Desktops how semantic technologies are put to use in those systems to realise the vision of the Memex, and more.

3.2.1 Common Challenges

The term *Semantic Desktop* was coined in 2003 by Decker and Frank and was taken up first by Sauermann in Gnowsis [Sauermann, 2003].

The [Social] Semantic Desktop saw the fastest rise and uptake during the 2004 – 2009 period, when most research work was put into the development of systems and applications, new and improved algorithms to support knowledge work. These systems and tools were aiming to solve similar problems as was the Memex before them: supporting and improving knowledge work by mimicking the way the brain works — using Bush's associative trails reworked with the help of the semantic technologies. The fast growth of the digital world, while providing easier access to information and communication, has created new problems, foreseen by Nelson in the 70s: there is so much information available that it becomes unmanageable.

Thus, the challenges that the Semantic Desktops aim to solve can be described as follows:

Information overload caused by the ease with which information can now be generated, shared, stored and accumulated. The amount of information that we see, receive, create every day is enormous. While the storage and processing capacity of hardware has increased many-fold, re-finding stored information has become a more complex task. The challenge is to enable users to manage this large amount of information better, while not spending more time and effort organising it.

Data silos are created by desktop applications which store their data in proprietary formats, in storage inaccessible to other applications which could possibly reuse that information, thus each mus recreate it in their own formats and store it in their own walled repositories. The negative effects of this vicious circle are *data duplication*,

and *disconnected information*. Having the same data in multiple applications leads to difficulties in keeping it up to date and synchronised. Additionally, having pieces of information relating to the same object or entity spread over several locations and tools makes working with that information more difficult, as it requires opening and browsing through several folder hierarchies and application structures, as well as piecing together the information every time it is needed. The challenge presented by data silos is to give users a unified and consistent view over their data, regardless of its source and location.

Dynamic data means that information changes over time, and thus it can become deprecated. Using deprecated information can have undesired effects, like sending emails to an old email address of a contact. The data silos problem above adds complexity to this — *duplication of data* in various applications makes it more difficult to decide which version is correct; *disconnected information* means that when a piece of information changes, care must be taken so that the changes are made in every place where the information is kept, increasing the effort required.

Associative trails as those described by Bush, follow the way we think about things, by connections and associations, unconstrained by the way the information is stored or presented in various applications. This point is also related to the data silos issue. As we discussed above, studies showed that the way we form mental models and work with information is task and data centric. As long as data is disconnected and locked away in application repositories, explicit associative trails can only be fragmented, short and thus hurting productivity.

The 2004 white paper by [Decker and Frank, 2004] ambitiously describes a solution, the Social Semantic Desktop as "a novel collaboration environment, enabling the creation, sharing and deployment of data and metadata" and names the sources that would contribute to the development of such a system: "Semantic Web, Peer-to-Peer Networks, and Online Social Networking." From the point of view of the future Social Semantic Desktop, the Semantic Desktop was just one piece of the puzzle, its function being that of a smart personal information management tool, that provides a common vocabulary, creates and maintains a network of interconnected personal information, stores it and gives users access to it through search and browsing.

The 2005–2006 workshop series on the Semantic Desktop, and the 2005–2008 series of hands-on workshops[3] served as a catalyst for early ideas and brought together researchers interested in the topic, who were already working on relevant projects. Some

[3] `http://semanticweb.org/wiki/Semantic_Desktop#Dissemination_activities`

of the systems that are described below were introduced at these workshops: *IRIS*, *DeepaMehta*, *HyperSD*, *WonderDesk*. The workshops saw also the presentation of standalone semantic applications like the Semantic Clipboard [Reif et al., 2006], Beagle++ [Brunkhorst et al., 2006], Semantic Wikis [Oren, 2005, Aumueller and Auer, 2005], Semantic Instant Messaging (SAM [Franz and Staab, 2005], Nabu [Osterfeld et al., 2005]), semantic federation [Park, 2006], semantic search and ranking [Chirita et al., 2005], and trust [Noh, 2006].

3.2.2 Semantic Desktop Systems

In this section we present a review of existing Semantic Desktops. The NEPOMUK project is described separately in Section 3.2.3.

The Semantic Desktops presented here have the same common goals, and tackle by some extent the same challenges we described in Section 3.2.1. Some of the systems cover a broad spectrum of activities, aiming for a general solution, while others are focused on precise PIM tasks like email or task management.

We begin by describing the systems, along with their characteristics and features, and then continue by delving into the details of the architecture, services and ontologies for a subset of them. We start with the more relevant ones, which we describe in more detail. We then continue with other systems providing similar functionality. The list is ended by very specialised systems or applications, with semantic features.

Haystack

The Haystack system came out of MIT's Computer Science & Artificial Intelligence Laboratory (CSAIL) as early as 1997. It started as a personal system for Information Retrieval, where users manage their own private information [Adar et al., 1999], and the system adapts to the users' behaviour. Although not explicitly stated, from the beginning Haystack has been a semantic system, as it created and saved metadata links, or associations between things in the user's corpus, and allowed these links to be followed from one document to another. The connections are made either by inspecting the content of the documents, or by tracking the user's activity and inferring the connections. The user can also annotate content with metadata.

The architecture of the system evolved over time, but the overall structure remained the same [Huynh and Karger, 2002, Quan et al., 2003] — a three-tiered design. At the bottom there is a storage layer for the user's content — initially a database, later, an RDF store. On top of the storage, Haystack provided a basic data model for the objects, metadata and links; later it moved to RDF for describing the data. The RDF model is general, but it allows customisation by users [Karger and Jones, 2006]. The top level is made of client applications which can have one of several roles: (i) proxies for extracting information from existing unstructured sources like user applications or the Web; (ii) connectors for linking already extracted information; (iii) observers of user actions . There are also client services through which the user can interact directly with the data — modify it, search it or visualise it.

Stuff I've Seen (SIS)

The Stuff I've Seen (SIS) system is presented by [Dumais et al., 2003] as a solution to re-finding previously seen information, be it in an email, on a web page or in a document. The authors recognise the problem of data loss due to "the multiplicity of independent applications used to manage information each with its own organisational hierarchy" and "the limited search capabilities" they offer. SIS was not meant to be a new PIM application, but a unified point of access to existing content from other applications. It uses a central index and provides a common search interface over existing information sources. It aids re-finding of information by employing contextual cues in the search interface.

Developed at Microsoft, the system is based on the Microsoft Search indexing infrastructure, thus having access to functions of the operating system inaccessible to third-party applications. The architecture is modular. Five main components are chained to feed data extracted from various sources into the index. There is no mention of any semantic technologies being used, although some structured information is extracted: type of the resource, relations to other resources, like sender of an email and author of a document. The structures are then used to provide faceted query refinement. Some extraction of information from the content of the files is done with natural language processing. SIS also allows users to create metadata annotations on their content, which is an important first step towards creating semantic relations.

An extensive user evaluation of the system explored the best ways of presenting the search results, and determined which are the most common ways of searching for in-

formation. It found that time and people are important cues in re-finding, and thus should be included as part of the saved context. The importance of the time dimension is further studied in [Ringel et al., 2003], also based on the SIS system. Lifestreams [Freeman and Fertig, 1995, Fertig et al., 1996b, Freeman and Gelernter, 2007] proposed a chronological ordering of information and a totally reworked metaphor for the desktop, based on time as a storage model.

MyLifeBits

MyLifeBits [Gemmell et al., 2002, Gemmell et al., 2003, Bell, 2007] is another project by Microsoft Research, which uses the Memex as a blueprint for a digital personal store [Gemmell et al., 2006]. The team behind the CyberAll project [Bell, 2001], which then was continued by MyLifeBits, started an ambitious task: "to digitally store everything from ones life, including books, articles, personal financial records, memorabilia, email, written correspondence, photos (time, location taken), telephone calls, video, television programs, and web pages visited." [Bell et al., 2004]. The initial focus was on improving the methods for capture and storage, populating a personal ontology with digitised information from many sources. The following step was devising methods for using the information gathered — "tools [...] for annotation, collections, cluster analysis, facets for characterising the content, creation of timelines and stories" [Bell et al., 2004].

MyLifeBits does not impose a strict single hierarchy over the user's data, but uses links and annotations to organise the objects, and collections to group them. The annotations and links play an important role in the system. In the style of Nelson's transclusion, the links are bidirectional, and serve as well to establish ownership of connected pieces of data. Annotations play several roles in the system, and are considered important especially for non-textual resources like audio, video and images, as they enable functionalities on these resources, like text search, and story-telling, which otherwise are not possible. The system includes features aiming to make annotation as easy as possible — bulk annotation, predefined annotations, audio annotation. Collections are a special type of annotation. They are a way of supporting a loose hierarchy, without enforcing it. Overlapping collections are allowed, as well as resources which do not belong to any collection. "Fluid", or virtual collections are also supported, by saving and reusing a query.

Another important aspect of the system is the flexible visualisation it provides. A resource or a collection of resources can be presented in several ways, as different views

can give different insights into the data. "Furthermore, the visualisation must become a UI — the user will want to click on a row of a table or a peak in a graph and see the data behind it." [Gemmell et al., 2003]

MyLifeBits uses a predefined schema [Gemmell et al., 2006] to describe the data collected. The schema is flexible and customisable, although the authors do not anticipate that users will make many changes to it [Gemmell et al., 2003], as this would lead to complications in using the system. There are a set of predefined types, based on common resources available on a user's desktop. Instances of these types have unique identifiers which are used to establish the relationships. Relationship between two entities are bi-directional, and the two inverse links involved must have distinct names, e.g. "is organiser" and "is organised by".

The architecture of the system is modular, with the MyLifeBits store at the centre. The store [Gemmell et al., 2003] uses a Microsoft SQL Server database to provide the schema and store the data.

Gnowsis

Gnowsis [Sauermann, 2003] is one of the first implementations of a Semantic Desktop which advocated the use of Semantic Web technologies on the desktop, and the creation of a "personal Semantic Web for PIM" [Sauermann, 2009]. It proposes an architecture based on a local Web server as a desktop service, while integrated desktop applications communicate with it via Semantic Web protocols. In Gnowsis we encounter for the first time the notion of *desktop services* in relation to the Semantic Desktop.

Unlike Haystack, Gnowsis proposed that existing applications be modified to work with the semantic infrastructure, rather than being replaced completely. The Semantic Desktop would play the role of integration middleware by lifting semantic data from desktop formats and storing it in a central repository accessible to the applications. The extraction of data is done by adaptors, and the resulting network of semantic information can be accessed through a Web browser type of interface, which also provides access to the underlying model.

The semantic data is described with ontologies, and the system provides a generic personal information model (PIMO) [Sauermann et al., 2006], which is rich enough to cover most use cases, but also flexible and customisable by the users. PIMO was created with the user in mind, and to support personal mental models.

The Gnowsis Semantic Desktop largely influenced the architecture and basic design of NEPOMUK [Bernardi et al., 2008]. Its main role was that of an integration system [Sauermann, 2005a, Sauermann, 2005b] for knowledge management.

Integrate, Relate, Infer, Search (IRIS)

Another Semantic Desktop system which is all about integration is IRIS (which stands for "Integrate, Relate, Infer, Search") [Cheyer et al., 2005]. IRIS was part of SRI's [4] Cognitive Assistant that Learns and Organizes (CALO), as a personal information knowledge source [Ambite et al., 2006]. It also played the role of semantic user interface, through its embedded suite of applications.

CALO is an intelligent agent system which consists of several connected agents [Ambite et al., 2005] specialised on different areas of knowledge work — time management (PCalM [Berry et al., 2004], PTIME [Berry et al., 2005, Berry et al., 2011], PExA [Myers et al., 2007]), task management (Towel [Conley and Carpenter, 2007], PExA), contact management, and more. The AI agents are not necessarily semantic, although they do communicate with IRIS and use the semantic information it provides in their actions. CALO not only uses the knowledge, but also creates knowledge back into IRIS.

IRIS uses ontologies to describe data — initially a comprehensive ontology was defined, but it was replaced in favour of the Component Library Specification (CLib), the ontology used in CALO. The CLib ontology is modular just like the architecture of the system — different agents require different ontologies: there is an Office Ontology, a Meeting Ontology, a Documentation Ontology [Ambite et al., 2005].

CALO integrated parts of Personal Radar[5] into IRIS — the triple-store implementation (Semantic Object framework), and user interface elements. One of the key features of IRIS and CALO as a whole, is the focus on machine learning.

Semantic Explorer (SEMEX)

The Semantic Explorer (SEMEX) is another platform for semantic PIM. The system provides on-the-fly integration of personal and public data [Dong et al., 2004], by extending a user's personal information space and providing a logical view over it. SEMEX aligns

[4] `http://www.sri.com`
[5] `http://www.radarnetworks.com/`

the information integration task with the user's environment, making it happen as a side effect of the normal daily tasks.

The system comes with a basic domain ontology, which can be personalised by the users either by importing new models, or by manually changing it. The authors introduce the concept of malleable schemas [Dong and Halevy, 2005b], which can be extracted from browsing patters, or even suggested by the system based on clustering of resources. Using the ontology, SEMEX extracts semantic objects from desktop sources and stores them in a central repository [Dong and Halevy, 2005a]. Reference reconciliation [Dong et al., 2004, Dong et al., 2005] plays an important role in the system, particularly for the on-the-fly integration. It uses background knowledge and previous mappings for integrating external sources whose schemas might not match.

SEMEX offers an interface, similar to that of Haystack and Gnowsis, for querying and browsing by association the underlying database.

Cross-COntext Semantic Information Management (X-COSIM)

The Cross-COntext Semantic Information Management (X-COSIM) [Franz et al., 2007] is a framework which supports seamless PIM and information linkage across different contexts that users might find themselves in.

The system provides a reference ontology called X-COSIMO. Additional to defining concepts and relations, the ontology also describes the various possible contexts and relations between the concepts and contexts. The ontology is comprehensive, therefore to simplify its use, the system offers an application development interface called X-COSIMA, aimed at developers.

The semantic functionalities are integrated into existing applications through plugins. [Franz, 2008, Franz et al., 2009] describes and compares COSIMail and COSIFile with their non-semantically enhanced counterparts. Like other systems above, X-COSIM provides a browser for the semantic data it handles.

Multiple Ontology based Semantic DEsktop (MOSE)

The Multiple Ontology based Semantic DEsktop (MOSE) [Xiao and Cruz, 2005] is a multi-layered ontology framework for personal information management. The user manages the data through many *Personal Information Applications* (PIA) which are spe-

cialised on certain tasks, like trip planning or bibliography management. The PIAs are the main feature of MOSE [Xiao and Cruz, 2006]. They each have their own ontology to describe the domain knowledge, a user interface and a specific workflow. The PIAs can communicate and share data through mappings of their ontologies.

MOSE stores its data in several repositories, one for the file descriptors, one for the resources and one for tracking provenance of resources to the files they were extracted from. These repositories are populated by several services of the framework, and by the PIAs. The data can be browsed by association, modified and queried through the resource explorer, a browser-like interface. Other user interfaces to the data are provided by the PIAs, which themselves can be customised or created from scratch by the users.

PlacelessDocuments

Placeless Documents [Dourish et al., 2000] offers an alternative document management architecture, based on the flexible structure created using tags, or *document properties*, not the rigid hierarchy of folders. The system, created at Xerox PARC, is based on an earlier prototype called Presto [Dourish et al., 1999]. It is not explicitly a "semantic" framework, but it has many of the features we find in the more recent Semantic Desktops: information management based on metadata, and interconnected resources. Besides using properties to organise information from documents and emails, the Placeless system is distinctive through the use of *active properties*, which store executable code, and which provide services on the documents they are attached to. These "property-based document services are centred on the document and document activity, rather than on a separate application" thus solving the issue of application silos and allowing the user to focus on the task. Examples of such services include translation, summarisation, and format conversion.

The system supports generic system properties, as well as user-generated properties which enable a personalised view of the information space of the user. The properties are stored independently from the documents they describe, in a database.

Documents can be aggregated in collections based on their properties, they can be shared and collaborated on. The framework integrates the existing storage infrastructure available, through content providers connecting to the local file system, the web, or any network-accessible repository. Placeless Documents also integrates with existing applications which can only work with classic file systems, through the NFS protocol.

WonderDesk

WonderDesk[Zhang and Shen, 2005] is a distributed Semantic Desktop for resource management and sharing, part of WonderSpace eScience suite. It works with scientific objects, like papers, presentations, and other research artefacts. The system provides a basic vocabulary for metadata about the eScience resources and allows different domain specific vocabularies to be loaded for each scientific discipline. A separate component of WonderSpace, a P2P super-node named WonderServer, acts as an information integrator and indexer for a network of WonderDesk peers. The hybrid P2P architecture allows sharing of information between the nodes in a network, while still functioning standalone as a Semantic Desktop. It provides annotation functionalities, as well as sharing of annotations and resources within the group, and distributed querying.

HyperSD

HyperSD [Schwabe et al., 2005] is a Semantic Desktop browser which allows navigation and access to desktop resources based on metadata about them, in the style of Haystack. The application was developed with HyperDE [Nunes, 2005, Nunes and Schwabe, 2006], a Semantic Web application development environment. It provides wrappers for standard desktop resources like files, contacts, events, which extract the metadata and store it alongside the schema in an RDF store. The schema is simple, reusing some properties and relations from existing Web vocabularies like FOAF. The interface allows faceted browsing and contextual navigation, as well as creation and editing of new items or new associations.

OntoPIM

OntoPIM [Katifori et al., 2005] is a framework for Personal Information Management that relies on the use of a Personal Ontology. It is part of a bigger project for Task-centred Information Management (TIM), and is motivated by the same challenges as described above. OntoPIM supports storing any object of interest from the desktop, and relating it to any concept from the ontology. The Personal Ontology is just one part of the data layer of the system. There are two other ontologies, one for domain independent objects, like documents and emails; which is mapped to another one for domain specific objects; which in turn is mapped to the Personal Ontology. The users can build and modify their model through the Personal Ontology Builder and Personalisation Tool.

The system provides other services on top of the semantic data — instance matching, semantic save, querying, mapping.

The Autonomic Semantic Desktop

The Autonomic Semantic Desktop [Breitman and Truszkowski, 2006] introduces a semantic layer to a self-managing (or autonomic) application architecture. The result is an integrated desktop environment capable of self-managing behaviour, which uses semantics to achieve its goal of simplifying Personal Information Management, by supporting the user in maintaining their personal information in an automatic (or semi-automatic) way. The user data is extracted from two domains, the desktop and the Web [Breitman and Truszkowski, 2005], and is described by a shared ontology. The data layer is used as communication interface between a set of independent pluggable services.

Chandler

Chandler [OSA Foundation, 2012] is a project that defines itself as a "Note-to-Self Organizer", integrating personal information from multiple applications, and supporting task management and collaboration. It does not use typical Semantic Web technologies, instead it defines its own lightweight flexible vocabulary for describing the types of data, called *kinds*, as well as implementing its own data storage system. However, the ideas are consistent with those of the Semantic Desktop, integrating personal information from different sources and interconnecting it for associative browsing. The system provides a single modular user interface for the data, with different views for specific data types. Collaboration and data sharing in Chandler is realised with a client-server architecture, where the server is called a *hub*.

SeMoDesk

The SeMoDesk [Woerndl and Woehrl, 2008] project aims to bring the Semantic Desktop to the mobile environment. It tackles the limitations of mobile devices in regard of storage, display and input, while at the same time integrating, and taking advantage of the added functionality that the devices offer, like calls, text messaging, and location [Woerndl and Schulze, 2009]. The system uses the PIMO model defined Gnowsis and Nepomuk, for describing the data, which provides the necessary concepts for PIM. Infor-

mation is extracted from the applications available on the device, like the address book, calendar, email, and task management, and from files. The architecture is also similar to the one of Gnowsis, although modified to reuse services available on the mobile platform — the SQL database on Windows Mobile for storing the data. The user interface is adapted to the smaller screen size and the different interaction mode, by limiting the number of properties displayed based on the context of use [Woerndl and Hristov, 2009].

MindRaider

MindRaider [Martin Dvorak, 2012] is an open source "Semantic Web outliner" which aims to help organise a user's personal resources — documents, friends, thoughts — "in a way that enables quick navigation, concise representation and inferencing". The system is modelled as a mind-mapping and note-taking tool, but it allows interlinking of more types of concepts than just notes, and provides more visualisations than a mind map. MindRaider uses existing vocabularies like FOAF, SKOS and Dublin Core in combination with custom ontologies used for classification. The data is stored in a triple store. It enables interoperation with Gnowsis, through a connector, which allows querying data, and reusing resources.

DeepaMehta

DeepaMehta [Richter et al., 2005] is a service oriented application framework with a data model based on topic maps. It uses visualisations guided by research from cognitive psychology, and similar to concept maps. Its main goal is to provide a user interface which follows the associative way in which the human mind operates with information, thus keeping cognitive overhead low. The framework integrates information from different applications into one unified user interface, thus reducing the context switching imposed by using multiple applications for a single task using the same concepts. The data is described by an extensible schema, and users can create new types, relations and concepts based on a small set of predefined types. Data can be stored in several back-ends and exported and shared through SOAP. Information from various desktop applications is extracted by adapters, and can be interconnected with the Web or other remote information seamlessly.

DeepaMehta has a service oriented architecture. The main component is a Web server which communicates with the storage and the applications built on top of the framework.

The system offers a thin client, a Web application to access the data through a browser, and even a mobile interface.

Semantic 'LS'

Semantic 'LS' [Krishnan, 2008] is a PIM system that adds semantics to document management, to enable better organisation and sharing of information in small focused groups, through a P2P network. The architecture is layered, and the functionalities provided by the semantic layer include annotation — semi-automatic or manual, extraction of metadata from files, and grouping of files in virtual folders. It also provides query functions and easy to use visualisations. The data is described using two vocabularies: a Domain Knowledge Model (DKM) based on ArchVOC, which only handles subclass and superclass relations; and an annotation schema. Semantic 'LS' uses a database to store the metadata extracted, so that the semantic file system it creates does not modify the underlying file structure or information.

mSpace

mSpace [Smith et al., 2005] is a project that aims to support knowledge building in the style of the Memex, through associative links between documents. With the goal of enabling information exploration, the system provides a faceted interface to semantic data from multiple sources, and supports distributed queries through a centralised query service. It tackles the multidimensionality of the information by providing *slices* from the space, with context and the possibility of further browsing. The data is modelled with a lightweight ontology, which can be extended with other existing vocabularies, as required by the data of specific mSpaces.

Phlat

Phlat [Cutrell et al., 2006] is focused on providing an intuitive user interface for searching and browsing one's personal information, going beyond simple keyword search by using a user's intimate knowledge of the data. The system supports tagging of resources (files, emails and Web pages) by directly attaching the tag to the resource, not storing the relation in a separate location. This distinct feature has some benefits but it also has the limitation of only being able to tag things which support it (NTFS for files and

MAPI for email). Developed at Microsoft, like the Stuff I've Seen system, Phlat is based on the Microsoft Search architecture.

Other Specialised Systems and Applications

Numerous standalone applications apply Semantic Web technologies on the desktop, without providing a unified framework for the personal data, as a full-fledged Semantic Desktop does. There are also many other systems or applications that start from the same ideas as the Semantic Desktops, and work towards offering solutions to the same challenges of managing personal information, which organise data based on the semantic relations between entities, without explicitly using Semantic Web technologies.

In this section, we include some of them, which are too domain specific or task focused to be included in the category of Semantic Desktops. They span over a wide range of domains. The **Semantic Pen** [Varadarajan et al., 2005] system is a Semantic Desktop for pen devices, allowing for smarter note-taking. Since semantic note-taking is a category of special interest for us, Section 4.5 describes existing applications in this area.

Life-logging systems are not included in the above list, as they are not necessarily concerned with personal information management, but with logging and tracking activities and experiences, rather than information and knowledge. Some examples of life-logging applications include **Forget-me-not** [Lamming and Flynn, 1994] and **Save Everything** [Hull and Hart, 2001]. [d'Aquin et al., 2010, d'Aquin et al., 2011b] describe a different life-logging system, for monitoring a user's personal data exchange on the Web, while [d'Aquin et al., 2011a] presents a method for the semantic analysis of the activity data. The just-in-time information retrieval system **Remembrance Agent** [Rhodes and Starner, 1996] and **PurpleYogi**[6] are intelligent assistants using personal information to pro-actively help the user perform tasks. Forget-me-not and the Remembrance Agent belong to the wearable computing category of systems.

Ontooffice[7] brings integrated access from Microsoft Office application to semantic knowledge bases. **X-Explorer** [Wang et al., 2005] adds semantics to document management through use of metadata and content analysis. It provides a multidimensional interface, and associative browsing. **DocuWorld** [Einsfeld et al., 2005] also developed a

[6] `http://www.purpleyogi.com/`

[7] by `http://www.ontoprise.de` taken over by Semafora `http://www.semafora-systems.com/`

3D context-sensitive interface for the visualisation of a user's document space, based on metadata and relations between documents. Another visualisation paradigm is explored by **Lifestreams** and **TimeScape** systems which enable time-based organisation and presentation of information.

MIT's **Semantic File System** [Gifford et al., 1991] proposes a new type of organisation based on associations between files, to replace the typical tree-based hierarchy. It also describes extraction of metadata from files, to support making the associations.

neither folders, nor files. They may point to physical objects (stored somewhere) or they may represents only logical nodes in the resource tree. Examples of possible objects are files, folders, user groups, users or any other objects that may be represented in a tree structure. Apogee runs as a plug-in inside the eclipse IDE.

3.2.3 The NEPOMUK Social Semantic Desktop

We have seen so far a multitude of systems which use semantic technologies to enable better management of personal information. We continue by describing the NEPOMUK Semantic Desktop in more detail, as it is the framework we chose for the research presented in this thesis. In the section to follow, we sum up the systems in an analysis and a discussion of their characteristics.

Build on the ideas described in [Decker and Frank, 2004], the NEPOMUK project aims to bring together Semantic Web technologies applied on the desktop for better PIM, along with Peer-to-Peer technology and social networks for better collaboration and sharing of information.

NEPOMUK started as an EU research project, involving partners from academia and industry. It set out to define the blueprint [Bernardi et al., 2008] for a generic Semantic Desktop, based on previous research as well as new studies. Many of the systems presented above were surveyed, and many of the pioneers of the Semantic Desktop research were involved. The partners contributed knowledge and existing useful components.

The report on the NEPOMUK final architecture [Reif et al., 2008] describes how the Semantic Web technologies are applied on local scale — "to integrate information between applications such as email, contacts, calendars, or file-manager", and on the global scale — "users socially interact by sharing resources, by communicating, and by collaborating over the network". The resulting Semantic Desktop brings improvements in

two distinct but intertwined directions: enhanced Personal Information Management through better interlinking of information across application boundaries, as well as improved information sharing and exchange across social and organisational boundaries.

The Blueprint of a Semantic Desktop

The main goal of NEPOMUK was to define a well-thought blueprint for the Semantic Desktop, which was to provide a template for frameworks to follow, and generic solutions to the design problems which arose. The blueprint evolved based on requirements and the final version is described in [Reif et al., 2008]. It was continuously used to provide a prototype reference implementation of the framework. There were also prototypes for special community scenarios, and usage studies done on them, which fed back into the blueprint design.

The blueprint of the Social Semantic Desktop as envisioned by NEPOMUK is defined on two dimensions:

1. the way the mental model of the user is represented by the framework, and
2. the services provided by the system to the users.

The first item in the list — the representation of the mental model — is described in more detail in the specification, as it is a crucial point enabling communication and data sharing across any future Semantic Desktop which follows the blueprint. For the second item, the blueprint does not provide recommendations on the exact architecture of the system, leaving this issue to the implementers to decide. General guidelines are given on the basic services required, like the storage service for example.

Describing the Mental Model — the NEPOMUK Ontologies

The NEPOMUK Semantic Desktop defines and uses a set of ontologies[8], known as "the NEPOMUK ontologies", or "the desktop ontologies", which are complemented by ontologies defined by the community, like Xesam[9]. They describe as completely as possible the way knowledge representation is done in the system, from the very high level concepts to the most detailed low level ones. Figure 3.1 [Reif et al., 2008] shows

[8] `http://oscaf.sourceforge.net/`, previously found at `http://www.semanticdesktop.org/ontologies/`

[9] `http://xesam.org/main/XesamOntology` - is used in Nepomuk-KDE

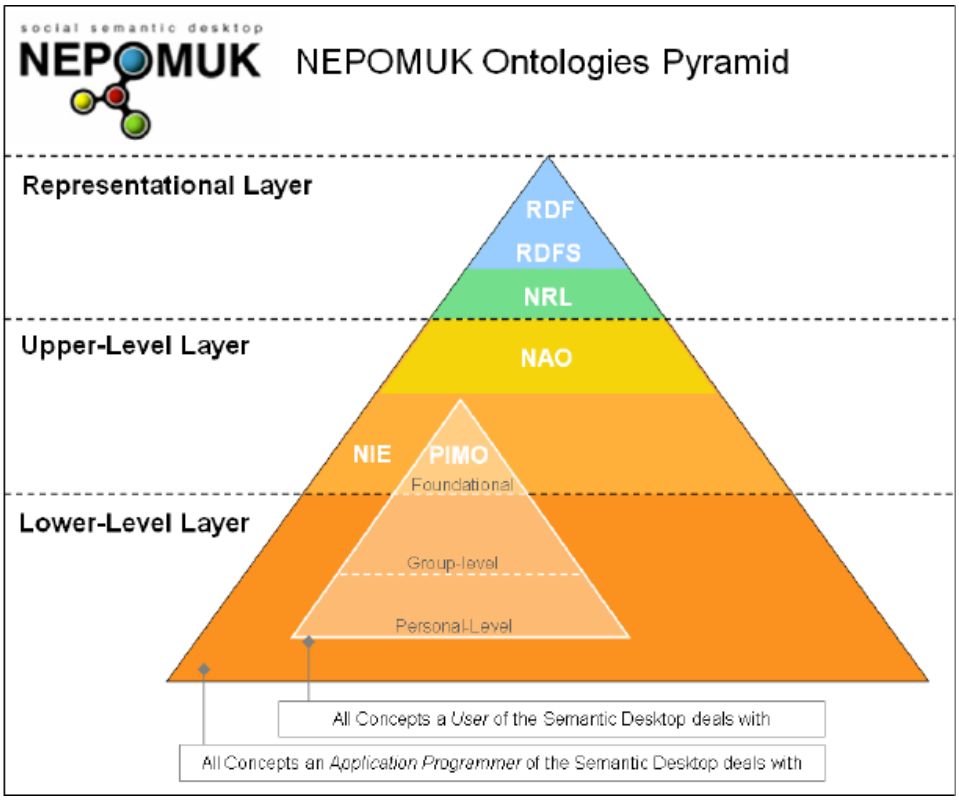

Figure 3.1: The pyramid of desktop ontologies.

how the ontologies build and depend on each other. The pyramid is divided into three levels, the top two levels containing ontologies which are more stable, and provided as part of the framework, while the bottom layer ontologies are customisable by users.

To accommodate some restrictions specific to describing desktop and personal data, an extension to RDF was developed. It is called NEPOMUK Representational Language (NRL)[10] [Sintek et al., 2007], and is a representational ontology, thus belonging in the top level of the pyramid. It adds Named Graphs and Graph Views to RDF/S and introduces the closed world assumption to the data. The named graphs in NRL are similar to the named graphs defined by [Carroll et al., 2005] except that they do not follow the open world assumption. NRL defines graph roles for named graphs, where roles contain information about a graph's data and how it should be handled.

The open world assumption which is usually used on the Web, states that everything that is not known is undefined, which makes sense when working with very large amounts of information, most of it unknown. It does not work as well on the desktop, where we

[10] `http://oscaf.sourceforge.net/nrl.html`, previously at `http://www.semanticdesktop.org/ontologies/nrl`

handle personal data that is implicitly known to the user in its entirety. That is why NRL introduces the closed world assumption, which states that everything that is unknown is false. However, NRL does not make any assumptions on the semantics of a graph defined with it. With the use of graph views over named graphs, there can possibly exist different views with different semantics over the same graph.

The upper level ontologies describe basic concepts generalising over multiple domains and activities specific to the desktop and PIM. There are three ontologies in this level:

NEPOMUK Annotation Ontology (NAO[11]) contains concepts which allow users to annotate desktop resources, including custom descriptions, identifiers, tags and ratings. Generic relationships between related resources can be made explicit through properties defined in this ontology.

NEPOMUK Information Element Ontology (NIE[12]) describes a unified vocabulary for native resources available on the desktop. NIE is in fact a larger framework, where the core part is the NIE ontology, complemented by several smaller vocabularies describing specific types of desktop resources, like files, music, emails, address book contacts, calendar entries, etc. Standards for representating many of these types already existed, either in the form of RFCs or in the form of Web vocabularies. In these cases, the existing resources were used as basis for the respective NEPOMUK ontologies. For example, the NEPOMUK Contact Ontology (NCO[13]) was designed based on the VCARD specification (RFC 2426 [Dawson and Howes, 1998]), and on the Vcard ontology[14], but it has a much broader scope than either of the two. Similarly, the NEPOMUK Calendar Ontology (NCAL[15]) is an extended adaptation of the W3C calendaring ontology[16]. NIE defines two disjunct classes of resources, `DataObject`s — the physical representation, and `InformationElement`s — the interpretation and content of resources. The two are then subclassed in each specific vocabulary of the framework. The NIE classes are designed for machine use in extracting semantic information from existing desktop sources and applications, not for direct handling by the users.

[11] `http://oscaf.sourceforge.net/nao.html`, previously at `http://www.semanticdesktop.org/ontologies/nao/`

[12] `http://oscaf.sourceforge.net/nie.html`, previously at `http://www.semanticdesktop.org/ontologies/nie/`

[13] `http://oscaf.sourceforge.net/nco.html`, previously at `http://www.semanticdesktop.org/ontologies/nco/`

[14] `http://www.w3.org/TR/vcard-rdf/`

[15] `http://oscaf.sourceforge.net/ncal.html`, previously at `http://www.semanticdesktop.org/ontologies/ncal/`

[16] `http://www.w3.org/2002/12/cal/`

Personal Information Model Ontology (PIMO[17]) is both a upper level ontology and a lower level ontology, as it contains both generic concepts and quite specific ones. From a user's point of view, PIMO is the central ontology of the Semantic Desktop. Its scope is to model the data that the user works with. According to its specification [Sauermann et al., 2009], "PIMO is based on the idea that users have a mental model to categorize their environment", and "each concept in the environment ... is represented as [a] Thing in the model". PIMO defines high level types like Person, Project, Event and Task, which reflect the user's mental image of the objects, not their representation in various desktop applications, nor the files used to store the information about them. That is the task of the NIE ontology and framework, while the PIMO is needed to provide an aggregated view of all the possible sources of information about a *thing* the user needs.

The lower level ontologies consist of domain ontologies and application ontologies — which have very specific or limited use cases. Users can import new ontologies into their Semantic Desktop either directly or through applications. When certain ontologies become widely used, it is recommended that they go through a process of standardisation and are included in the desktop ontologies, in order to support reuse.

Existing ontologies can be customised by users or by application developers, this is however discouraged for the upper level ontologies, as it would affect interoperability between Semantic Desktops.

The NEPOMUK Reference Implementation

The reference implementation of NEPOMUK is based on the blueprint defined within the project, but it is not an exact realisation of it. The general structure is maintained however, and several of the possible services are provided. The implementation is done in Java, for portability.

The service oriented architecture of the Java implementation of NEPOMUK uses SOAP and OSGi for communications between the services, and for discovery. The component which provides the registry and discovery functionality is called the NepomukMiddleware (or Middleware), and it is a service itself. The implementation evolved

[17] `http://oscaf.sourceforge.net/pimo.html`, previously at `http://www.semanticdesktop.org/ontologies/pimo/`

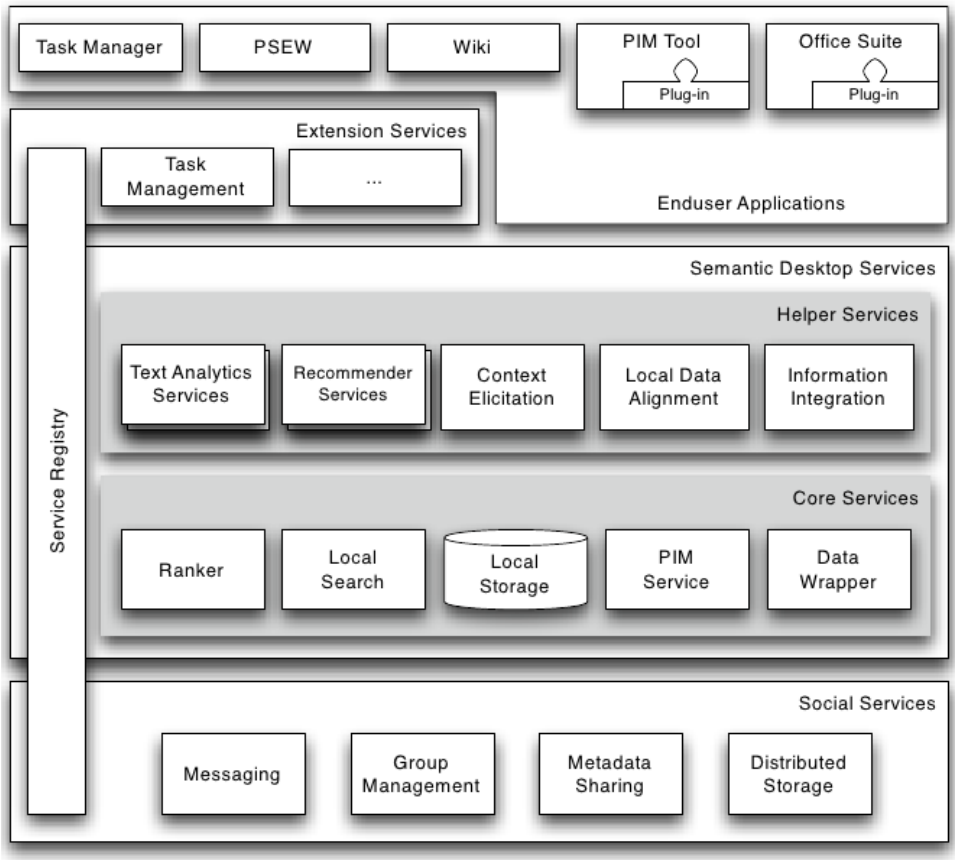

Figure 3.2: The architecture of the NEPOMUK Java implementation.

from a local web server with SOAP, to Eclipse, as it has better support for inter-service communication.

Figure 3.2 [Reif et al., 2008] shows the services composing the reference implementation. The core services provide functionalities for extraction, storage, and retrieval of the semantic desktop data. They are fundamental for the functionality of the Semantic Desktop, as they are used by all the other helper services and applications built on top of the framework.

The storage service is the central service of the Semantic Desktop. All the semantic data is stored and retrieved from here, thus acting as a blackboard (see Section 3.2.4). The storage service uses existing triple store implementations, current version using Sesame2. Additionally it provides some basic inference and query support. Lucene is used for indexing.

The DataWrapper service extracts information from desktop sources and stores it in semantic form in the central RDF store. It uses plugins to access and extract data from

application specific formats and repositories. Other helper services can be combined with the DataWrapper, to ensure that the data is well integrated and that duplicates are removed (e.g. an integration service), a data alignment service, or an inference service to extract new information based on the existing triples.

The PIM service provides convenience methods to access and create information in a user's PIMO.

The local search service provides search functionality over the local repository, supporting both structured queries and full text search. It includes the functionality provided by a Ranker service, which computes resource relevance for better search results.

Additional helper services were developed: a Context service, Recommender services, Text Analytics services. On top of the core services and the helper services, several extension services were developed, to showcase possible uses and to demonstrate how the services can be combined. On top of the services, applications were built to provide functionality to the users in a friendly user interface. The interface to this implementation of the NEPOMUK Semantic Desktop is called PSEW (Personal Knowledge Workbench) [Grimnes et al., 2009]. It offers a visual way of interacting with the services, as well as several views of the data contained in the local repository. Other specialised applications were developed: a semantic email extension [Scerri et al., 2009] to Microsoft Outlook and to Mozilla Thunderbird, a plugin for Microsoft Word [Groza et al., 2011] to semantically annotate documents, etc.

The social part of the Social Semantic Desktop was meant to be fully distributed, thus a P2P network implementation was used for the majority of the services. However, a centralised NepomukHub was developed as well, using XMPP. The NepomukHub acts as a server for inter [Semantic] Desktop communication, passing messages between NEPOMUK instances, and also as a shared triple store, where social information, like group memberships, is stored and managed. The messages sent through the NepomukHub transport RDF graphs, thus any semantic information can be shared among desktops. Several social services were developed on top of the P2P network and the NepomukHub infrastructure. They include a Messaging service, a Metadata Sharing service, a Distributed Storage service and Distributed Search. Local services can be extended to use the social services.

Nepomuk-KDE

By far the most successful part of the NEPOMUK project was the "Mandriva Community scenario" [Lauriere et al., 2006] or Nepomuk-KDE[18] as it became known. It was initially meant as a proof of concept, showing that the blueprint can be realised outside of the reference implementation. The start of the NEPOMUK project coincided roughly with the beginning of development on a new major version of the KDE[19] Desktop Environment for Linux — KDE4. Thus it provided the opportunity of including the new research ideas into an emerging platform, more deeply than it could have been done for a system that was already completely designed and implemented. As a result, Nepomuk-KDE is part of the core libraries of KDE and is used in several central applications, the best example being Dolphin, the default file manager. Desktop search is also done through NEPOMUK, and metadata creation, such as tagging, rating and commenting, is available desktop-wide. The adoption of NEPOMUK and semantic technologies has been very good, and development on the framework continued after the end of the EU project, driven by the community that was formed.

Including NEPOMUK in KDE also meant convincing the developers of KDE applications to make their software use the new semantic features. It required familiarising them with the semantic technologies used, and encouraging them to participate in the development of the framework and to collaborate on the ontologies.

For open source projects like KDE, there are restrictions on what libraries can be reused due to software licences. This had an effect in the first stages, on the fact that the Java code of the NEPOMUK reference implementation could not be directly reused. Although it is possible to run Java on Linux, there are restrictions on the distribution of Java libraries. That was one of the reasons why Qt, the language in which KDE is developed, was preferred for Nepomuk-KDE. Although it is possible to use Java libraries in applications, code written in Qt/C++ was preferred by the packagers of the distributions that used KDE4. Thus, despite inferior results and features, many Nepomuk-KDE Semantic Desktops used the Redland[20] C library for the storage service, while at the same time Sesame2 library was available but since it was written in Java, it required additional configuration. Currently Nepomuk-KDE uses a customised trimmed down

[18] `http://nepomuk.kde.org`

[19] KDE has been rebranded in the mean time to refer to the community instead of the software produced. Thus KDE is no longer an acronym for "K Desktop Environment".

[20] `http://librdf.org/`

version of OpenLink's Virtuoso[21]. It is feature-rich, stable, and the close collaboration with the OpenLink developers resulted in a version tailored specifically for running a triple store on a desktop.

Architecture-wise Nepomuk-KDE differs from the blueprint. For simplicity the social part has been ignored in the start, and the focus was on providing easy and simple to use semantic features to attract more contributors before diving into more complex tasks. Several social aspects are being implemented now with the use of NEPOMUK in the Telepathy[22] project.

The services in Nepomuk-KDE are also different. The preferred inter-service communication method is either through DBus[23] or API calls. The services themselves are not defined as Web services running on the desktop, but as KDE modules. They are managed through a NepomukServer service. As already mentioned, the current implementation of the Storage service uses Virtuoso as the default triple store, although other backends can be configured for use. The Data Wrapper function is done by Strigi[24], but instead of independent plugins for external applications, Strigi is a plugin-based system, with each plugin being responsible for certain types of files. Other exporters of semantic data come from Akonadi[25], the PIM framework of KDE.

The Nepomuk-KDE framework is changing continuously, due to the active community around it. It is evolving together with KDE and because of it, the uptake has been considerable, with many developers adding semantic features to their applications, and thus increasing the visibility and attracting more users.

3.2.4 Characteristics of Semantic Desktops

Architecture

Most of the systems described in Section 3.2.2, along with NEPOMUK, agree on the need for a layered architecture for Semantic Desktops. The layered general architecture shares many common points with the conceptual architecture for applications on the Web of Data, as surveyed in [Heitmann et al., 2012]. It can be divided into three major

[21] http://virtuoso.openlinksw.com/
[22] http://telepathy.freedesktop.org/
[23] http://www.freedesktop.org/wiki/Software/dbus
[24] http://strigi.sourceforge.net/
[25] http://community.kde.org/KDE_PIM/Akonadi

layers which build on top of each other, with dependencies and even overlaps between them.

Data layer The Semantic Desktop revolves around the [semantic] data that it contains. As such, the architecture is also centred around handling this data. This layer contains the vocabularies for describing the data, and the data itself.

Service layer Based on the data, the Semantic Desktop provides an enabling framework of basic services, which can be either visible or transparent to users. This framework enables the functionalities of the applications. The set of basic services vary among the systems described, but are indispensable to them. These services are central to the desktop, and generally accessible from all applications. Some of the basic services include storage, extraction, integration, query, inference, annotation.

Storage service can vary from a database system like MS SQL server used by MyLifeBits, to a fully fledged RDF store like Sesame or Virtuoso in Gnowsis and NEPOMUK. Many systems use Jena (SEMEX, IRIS) and some use files. Some use a combination of storage from the options above (MOSE has three — database, file and Jena RDF store). Depending on the type of storage, semantic resources are identified either by URIs or by unique database IDs.

Extraction service can come under several names — crawlers, wrappers, gatherers — however, they provide the same function, that of extracting semantic information from non-semantic sources, whether structured or unstructured. It can vary from simple extractors of metadata already available for files — like photos (EXIF) and music (ID3), or emails — sender, subject, to parsing and analysing unstructured text to extract information from multiple file formats (Stuff I've Seen, SEMEX, Gnowsis, NEPOMUK). CALO features natural language analysis to extract entities and relations among them. MOSE also extracts resources and relations from files, and stores two types of information — R-F links, which specify which resource was extracted from which file, and R-R links, which specify connections between resources. Semantic information extraction plays a significant role in providing the basic functions of a Semantic Desktop, and all the systems described here implement it. The extraction service generally comes bundled together with an instance matching service.

Integration service (instance matching, entity matching, or de-duplication) has the role of checking if two instances are the same. The definition of *the same* varies from system to system. One of the main uses of this service is complementing the functionality of the extraction service, by checking if a newly

extracted entity already exists, in which case, depending on the policies the existing entity is reused instead of creating a copy, or a new entity is created and linked to the existing one.

Query service All the systems allow querying their semantic data. Keyword search is supported by IRIS, Semex, Haystack, and MOSE. For keyword search, an **indexing service** is used, which can be an independent service, or part of the storage or the extraction service. Structured query languages like RDQL and SPARQL are also supported in MOSE, IRIS, Gnowsis, X-COSIM, and NEPOMUK. The ways in which the results are displayed are discussed in the presentation layer.

Inference service Inference on the semantic data is supported by IRIS, Gnowsis and NEPOMUK. This service can be standalone or included in the extraction or the integration service. The quality of the inference results depends on the engine used and the ontologies used to describe the data.

Annotation service All systems allow some type of annotation of resources, however, not always in the form of a standalone service. Annotation refers to creation of metadata for resources, or of new connections between resources. Manual creation of new resources can also be considered annotation. Some automatic annotation is performed by the extraction and the integration services. In Haystack, where there is one access point to the data, the annotation service is in fact a user application, as it happens directly. In MyLifeBits annotations play a central part – the system allows sharing, annotation of annotations. The most basic types of annotations are tagging and grouping in collections, which are both used for categorisation.

Presentation / Application layer The user-facing interface of the Semantic Desktop makes up the presentation layer, built on top of the supporting framework. The systems have a large variety of user interfaces, providing functionality that varies from simple resource browsers, to complex PIM tools like email clients and task managers.

Regarding the applications they provide, the systems are divided in two categories, depending on whether they choose to enhance existing applications with semantic capabilities, or propose new semantically enabled applications to replace existing ones for PIM tasks. X-COSIM, Gnowsis and NEPOMUK belong to the first category, while IRIS and MOSE belong to the second.

The flexible and customisable visualisation of information is one of the distinguishing features of Haystack. It is a Semantic Web browser [Quan and Karger, 2004], providing a unified interface for the data it contains, with the added functionality of allowing edits [Quan et al., 2003] and customisations. The feature that sets Haystack apart from other semantic browsers is its dynamic creation of user interfaces. This is realised by recursively rendering semantic resources [Huynh et al., 2003, Karger et al., 2005]. The way an object should be rendered is described in RDF. General visualisations are provided out of the box, but users can customise them according to their needs.

Most systems provide a resource browser and search interface in the style of Haystack, although usually not as flexible. Some browsers display the underlying ontology. They allow changes to be made to the data directly through this interface. Faceted browsing and browsing by association are available in all resource browsers.

MyLifeBits and SEMEX propose that multiple visualisations be supported for resources, depending on the user's context.

Blackboard and fuzzy layers

The layers of the architecture can overlap. The storage service is at the fuzzy border between the data layer and the service layer. It is responsible for the persistent storage of the data and for providing low level access to it. Since the storage service is highly connected with the data, it can be seen as part of the data layer, but at the same time it is a foundational service, and as such it is part of the service layer.

Similarly, services which provide any type of user interfaces are at the border between the service layer and the presentation layer.

Inside the service layer itself we can see a separation based on the level at which the services operate. Some foundation services are more oriented towards the data - like an inference service or the analyser and extractor services. Other services provide more user-oriented functionality, like an annotation service, or a query service.

Services can use functionality provided by other services, communicating and building on top of each other. The communication between the services, as well as the communication between the services and applications can take several forms: through programming interfaces (APIs provided by the framework), Web standards (Gnowsis promotes a Web server as a desktop service), or P2P communication (MOSE, NEPOMUK).

The architecture of many of the systems uses the Blackboard pattern, where the blackboard is the data storage, or even the entire data layer, which is accessible to all services and applications. They have the role of the specialists in the pattern. The control element is the storage service. Specialists populate the blackboard with data which can then be accessed and refined by other specialists. For example, a PDF extractor service can parse documents and extract titles, authors and affiliations. The data must then be processed by the integration services, so that duplicate authors and affiliations can be merged into a single unique representation. Furthermore, the inference service can then extract co-working relations between the people, based on common affiliations.

Data Representation

Semantic data is the most important part of the frameworks, and the way it is described influences the quality and quantity of things that can be done with it.

All the systems define a data model for the data they use and extract. The data models vary from very small and generic, like the one in SEMEX, which has a restricted number of classes and associations, to the comprehensive one provided by X-COSIM.

Being small, SEMEX's ontology is unitary, but the bigger ones usually are modular. MOSE's ontology is divided in small application ontologies belonging to the PIAs and domain ontologies used by the services. CALO also has different ontologies for specific domains and used by specialised agents. X-COSIMO is divided into modules representing different aspects of the data. More than being modular, the set of ontologies used by NEPOMUK is also layered. The representational level defines the vocabulary used to define the other ontologies. The upper-level ontologies contain domain-independent abstract knowledge. The lower level ontologies contain specialised concrete terms from the user's mental model. The low level PIMO plays an important role in both Gnowsis and NEPOMUK, as it is used to describe the data most often handled by the user.

Most of the frameworks use RDF or OWL ontologies to describe the data. Only MyLifeBits and SIS do not mention the use of ontologies, MyLifeBits using a flexible database schema to define the data model.

Another differentiating characteristic is whether or not the data model can be personalised by the users by creating new types and new relations. X-COSIM does not allow the modification of the domain ontology, but it does allow the creation of custom mappings from and to other external ontologies. Haystack and SEMEX on the other

hand argue for the need for personalisation of the ontologies by the user, as only in this way, can a truly personal mental model be created. Most systems do support the customisation of the underlying model, as well as the import of new vocabularies in the system, by linking to them or through mappings. This fact raises the challenge of reconciling data described with customised models.

The long-term study by [Sauermann, 2008] found that very detailed ontologies are not necessarily more useful for the users, as simpler and more generic relations are preferred over more specific ones. The intuition is that the more precise properties, although better at classification, present the added challenge of choosing the suitable one. The same study of using Gnowsis has shown that although customisation of the underlying ontologies is supported, the users only make minimal and rare changes. This result confirms the MyLifeBits hypothesis that although their schema is flexible and can be modified, users will probably not make any changes to it. While a layered system like NEPOMUK's gives more flexibility, allowing users to change and update the ontologies gives too much flexibility for the normal use cases, and would be used only by a few power users, in a very limited number of occasions. Most customisations were sub-classing existing classes and specialising existing properties by creating sub-properties.

3.2.5 Discussion

In this section we covered some common aspects of the systems regarding their architecture, the functionalities they provide and the way they represent and work with data. Following again the layered structure used before, we continue with a discussion of some of the shortcomings and possible developments which appear to affect all or most of the Semantic Desktops.

At the data level, as the systems have evolved, the ontologies they employ have become more complex. While providing good coverage of the PIM domain, comprehensive vocabularies with detailed relationships among types showed that the simpler, more general relations are used much more often, regardless of the possible loss of meaning [Sauermann, 2008]. Hence, rich ontologies are not necessarily better, since most users prefer simple relations between resources, which are enough to remind them of the connection, without specifying it fully. Detailed vocabularies might prove more useful in the case of automatic processing of data, but using them to manually annotate is costly.

Moving into the service layer, the storage and the indexing services provided by the systems use semantic technologies which have evolved at a rapid pace. A slow repository, and thus slow response times for queries and browsing have at least partially been at fault for the poor uptake of the Semantic Desktops. Fast and memory efficient triple stores are now available.

Regarding the applications provided by the systems, we observed the distinction between integrating semantics into existing applications versus creating new semantic applications. Forcing the users to switch from the applications they know to avail of the power of the Semantic Desktop in applications they would need to learn how to use, has not proved to be a successful strategy. However, systems like Gnowsis, X-COSIM and NEPOMUK use plugins to add semantic functionality to existing popular tools, thus letting users continue to use their preferred tools while benefiting from the added semantics.

One of the reasons for the slow uptake of the Semantic Desktop could be the cold start problem, which is observable despite multiple automatic ways of extracting, linking and importing resources, and kick-starting the system. This could prove that the user's manual annotations are more important than the automatically extracted data, which has its own important role though. However, since the automated data comes from sources which are accessible to the user anyway through conventional tools, there is no immediate incentive to use the semantic applications, which translates in little manual information being added into the system, and the benefits delayed further, despite several evaluations [Franz, 2008, Sauermann, 2008] proving that Semantic Desktops *are* better for PIM tasks [Franz et al., 2009].

It could also be that the visualisations used for the automatically extracted data are not suitable for the purpose, or not attractive enough. Generic graph or table visualisations are not appealing, and treating every resource the same is not an effective way of conveying information.

In recent years two developments occurred and influenced the direction in which the Semantic Desktops evolve: (i) the exponential growth of semantic data available online, mostly due to the Linked Data initiative and the Linking Open Data project which have a large success, and (ii) the growing concern about the privacy and security of personal data. Some of the information available as Linked Data might be relevant to the users of Semantic Desktops, so using it in applications and services to add value to the users is a low hanging fruit, -waiting to be picked. However, the open aspect of most

of the available data causes concern especially when it becomes mixed with valuable private personal information. Privacy is not the only concern, albeit a very important one. Establishing whether the Web information is trustworthy is another concern, and possibly a harder one to tackle.

3.3 Conclusion

This chapter covered Semantic Desktop systems and applications described in literature, starting from the historical ones like the Memex, NLS and Xanadu, to recent ones like NEPOMUK. We showed that they share common goals and characteristics, and we extracted and discussed their general architecture and data representation means. Understanding these is very important for the core part of the thesis, as our work is grounded and builds on top of the framework provided by the Semantic Desktop, specifically, the Nepomuk-KDE Semantic Desktop.

In recent years other systems have emerged, targeting the issues described above, on the desktop or other devices, by using semantic technologies. However, the focus has changed from exploring possible architectures and creating vocabularies, to a more data centric approach. The Semantic Desktop has matured, along with the semantic technologies it employs, so that now new and more exciting, as well as harder, problems arise. Now that the infrastructure has been put in place, the Semantic Desktop awaits the killer app which would bring it and the possibilities it opens into the public eye. Siri[26] and Evi[27], as well as IBM's Watson[28] have led the way.

In the next part, we continue by presenting the core research of the thesis, which builds on top of the base set by the Semantic Desktop, to better interlink personal information, not only within the desktop, but also to the Web.

[26] http://www.apple.com/iphone/features/siri.html
[27] http://www.evi.com
[28] http://www-03.ibm.com/innovation/us/watson/

Part III

Core

In the core part of the thesis we present o-ur main contribution, consisting of research and applications towards better interlinking of information on the Semantic Desktop. We introduce interlinking of personal notes with the semantic note-taking tool SemNotes, then we present a method for connecting Semantic Desktop data to the Web of Data, followed by the description of a use case and system for semantic publishing which incorporates and builds on the two previous chapters. Each of the chapters is based on previously published works.

We start from the premise of the Semantic Desktop — it provides the framework we need for building our network of linked personal information. The order of the chapters follows the logical order of interconnecting Semantic Desktop information.

We start at a local level in Chapter 4, by describing ways of creating new semantic data within the desktop, integrating it with the existing information from the desktop, and making it accessible across application borders. In this context, we present the challenges we found and our solutions to semantic application development on the Semantic Desktop.

Since the personal information we use is no longer restricted to the desktop, it has become important to expand the network of personal linked data beyond the desktop, into the Web of Data. Making use of the large amount of data available online in linked form, in Chapter 5 we describe a method for bridging the gap between the two linked data networks — the Semantic Desktop and the Web of Data.

In Chapter 6 we merge the interlinking of new notes with existing semantic desktop data from Chapter 4 with the interlinking of the same data with Semantic Web from Chapter 5, for the purpose of semantic publishing. The solution we present is not limited to the use case of semantic blogging, but it is valid also for e-health use cases like patient records and doctor notes, or for the enterprise domain, for reporting or collaboration.

In the last chapter of the section, Chapter 7, we describe several other proof of concept applications we developed with the same goal of enabling and supporting better interlinking of personal data on the desktop and beyond.

The work builds on top of the Nepomuk-KDE branch of the NEPOMUK Social Semantic Desktop, described in detail in Section 3.2.3 and it reuses the ontologies defined within the NEPOMUK project. Using the libraries and vocabularies in an application has led, as a collateral result, to improvements and changes in both.

Chapter 4

Creating and Interlinking Semantic Data on the Desktop with SemNotes

Based on "The Semantic Desktop at Work: Interlinking Notes" [Dragan et al., 2011b] published at the 7th International Conference on Semantic Systems (I-SEMANTICS 2011)

The Semantic Desktop has been proposed as a solution to the ever growing problem of information overload on our computers. It provides the foundations necessary to integrate and manage personal information. However, the challenge of designing and realising semantic applications that use this infrastructure still remains. In this chapter, we present SemNotes[1], a semantic note-taking tool for the Nepomuk-KDE Semantic Desktop, as a tool for creating new semantic information on the desktop and integrating it seamlessly in the existing network of linked desktop data. SemNotes provides a real-world, functional use case for fully exploiting the capabilities of the Semantic Desktop: interlinking, organisation and management of personal information, improved search and browsing. Furthermore, it represents our solution to a set of identified generic challenges for applications on the Semantic Desktop, as described by the first research question in Section 1.2. We describe a task-based user evaluation comparing SemNotes with a conventional note-taking tool. The results show that complex searches on interlinked information created with SemNotes are significantly faster, with little or no extra effort required from the users.

[1] `http://smile.deri.ie/projects/semn`

4.1 Introduction

The Semantic Desktop provides a framework for creating applications and tools that simplify the daunting tasks of managing personal information accumulated on the desktop. The information overload problem, combined with the disconnection of data caused by application silos, is solved by the use of Semantic Web standards for storing and interlinking the desktop information. Data which before was stored separately by different applications can be now explicitly connected. The result is a network of interlinked personal information, which can be browsed by association, filtered and searched in a unified way.

The Semantic Desktop overcomes the limitations of conventional desktops, where information is kept in different formats and application repositories, by using common vocabularies to describe the data, and a desktop-central place to store it, in a standardised format, accessible to all applications.

Indeed, the Semantic Desktop makes the information load manageable. However, new challenges emerge: one such new issue is how to design and realise semantic applications that use the infrastructure provided by the Semantic Desktop. We address this problem by dividing it into smaller, simpler challenges and providing solutions for each of them. To illustrate the solutions, we describe the design of a semantic note-taking application for the Nepomuk-KDE Semantic Desktop, called SemNotes. We use note-taking as an example because it is a desktop activity that is not limited to a specific domain, since the notes can widely vary in topic. It is also a common activity that plays an important role in Personal Information Management and that we believe would benefit from the use of semantic technologies.

4.1.1 Challenges

In Section 1.2 we broke down the first research question **Q 1.**: *How to build semantic applications and tools for the Semantic Desktop as to provide the best experience for the users, while creating reusable semantic data?*, into several sub-questions. These are the challenges we found in designing new semantic applications for the Semantic Desktop:

Q 1.1. *How to create semantic data that is correct, and maximises the reuse of existing Semantic Desktop data through interlinking?*

Applications should be aware of the data that is available on the desktop, from

other applications. The data they produce is also accessible to other applications, and this should also be taken into account, because it raises the dual challenge of making sure that the data is represented correctly so that other applications can use it if they choose, as well as making the most of existing information through reuse and link creation. How to interlink information items is the most important challenge for adhering to the Semantic Desktop requirements of creating and maintaining a network of linked desktop data. It refers in the first place to the new information coming into the desktop through the application, and how to integrate it with the linked information that exists on the desktop. But it also covers the situation when the only new information created is in fact a new link between existing entities. Existing desktop data is heterogeneous. The reasons for this include the fact that different parts of it are created by different applications, with different functionalities and needs. So, although represented in a standardised way, the data is not always consistent, up-to-date or even correct. Furthermore, the data is also heterogeneous in terms of form: there are new types of information available to be integrated (i.e. tags, relations). Best practices advise the reuse of as much of the existing information as possible, because through reuse, the interlinking becomes deeper and richer. However, the possibility of reusing vast amounts of existing data raises other challenges, in selecting the right amount of necessary information for maximum benefit for the users, while not overloading them. The selection is complemented by the way the information is shown.

Q 1.2. *How to design the human-computer interaction in an application for the Semantic Desktop?*

This question relates to designing interfaces which support the existing workflow of the user, while integrating the additional semantic information in a useful way. A balance must be found between too much information, so that it interferes with the user doing tasks, and too little, or not enough to make a difference. The question also refers to more than just displaying the information in an interface — allowing users to interlink information items easily, and generally aiding the creation of new semantic data is also a challenge for application development. Extra difficulty is added by the variety of interlinked information available on the desktop, combined with the reduced control of developers over what resources are linked from other tools.

Q 1.3. *How to correctly evaluate a semantic application?*

A challenge related to evaluation is the lack of a standardised dataset to use, due to the highly personal data required. Even if such a dataset existed, due to the

fact that the applications on the Semantic Desktop are mainly related to personal information, it is difficult for participants to use the data provided, as they are not familiar with it. Evaluating PIM tools has been shown to yield best results when the test users are asked to perform their own tasks in their own set-up rather than attempting to simulate it with artificial tasks and artificial data. The reduced number of semantic application in each area makes it difficult to evaluate a new tool against an existing established one, thus the best candidates for a comparative evaluation are conventional (i.e. not semantic) applications with similar functionality. While possibly re-demonstrating that indeed linked data is more useful, comparing a semantic and non-semantic application requires well thought metrics, as the semantic features that need evaluating have no direct counterparts against which to evaluate them. For task based evaluations, it is difficult to find a common set of tasks that both applications can perform. A solution is to choose general, high level tasks, but that influences the granularity of the results.

4.2 Tackling the Challenges: The Design of SemNotes

We use our note-taking application, SemNotes, to describe the general principles we used for designing an application for the Semantic Desktop. Note-taking is a good general use case for the Semantic Desktop because the information contained in notes is not restricted to a specific domain. Personal notes are naturally connected to the user's context, and can thus be meaningfully interconnected with much of the existing network of personal data on the desktop.

We divided the design into specialised modules. Each module handles a set of related tasks. We describe the modules as they would be for a general semantic application, followed by a more specific description about the implementation of SemNotes in the next section.

4.2.1 Data Representation

This module handles the vocabulary of the application: the *data types* around which the application is centred, and the kind of *metadata* that is needed for them. To enable other Semantic Desktop applications to understand the data produced, as well as for our application to understand data from the underlying Semantic Desktop, the best

practice is to reuse existing desktop ontologies where possible. The basic data type handled by SemNotes is the note, represented by a short snippet of text containing personal information.

4.2.2 Data Management

This module manages the life-cycle of the semantic data that the application handles, by enabling the transition between the phases of the life-cycle. We adapted the Abstract Data Life-Cycle Model [Möller, 2009] to illustrate a comprehensive workflow for Semantic Desktop data, from its creation to its termination. Figure 4.1 shows the phases and transitions between them, focusing on the possible actions that the user can execute, and hence, that a semantic application could support.

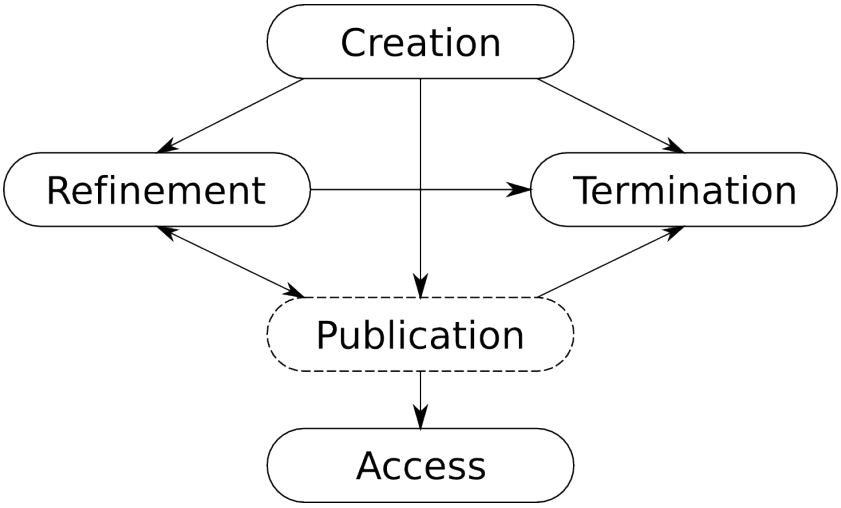

Figure 4.1: Semantic Desktop data life-cycle.

Creation. Most often creation implies creating a new resource, of a given type. However, the import of existing data from other applications or formats (e.g. crawlers) is also included here.

Refinement. This phase includes any activities that make changes to existing data. It also contains the creation and deletion of links between existing resources, although alternatively they can be included in the Creation or Termination phases, as links are data too.

Access. This phase is represented by accessing the data through either querying or browsing. Because we are discussing interlinked data, accessing a piece of semantic data implies recursively accessing the sub-graph of resources semantically related

to it. How much of the sub-graph is traversed can vary, and further traversal by the user should be supported and encouraged.

Termination. In this phase the data is deleted from the system. As with the access phase, rules must be defined to determine how much of the dataset a deletion will affect—e.g. it might make sense to delete all the subtasks of a task when the parent is deleted, but not to delete the documents related to the task.

Publication. This phase represents making the data accessible to users from outside the system. Also included here is exporting the data to other formats and applications. When handling semantic personal data, applications should ensure that sensitive data is well protected against unauthorised or accidental publication.

Furthermore, this module handles where and how the data is stored. The Semantic Desktop provides the framework for storing semantic data, therefore it is best that the central desktop repository be used, when practical. This enables easier interlinking with the rest of the data. In the case of SemNotes, the notes and all the metadata about them are stored in the desktop repository.

4.2.3 Interlinking

The interlinking module is logically a sub-module of the Data management one, as it specifically manages a part of the refinement phase of the data life-cycle. However, because it is an important part of any semantic application, relating to the first two challenges listed in Section 4.1.1, we describe it as a standalone module. The interlinking module effectively realises the goal of integrating the new semantic data into the pool of existing linked desktop data. The functionalities offered can vary from simple automatic linking of new resources to a specified context or to their author, to complex extraction and inference of new relations and resources. This module provides the feature that sets SemNotes apart from other note-taking tools, the interlinking of the notes with the desktop resources mentioned in them. In our application, there are two sub-modules, that handle (i) entity recognition, and (ii) information extraction, suggesting possible new connections to be created by the user.

4.2.4 Visualisation

The visualisation module presents the data to the user in a simple, yet useful and versatile way. It addresses the third challenge listed above: designing the interface. Depending on the application, the visualisation can include *aggregated* views on the data, and *filters*. Faceted search [Yee et al., 2003] has proven useful for semantic data, and it can be used to present the interlinked information to the user in a meaningful way. In SemNotes, the data that needs to be visualised is basically an enhanced version of a list of notes, with sorting and filtering. The module also provides the note editor. An important part of the module is displaying the recommendations for interlinking, a difficult task due to the heterogeneous nature of the information to be presented in a uniform, uncluttered way.

We describe how we tackled the last question — the evaluation of the application — in Section 4.4.

4.3 Implementation of SemNotes

In the development of SemNotes, we tried to reuse as much as possible of the features provided by the host Semantic Desktop, Nepomuk-KDE. Using the existing functionality enabled better integration with the rest of the system, while reducing the effort required for the implementation. Nepomuk-KDE provides out of the box central RDF storage for the desktop, and an efficient means to access and query the data.

We describe below the implementation of each of the modules introduced in the design section.

4.3.1 Data Representation

We describe the data created by SemNotes using a subset of the desktop ontologies described in Section 3.2.3 — Personal Information Model (PIMO) and Nepomuk Annotation Ontology (NAO). Figure 4.2 shows a basic note with metadata, and Listing 4.1 contains the Turtle representation of the same example.

The central unit of information handled by SemNotes is the note — represented as an instance of `pimo:Note`. The information stored for each note consists of:

- title — `nao:prefLabel`
- content — `nao:description`
- creation date — `nao:created`
- last modification date — `nao:lastModified`
- rating — `nao:numericRating`
- tags — `nao:hasTag`
- related desktop resources — `pimo:isRelated`

The Nepomuk ontologies make the distinction between resources representing native computer structures, which are described with the Nepomuk Information Element (NIE) ontology and resources representing concepts in the real world, which are described with the Personal Information Model (PIMO).

Before representing the PIMO concepts, most of the semantic data on the desktop is extracted from `nie:DataObject`s and interpreted as `nie:InformationElement`s. This is due to the fact that generally the information is still created by non-semantic applications, and to make it useful to the Semantic Desktop it has to be transformed, while keeping provenance information and feeding back into the applications that created it.

The extra step of extracting resources is not needed in the case of the information created directly for the Semantic Desktop by semantic applications. Thus, if no data is stored outside of the repository, no NIE resources are created. This is a characteristic of SemNotes and of other semantic applications for the Semantic Desktop.

The `pimo:Note`s created with SemNotes can however be exported as text or HTML files, for backup or other purposes, thus associating a NIE resource to a note. This is the reverse of the usual process, and the change stems from working directly with the framework provided by the Semantic Desktop.

Note metadata

Tagging, rating and commenting are basic features provided out of the box by the Nepomuk-KDE system, for all types of resources. In SemNotes the only categorisation mechanism we use are tags, preferring simplicity over the more accurate mix of categories, topics and tags. The `nao:hasTag` relation links the note to system-wide tag instances, thus enabling the reuse of tags throughout all applications, reducing duplication of classification work for the user.

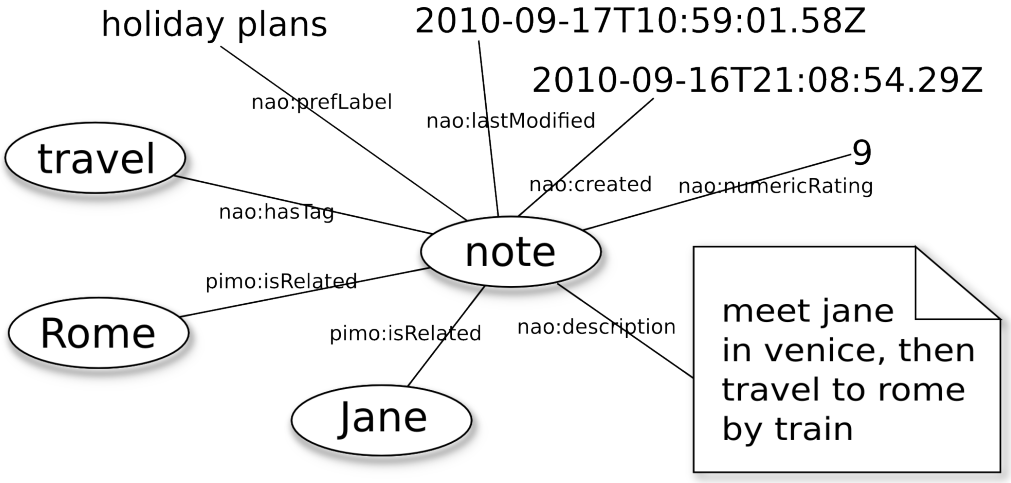

Figure 4.2: Graph representation of the information about a note.

We later added support for rating to the interface, leaving the meaning of the rating open for the user to decide — some possible examples include importance, urgency, quality of the content, or readiness for publication (in the case of a draft blog post as will be shown in Chapter 7).

Commenting on notes, although supported by the underlying system, because notes are resources, is not supported in the interface of SemNotes. We made this decision because the notes are themselves a type of comment, and we considered the feature redundant.

```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix pimo: <http://www.semanticdesktop.org/ontologies/2007/11/01/pimo#> .
@prefix nao: <http://www.semanticdesktop.org/ontologies/2007/08/15/nao#> .

<nepomuk:/res/thenoteuri> a pimo:Note ;
        nao:prefLabel "holiday plans"^^xsd:string ;
        nao:description "<html>...</html>"^^xsd:string ;
        nao:created "2010-09-16T21:08:54.29Z"^^xsd:dateTime ;
        nao:lastModified "2010-09-17T10:59:01.58Z"^^xsd:dateTime ;
        nao:numericRating "9"^^xsd:int ;
        nao:hasTag <nepomuk:/res/travel> ;
        pimo:isRelated <nepomuk:/res/Rome>, <nepomuk:/res/Jane> .
```

Listing 4.1: RDF representation of a note.

Note content

Because notes are generally short [Bernstein et al., 2008] we decided to store the note content in the RDF repository, as a property of the note (`nao:description`). The value is the HTML string representing the content of the note, including formatting. This decision enables us to use the indexing and full text search feature provided by Nepomuk-KDE.

Using the general property `nao:description` to store the content of the notes, opens up the possibility of treating any semantic resource on the desktop as a note in SemNotes. This is equivalent to adding comments on each resource, but employing the functionalities provided by the application, including the analysis of the text to suggest relations. This enables serendipity — discovering non-obvious connections between any desktop resources.

Related resources

As we discussed, the most important feature of SemNotes is the interlinking of notes with relevant resources from the desktop. The relations are stored using `pimo:isRelated`. In the current revision of SemNotes, this is the only type of relation. This decision was based on the results of the long term study of Gnowsis usage [Sauermann, 2008] which found that for PIM tasks it is enough to express that two things are related, and that the simpler properties are preferred by users over the more specific ones, regardless of the possible loss of meaning. However, we consider extending the range of possible relations in future versions. Having the information about the resources that are linked, specifically about the type of the resources, we can infer the possible relations to suggest, based also on knowledge from the desktop ontologies.

Restricting the types of possible relations also keeps the interface simple, which was one of the design goals, and one of the biggest challenges we encountered, as we show in more detail in 4.3.4. We further explain the extraction and creation of relations in 4.3.3.

4.3.2 Data Management

SemNotes supports all the phases of the note life-cycle.

When a new note is created, a new URI is generated for it, and the creation time is set. The rest of the properties are not set at creation time. As note data is added (i.e. the refinement phase), the metadata about the note and the content is updated. At each new update or annotation of the note, the last modification date is also updated. The notes can be found (i.e. the access phase) through full-text search, filtering by tags, related resources, and by creation date. Once the required note is found, it can be viewed in the editor. When a note is deleted, all metadata and relations about it are deleted as well, however, none of the tags and related resources are removed from the system, as they might originate from, or be used by other applications. The publication phase is also supported by SemNotes, as notes can be exported as HTML or text files, or even directly published online as blog posts [Dragan et al., 2010].

As mentioned above, SemNotes uses the RDF repository provided by Nepomuk-KDE for all data storage.

4.3.3 Interlinking

The interlinking of notes with related resources is the key feature of SemNotes. This feature realises the actual integration of the new information with the existing network of linked desktop data. Annotating the notes with related information captures the context of the note. Context is important for personal information management because it enables reminding and better (more precise, faster) search. Links between resources also support wayfinding [Jones, 2008] and encourage exploratory browsing and serendipity.

The module uses entity recognition and string matching algorithms to detect and suggest possible related resources, but no link is created until the user selects the correct one. This mixed-initiative approach is a compromise between the precision of the links created and the amount of interference with the user's workflow.

The current implementation suggests annotations based on the knowledge about existing desktop resources, using entity matching techniques to identify the likely candidates. Certain types of resources are more likely to be related to notes (e.g. people, organisations, projects, events, tasks, other notes, locations). By default, SemNotes restricts the search for suggested resources to these types, but the user can easily modify the list. We do not include every resource on the desktop because of the large number of files that are indexed by the Semantic Desktop, which would clutter the suggestion list. For the resources of the given types, all textual properties are compared against

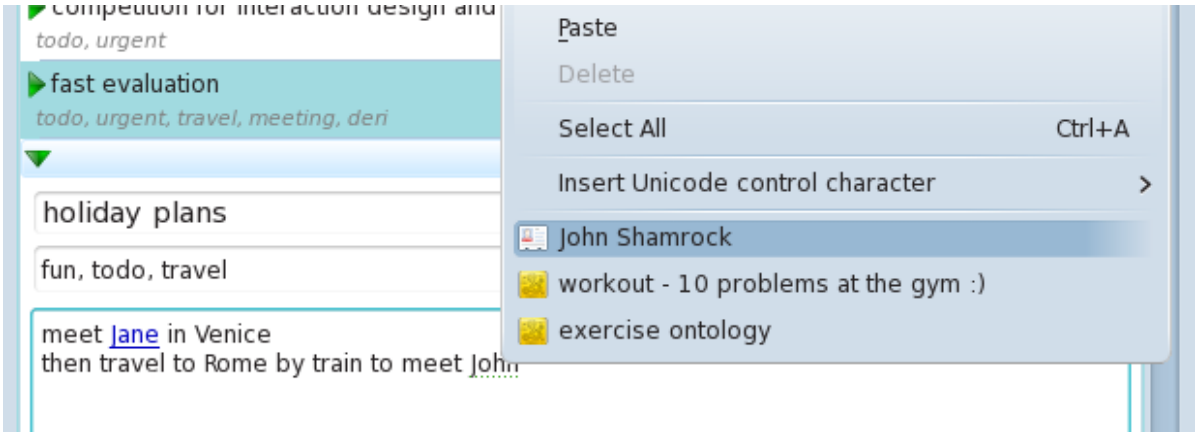

Figure 4.3: SemNotes annotation suggestion and link.

the note text. This way, resources that do not explicitly have the search term in their label will show up in the suggestion list. An example is shown in Figure 4.3: for the word "John" other notes that mention him are suggested, even though his name is not present in the label.

SemNotes does not currently offer suggestions based on online resources[2], unless there is a desktop resource previously created for the relevant Web data (i.e. a bookmarked Web page becomes a desktop resource).

We are working on an information extractor module, which identifies new information in the content of the notes. It will suggest the creation of new desktop resources from the text, like events, tasks or contacts, that will then be linked to the note.

Since the annotation suggestions are computed while the user types the note, efficient processing is required. The process of finding possible matches follows:

1. Scan the text and identify possible candidates represented by a single word or a sequence of words.
2. For each possible entity found in the text find a list of existing desktop resources that match it. We use string matching to compute a score for each resource found. The score takes into account the length of the matched string, and if the resource has been linked to the note before.
3. Sort the matches by score and present them to the user in a non-intrusive way (see Section 4.3.4).

[2]Similar to `www.zemanta.com`.

4. If the user chooses any of the presented suggestions
   - create a link between the piece of text identified as an entity and the actual resource it represents.
   - use the selected suggestion in the recalculation of the scores for the entities found for the rest of the note. Once a note is linked to a resource, that resource is more likely to appear again, and therefore it will be ranked higher.
5. If the user ignores the presented suggestions, no links are created, but the possible matches are saved for later use.

For the purpose of establishing context, and for organising notes, it is sufficient to create a single link between a note and a resource it is related to, regardless of how many times that resource is mentioned in the content of the note. Therefore the relation between the note and the desktop resource is created only once in the repository. However, if the note is viewed in SemNotes, all the links that the user created to the related resource are displayed.

The interlinking module also manages the removal of links between notes and desktop resources. Because the suggestion of related resources is based on the content of the notes, once the last textual link to a related resource is removed, so is the relation between the note and the respective resource.

SemNotes also supports the manual creation of links between the note and desktop resources that are relevant but have no explicit mention in the text.

4.3.4 Visualisation

SemNotes displays the notes as a list that can be sorted by title, creation or last modification dates, or rating. Each note can be opened in-list for quick access, or maximised over the entire window, for viewing and editing. In the in-list mode, several notes can be open and edited at once (see Figures 4.4 and 4.5 for the current version of the SemNotes user interface.).

An aggregated view of the list of notes is based on a restricted set of properties that the notes have in common. Depending on the set of properties used (one or more), the most suitable visual representation of the aggregated view varies. Currently SemNotes offers a tag cloud, a timeline and a related-resource view; each view aggregates information about the notes based on a single property (i.e. the tags associated, the creation date

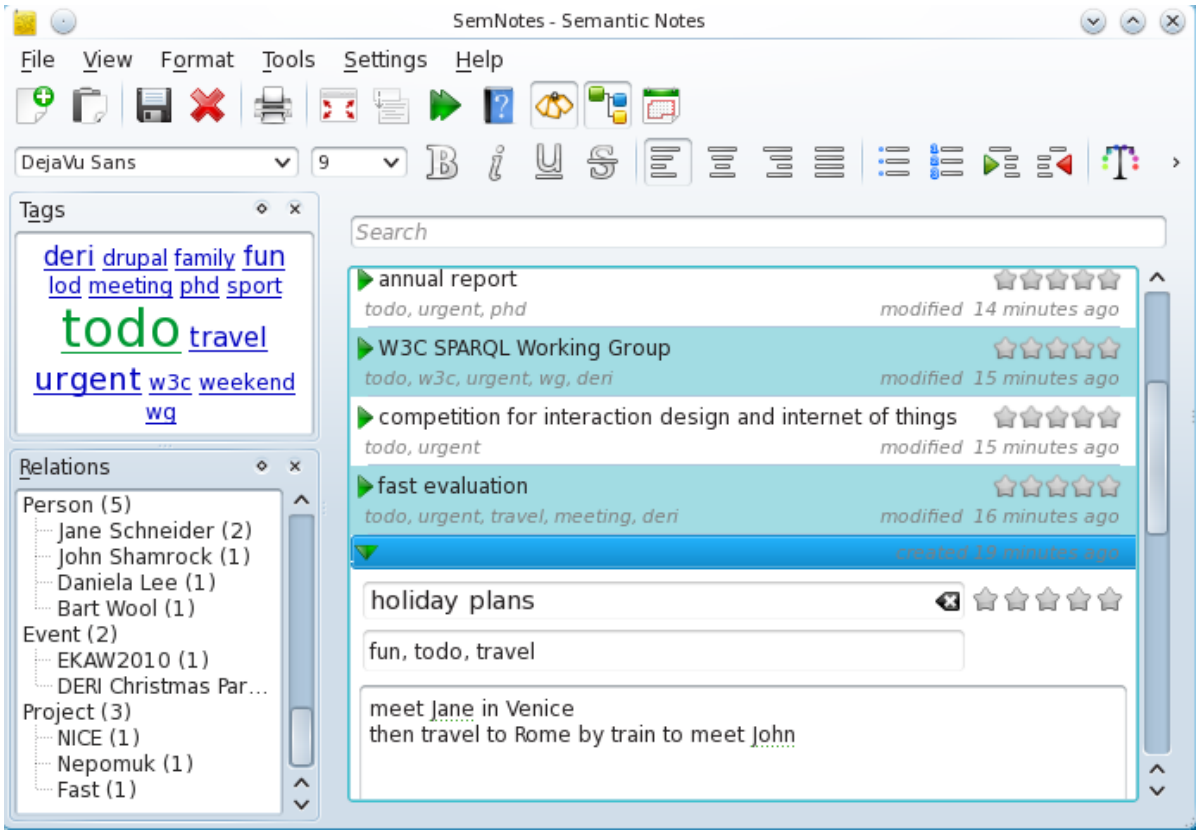

Figure 4.4: Current version of the SemNotes user interface.

and the related resources, respectively). Aggregated views are displayed adjacent to the list of notes, and can be hidden by the user to allow more space for editing.

These views also act as a custom faceted browser, as they provide filters on the list of notes. Filtering the list is as easy as clicking on a tag, time interval or related resource. A full text search box also acts as a filter on the list of notes, highlighting the search keywords in the content of open notes, if found. Multiple filters can be set at once, of mixed types. Figure 4.4 shows SemNotes with the tag cloud and related-resources views visible, and a tag filter set. A note is open in-list for editing.

The editor component provides rich text editing of the note content, as well as easy editing of the note metadata. Tagging provides auto-completion based on all the tags on the Semantic Desktop, and creating a new tag is done just by typing its label. If the user does not set a title, SemNotes automatically sets it to the first line of the note. For the rating we used the default visualisation provided by the Nepomuk-KDE libraries, for a uniform interface across the desktop. The creation and last modification dates are

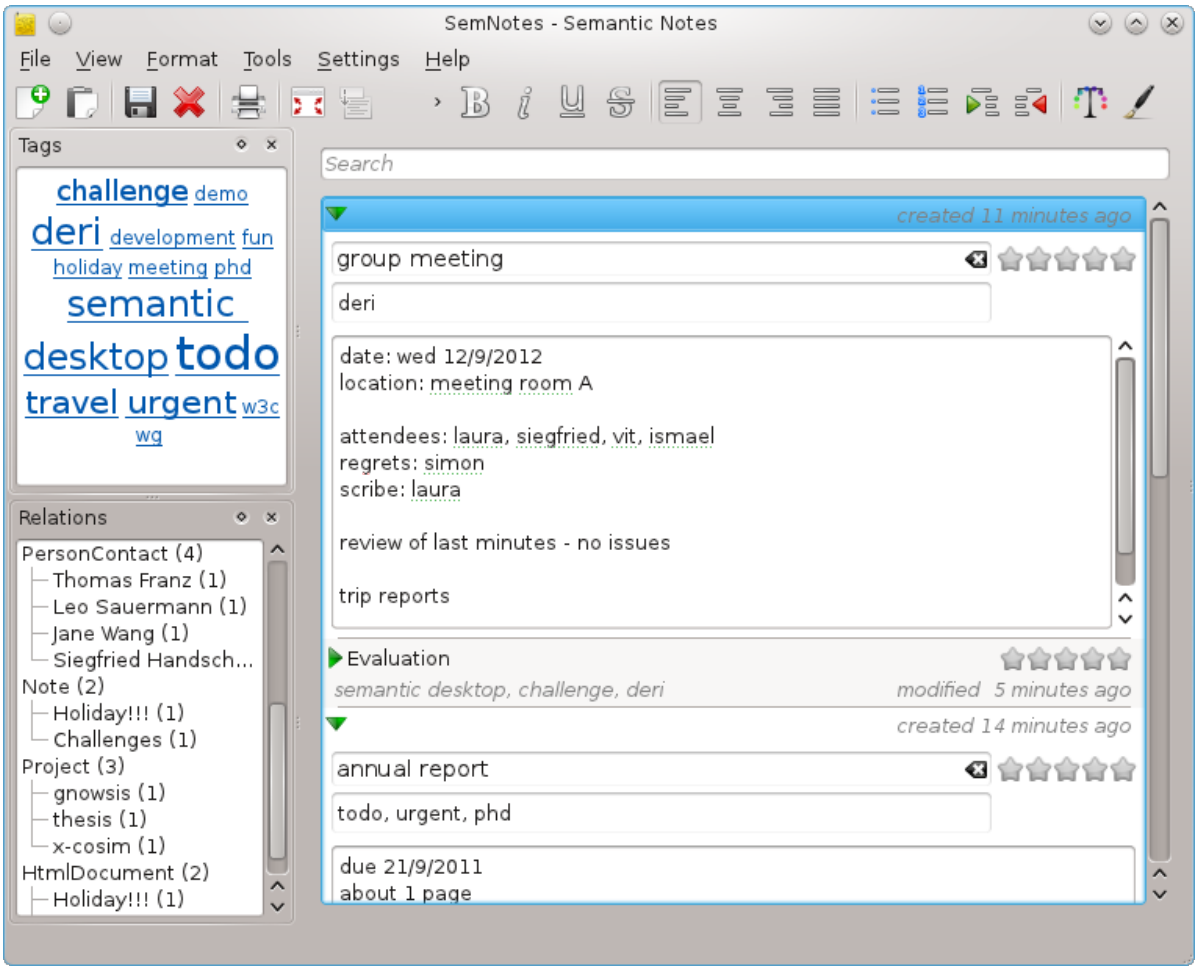

Figure 4.5: SemNotes main window, with two notes open in-list.

the only metadata which cannot be changed through the user interface of SemNotes, as these properties are set automatically. They can be tweaked by expert users, by accessing the RDF repository directly.

The suggestions for annotations are presented in a simple non-intrusive way, in the "spell-checker" style (i.e. the words for which suggestions were found are underlined with a green dotted line instead of a red squiggly line), and are available as context menu items, by right-clicking. Figure 4.3 depicts how annotation suggestions are presented, and how a linked resource is displayed in the note. To remove an annotation is just as easy as creating it — through a context menu item.

This presentation of the suggested resources is the result of several iterations of design, each improving on the previous one.

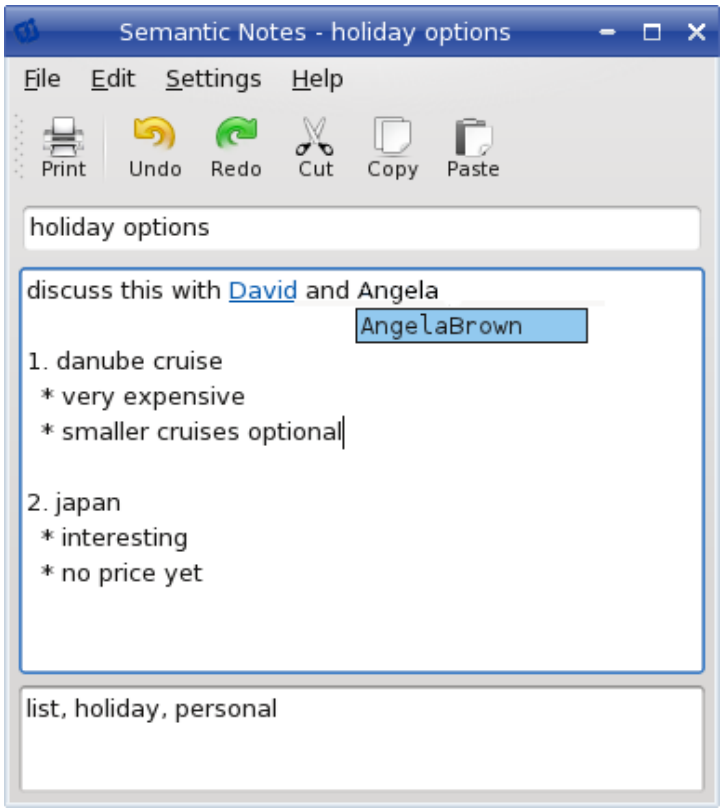

Figure 4.6: First version of the SemNotes note editor with pop-up style annotation suggestion.

The first iteration relied on localised pop-ups with labels of the suggested resources, in the style of text auto-completion (see Figure 4.6). The style worked well when typing the text, and it also enhanced the speed of writing through the auto-completion it provided. The type of annotation of the text during its creation is called latent annotation [Davis et al., 2010], and it is an improvement over the two-step process of first creating the content and then annotating it. Depending on the text of the note, the pop-ups could appear often and distract the user, but they could be dismissed by continuing to type. However, when portions of text were pasted in the editor, several pop-ups were generated at the same time, which was confusing and did not allow the users to make all the connections they would. Changing slightly the pop-ups to only be displayed one-by-one when text was pasted proved not to be a good solution either, as in the case of large paragraphs it would take a long time and many mouse clicks for the paste operation to finish, thus interfering with the flow of work.

Not just the interlinking support was changed from the first version of the application, but the entire user interface. The major redesign was supported by a usability study

done as part of the Season of Usability[3] 2009, by Daivee Patel, a Human-Computer Interaction student at Drexel University, Philadelphia, USA, with the mentoring help of usability expert Paul Hibbitts of Hibbitts Design, Canada[4]. The goals of the project were to recommend changes to improve and/or redesign the SemNotes user interface based on analysis and user research methods, and to perform a usability evaluation of the new changes with a range of users to validate the design recommendations.

The initial step of the project aimed at forming an understanding of what activities and tasks a user would perform with a note-taking application. This led to the use of a specific technique of task analysis called the activity grid based on the Activity Centred Design methodology. Using this method, we obtained a list of possible activities associated with using a note-taking tool, and each activity was subdivided into the tasks required to perform that activity and each task could then be further subdivided into possible actions. As the tasks became clearer, we identified the tasks that were relevant to SemNotes. These tasks helped focus the effort on the main usability issues and design specifically to address existing issues. A comparative analysis of five popular note-taking applications was conducted to help build a knowledge base for reference during the usability inspection of SemNotes. Each of the applications had their own unique features yet certain key functionality appeared standard across these kinds of applications. The examination proved helpful in determining the elements for redesign of the interface.

The usability inspection of SemNotes was performed to identify usability issues with the existing interface that could be captured without the need to user test before the redesign. Since the redesign was going to be based on key tasks or activities, we used a heuristic evaluation to identify top level issues. The criteria for evaluation was the ISO standard ISO 9241[ISO, 2006], the principles associated with this standard are based on research and have the benefit of international consensus. The results of the evaluation were used to help identify areas in the user interface most in need of design improvements, and to create a set of low fidelity mockups.

A series of usability tests were conducted using the paper prototyping method to validate the initial recommendations for the interface redesign. During each testing session, participants were asked to think aloud and point to elements on the illustrated paper-based screen that they would click or look at based on the tasks provided. No assistance was provided to the users during the testing sessions. The usability test participants

[3] `http://openusability.org`
[4] `http://paulhibbitts.com`

were selected via a screening survey on the criteria of experience with computer-based note-taking applications and note-taking habits. Based on the feedback from the first set of four users, the mockups were revised for the second round of usability testing while the tasks remained the same. The designs were further enhanced based on the feedback received in the second round.

Recommendation	Rationale
Multi Document Interface (presented as a single window with multiple panes)	By providing all key functionality within a single window, users did not have to manage multiple application windows.
Retain optional linking of text within linked editor via use of existing functionality (to click outside the auto complete pop-up) but to be supported by use of icons to indicate the type of resource being linked too	Users showed concern over unwanted text being linked.
Search should support searching by tags	Users realised that since the notes were already tagged during creation, they would like to search by tags for better results.
Sorting of notes and tags in the left pane	Multiple options for sorting would help users in locating specific notes and/or tags.
First user experience should be provided	Learning curve associated with new applications – having some introductory information and instructions in the opening screen will be useful.

Table 4.1: Final recommendations for the redesign of SemNotes user interface, based on the usability study.

Based on the accumulated results and feedback, we created a high fidelity mockup of the design, which included all the recommendations (see Table 4.1) made as a result of the project, and which were incorporated in the second version of SemNotes. The next iteration of the display for the annotation suggestions featured a side panel for each of the note edit boxes (see Figure 4.7). The side panel proved to create less interference with the user's workflow, although the suggestions became less obvious. The panel presented the suggestions as a list of resources, that when clicked would highlight the

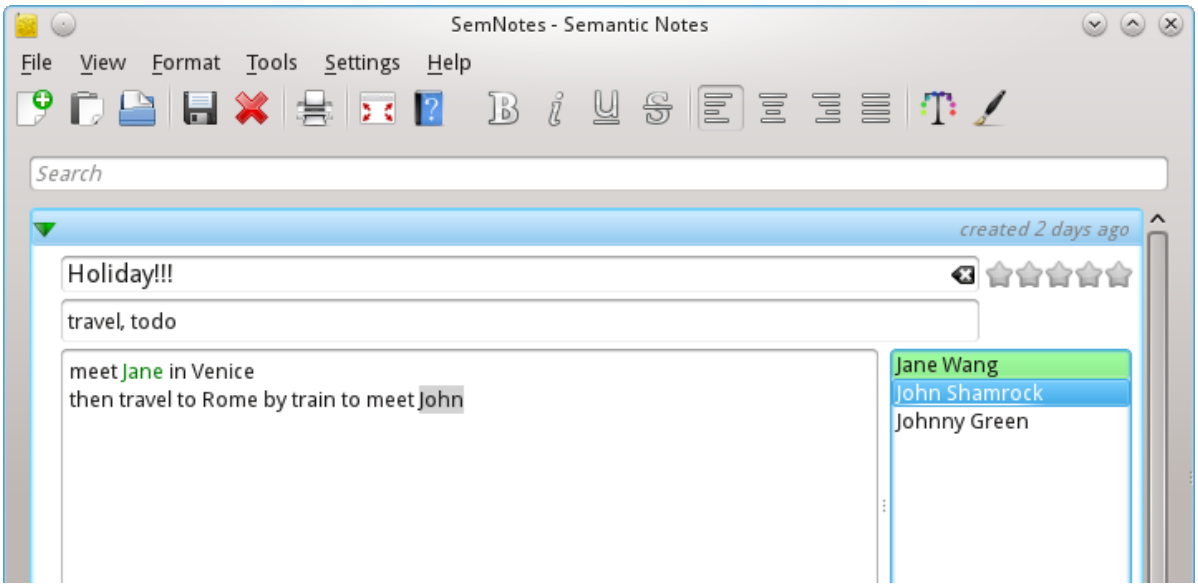

Figure 4.7: Second version of the SemNotes note editor with a side panel to display the annotation suggestions.

parts of the content to which they were relevant. This helped the users understand from where the suggestion stemmed and if it was indeed correct. To create a link between the note and a suggested resource, the user had to right click on the corresponding item in the list and select "Link". Resources could also be removed from the suggestion list, and never be shown again for the current note, to help clear the list from overpopulating. This interface shifted the initial latent annotation style of SemNotes, back to a classic two-step process. Although the suggestions were still computed on the fly, as the user typed or pasted new text, because they were out of focus, users were inclined to leave the annotation step for later. Another issue of the side-panel variant was that it limited the screen real-estate for the most important function of SemNotes, that of note-taking.

To keep more of the screen space for the note editor, in the next iteration, we have abandoned the side panel in favour of the "spell-checker" type of notification. It is a mix of the first and second iterations, by giving the users the immediate feedback of the pop-ups, through the green dotted line that underlines words, while being unobtrusive in the note-taking activity. The suggestions are computed on-the-fly like before, but they are only shown if the user right-clicks on the underlined word or words, in keeping with the spell-checker metaphor.

The three iterations of user interface design have improved significantly the usability of SemNotes, as well as the ease of interlinking. However, further usability testing is

needed to determine whether and how different types of relations can be created between the notes and the related desktop resources.

4.4 Evaluation

We conducted a task-based summative user evaluation, comparing SemNotes to the popular note-taking application Evernote[5]. The goal of the experiment was to determine whether the effort required for the creation of links between notes and resources is repaid by easier search. Towards this goal, we measured the effort needed to execute the same set of tasks with both tools, and compared the results. We used time spent, number of mouse clicks and number of key presses, as measures for effort. After the experiment, we asked the participants about their experience in a questionnaire.

The two applications compared have the same main functionality, note-taking. We chose Evernote as baseline, as it is a very popular note-taking tool that is freely available. Its set of functionalities are richer than those of SemNotes, but the basic features are present and similar in both applications. The feature that distinguishes SemNotes is the same that we want to measure—the creation of links between the notes and desktop resources, based on suggestions given by the application. SemNotes runs on KDE on Linux, and uses the framework provided by Nepomuk-KDE Semantic Desktop. Evernote runs on several operating systems for desktop and mobiles, but Linux is not one of them, therefore we used its Windows version.

Evaluation is one of the challenges for application design on the Semantic Desktop. Adding to the challenges of evaluating PIM applications with regards to finding appropriate data and tasks for significant measurements, discussed in [Kelly, 2006], semantic PIM applications have the difficulty of lacking equivalent semantic tools for a one-on-one comparison. That is why for this evaluation, we followed a methodology similar to the ones described in [Elsweiler and Ruthven, 2007] and [Franz et al., 2009], of comparing our semantically enabled SemNotes, to a conventional application.

One aspect that we did not evaluate as part of this study is the effect of having the notes connected to the relevant desktop resources from the other applications using the respective resources — for example, would the extra information from the notes provide an advantage when using a task manager application, and seeing all the linked notes

[5] `http://evernote.com/`

when looking at a task. Such a study would require a longer duration, and also a richer ecosystem of semantic applications, which make use each other's data.

4.4.1 Participants

Twenty participants took part in the evaluation, all members of our research institute. They are researchers in the field of Semantic Web, thus possibly favouring the semantic application. This bias (if it exists) would however influence only their responses in the questionnaire and not the measured values. Fourteen participants regularly use note-taking and five of them use Evernote as their preferred note-taking tool. None of the participants used Linux as their operating system. Their familiarity with one environment and one application could have influenced the measured results, in favour of Evernote.

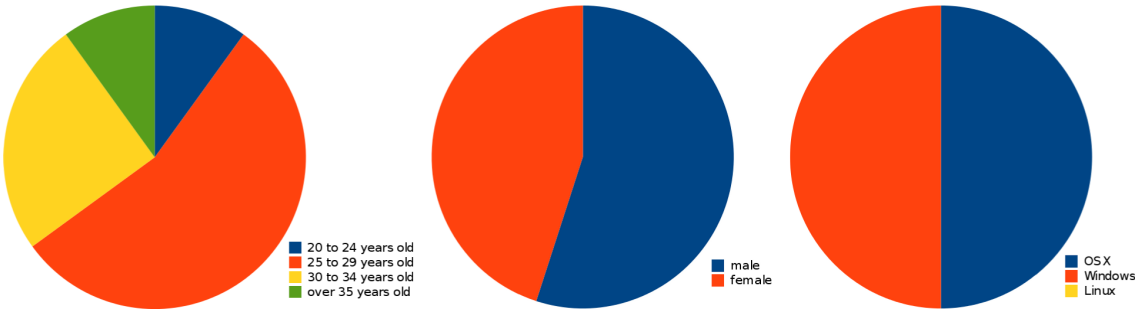

Figure 4.8: Age, gender and OS distributions of participants.

Some additional demographic details of the participants: gender distribution was close to even (eleven men and nine women); most of them were in the age groups between 25 and 29 (eleven, equivalent to 55%), and between 30 and 34 (five, or 25%). There were seven Master's students, seven Ph.D. students, five senior researchers and one intern. For demographic distributions see Figure 4.8 and Figure 4.9.

4.4.2 Data

We used two virtual machines for the experiment, preloaded with identical data. To reduce the artificialness of the study, we chose general data familiar to the participants, to which they have access in their everyday work. The dataset contained contact information for 130 members of our institute; 655 recent emails from our mailing lists; 20 scientific papers authored by our colleagues; and photographs from institute events. The

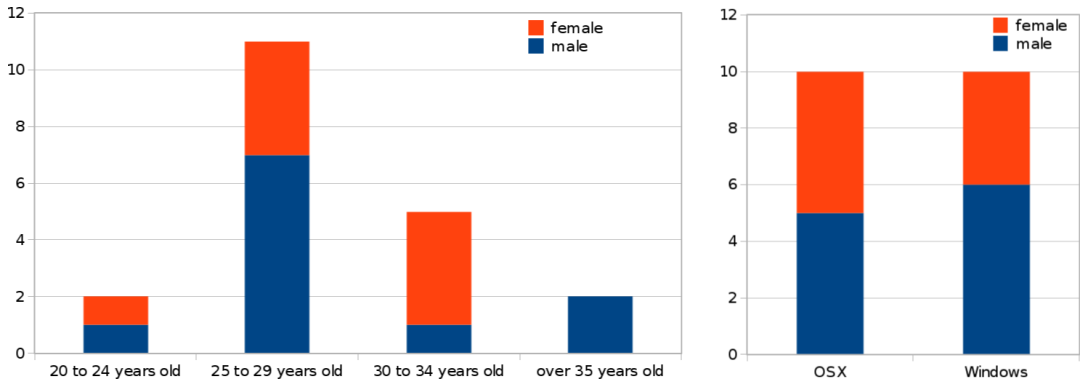

Figure 4.9: Age and OS distributions of participants per gender.

note data was also identical: 50 notes on a variety of topics, personal or work-related, tagged with 23 tags. In SemNotes, we also provided links between the notes and the resources mentioned in them: people, projects, events or other notes, within reasonable limits we expect users to interlink their notes (i.e. minimum 0 links, maximum 10 links, average 1.8, median 1). The majority of connections were made to people mentioned in the notes.

4.4.3 Tasks

We prepared a set of eight tasks. Each participant was requested to run all the tasks in each of the two environments. To prevent order effects from influencing the results, half of the participants started with SemNotes and the other half with Evernote.

T1. Find what information is available about yourself (contact information, documents, emails, photographs)

T2. Find the paper titled "Bridging the Gap between Linked Data and the Semantic Web". Who are the authors?

T3. Find notes tagged with "todo".

T4. Find to-dos that are related to our institute.

T5. Find a to-do related to a presentation by a colleague John.

T6. Take a note about planning a social event for your research group. Write the names of two people that have already confirmed. Annotate the note as you see fit.

T7. Find a note containing the minutes from the last meeting about a given project. Change the date of the next meeting planned.

T8. Take a new note for the action item assigned to you at the last meeting of the project. The action item is in the meeting minutes previously edited, and it requires drafting a document using a paper authored by a colleague. Annotate the note.

The first two tasks were intended to help the participants get accustomed with the data available and the environment; we did not include the time spent on these tasks in the results.

The last six tasks were focused on note-taking, and their complexity increased gradually. T3, T4, and T5 are search and filter tasks (S), from the very simple to the complex. T6 and T8 are editing tasks (E), including any annotations that the participants made on the notes. T7 is both a search and edit task (SE). It prepares the participants for the most complex search and edit task, T8, which required interaction with the rest of the system (i.e. finding the required paper).

4.4.4 Measurements

The participants were recorded while doing the tasks of the experiment, and the videos were used to extract the measurements used in the analysis. For each task we measured the effort in seconds spent, and number of mouse clicks. We also counted the number of key strokes for the tasks involving creation of notes, and we used this measure to normalise the values for time. Although we had two separate measures for each task, we were interested in the difference between these values, computed according to:

$$value_{difference} = value_{SemNotes} - value_{Evernote}$$

Thus, a positive value means that the value recorded for SemNotes is bigger than the corresponding value for Evernote, and a negative value means that the value for SemNotes is smaller.

For the time measurements, the time difference represents the extra *effort* required for annotating the notes with links in the editing tasks. For the search tasks the difference shows which of the applications enables the users to find the notes easier, thus the *benefit*. We must specify here that actual searches done by the software were considered instantaneous, as the datasets are small and we did not intend to measure the performance of the search algorithms used.

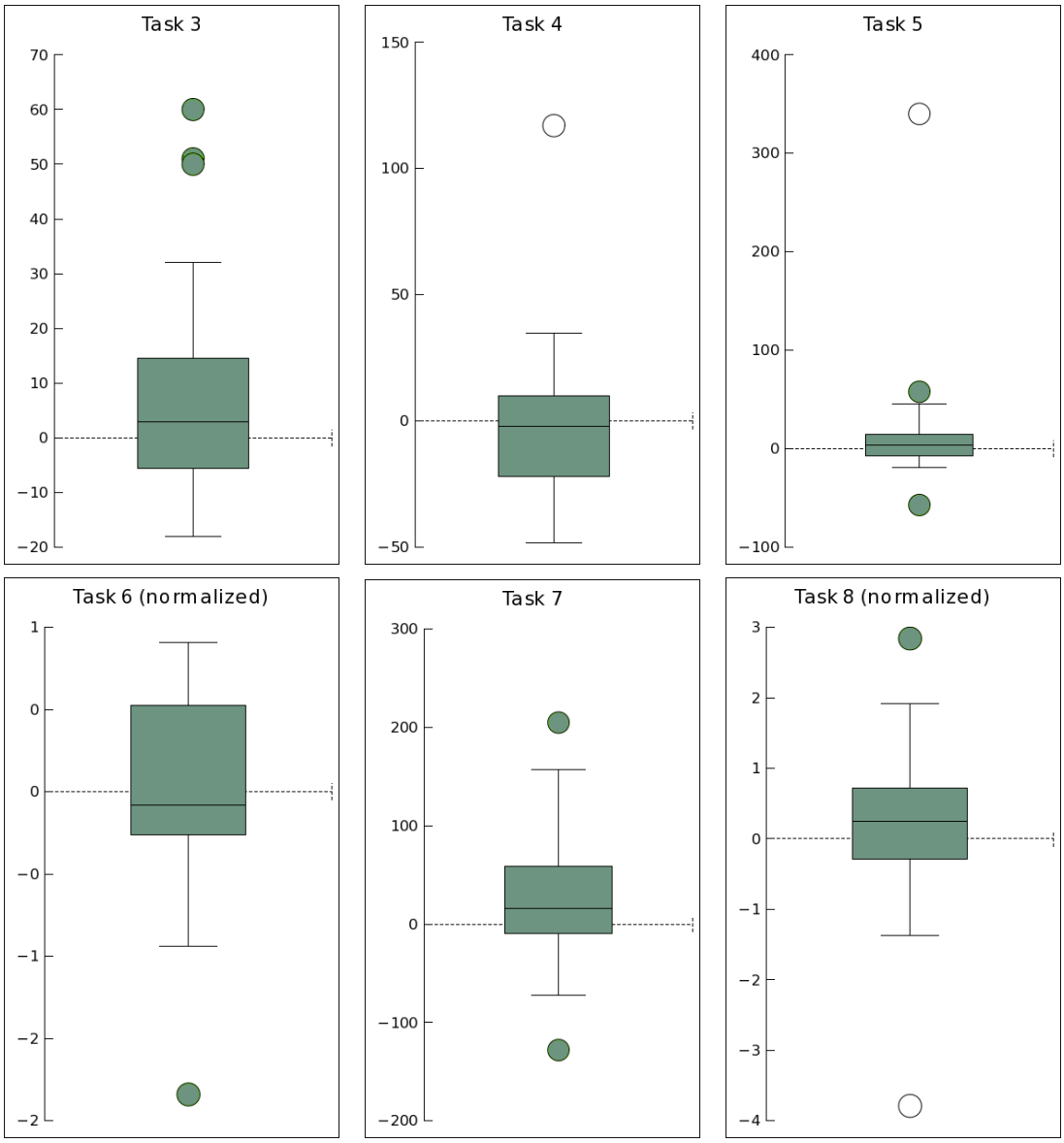

Figure 4.10: Inter-quartile ranges for the time difference.

For the number of mouse clicks, the difference represents the extra *effort* required, for both types of tasks.

After a first analysis of the results, we noticed that some values for time were unrealistically high or low. They coincided with the measurements when the participants stopped to ask a question or comment on a feature. We decided to eliminate these outlier values by using only the measurements that fall in the inter-quartile range, for all tasks — see Figure 4.10.

4.4.5 Quantitative Results

We tested if the effort measured in time and mouse clicks was different when using SemNotes than when using Evernote. Table 4.2 shows the results, with the statistically significant values in bold ($p < 0.05$).

T		Time (s)			Clicks		
		Avg	**Med**	***t***	**Avg**	**Med**	***t***
T3	(S)	0.5	0.0	0.152	0.167	0.0	0.692
T4	(S)	-8.0	-8.0	**-2.94**	-0.333	-1.0	-0.48
T5	(S)	-0.125	1.0	-0.046	0.857	1.0	1.426
T6	(E)	0.063	0.016	0.486	6.067	8.0	2.026
T7	(SE)	14.357	13.0	1.713	4.812	2.0	1.527
T8	(SE)	0.249	0.243	1.004	20.8	12.0	**3.08**

Table 4.2: Statistics for time and click differences.

Results show that for the simple search task T3 the difference is positive, which suggests that it takes longer to finish the task with SemNotes. However, the difference is not significant. For the complex search tasks that follow, T4 and T5 the difference has negative values, showing that the users spent less time on complex searches when using SemNotes, thus supporting the claim that the use of interlinked data makes notes easier to find. Only the results for T4 are statistically significant. None of the measures of number of mouse clicks for the search tasks are statistically significant, the differences being in average less than 1 click, with a median value of 0, -1 and +1 respectively for the tasks T3, T4 and T5.

For the editing task T6, the values are positive, thus it took longer to finish it with SemNotes. This was expected, as in SemNotes there is the additional step of annotating the notes with links to desktop resources. However, the differences are very close to 0 (0.063s average and 0.016s median) and not statistically significant. This editing task did require in average 6 more mouse clicks in SemNotes, with a median value of 8, but the difference is also not statistically significant.

For the more complex search and edit tasks T7 and T8, the time differences, while positive, thus in favour of Evernote, are not significant. For both tasks the values were

positive, which mean more clicks when using SemNotes. However, task T8, that required creating and annotating a complex note based on information from other sources, had statistical significant difference in number of clicks, in favour of Evernote (more clicks needed in SemNotes).

The positive difference in the number of clicks required for all editing tasks (T6, T7 and T8) is not surprising, as the participants recognised the value of creating links and proceeded to link the new note to the relevant resources. This was however a motivation for providing better keyboard support for the annotation of notes in the future version of the tool.

In summary, the results show that there are significant improvements (for one of two tasks) in the time spent on complex searches, when the data is interlinked, at no significant extra cost for the creation of the links. Linking does however significantly increase the effort measured by number of mouse clicks (for one of two tasks).

4.4.6 Questionnaire

We asked the participants to fill in an anonymous questionnaire related to the experiment. The questionnaire is listed in Appendix A. According to the answers, on a scale from 1 to 5, the tasks were simple (mean 2.25) and similar to the ones in their daily work (mean 3.4), and the data provided was familiar (mean 3.2).

The answers also show that 60% of the participants (12) felt that SemNotes helped them finish the tasks *faster*, while only 20% (4) said Evernote, and 20% did not feel any difference between the tools. When asked which of the two applications helped them perform the tasks *better*, 80% of the participants chose SemNotes, while the remaining 20% did not feel that there was any difference (see Figure 4.11 for a graphical representation).

The participants had a good overall impression of SemNotes (with a mean of 4.15 on a scale from 1 to 5). This rating was supported by comments like "*That was cool.*", and requests for SemNotes for other operating systems: "*Maybe you could make an OS X version?*" and "*When is a port for Windows 7 coming?*".

The semantic annotation was one of the most liked features of SemNotes, and one of the features most missed in Evernote, according to the questionnaire. Another advantage of SemNotes was considered the multiple filtering by mixed criteria, with several

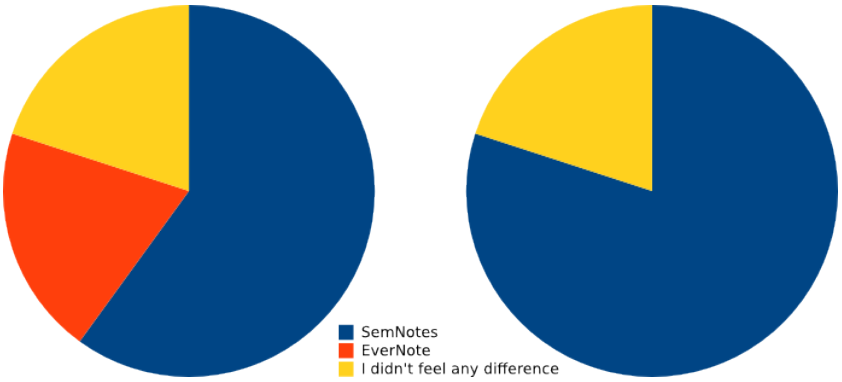

Figure 4.11: Which tool helped perform the tasks faster (left) and better (right).

participants considering it the best feature. The rating of notes was listed as the least liked feature of SemNotes by three participants. The tag cloud was the most controversial feature, as many participants liked it and found it very useful, while others would have preferred a simple list of tags: "*Usually I don't like tag clouds, but the one in SemNotes was really useful.*" and "*While the tag cloud helps in determining the most used tag, a simple list with the tags seems to be easier to search.*"

4.5 Related Work

Semantic note-taking means enhancing the note-taking process using Semantic Web technologies. It can refer to the techniques and methods used in the implementation, like ontologies and RDF, but most importantly it is about creating a semantic network around the notes and the information contained in them. There are several applications that enable more or less semantic note-taking: some are browser based (online or offline), while others are standalone desktop applications as is SemNotes.

The List.it browser-based note-taking tool [Kleek et al., 2009] and Jourknow, its predecessor [Kleek et al., 2007], save context alongside the information scraps, to improve re-finding and reminding. List.it also features information extraction from the unstructured text of the notes, recognising entities and relations between them, with the *pidgin* language processor. However, unlike SemNotes, the contex of the notes they create does not include links to any existing desktop resources. SnapShoot [Iga and Shinnishi, 2006] is another browser-based note-taking tool that explores new visualisation techniques to improve reading of the documents produced. It features categorisation, and limited interlinking with documents within the system.

MindRaider[Martin Dvorak, 2012], described above in Section 3.2.2, is an open source "Semantic Web outliner" and extended note-taking system for information organisation. While it only interlinks concepts within its maps, it can connect to the Gnowsis Semantic Desktop through a plugin, thus potentially it can use any Semantic Desktop resources through a similar mechanism.

A distinct category of semantic note-taking applications are personal semantic wikis, like Kaukolu[van Elst et al., 2008], IkeWiki[Schaffert et al., 2006] and GDKTiddlyWiki. Each wiki page represents a resource and its semantic relations to other resources are encoded within the page, using an extension to the wiki syntax. Only predefined relations and types are available, and the wikis offer limited access to other desktop information sources. Unlike SemNotes, connections are only possible between resources within the wiki system.

OneNote from Microsoft's office suite provides quasi-semantic functionality by interlinking the notes with address book information, calendar, and tasks. It does not use any semantic technologies though, and the data is locked in by proprietary formats and storage.

Zemanta[6] is a blog assistant that suggests possible enhancements to blog posts, like linking external content and images. Unlike SemNotes, which uses the local repository to search for matches, Zemanta looks on the Web. It does not assign any semantics to the links.

There are also specialised systems like the SemanticPen [Varadarajan et al., 2005] which provides support for semantic note-taking with pen devices. Other note-taking applications that provide semantic features include: Jenga Note[7] — allows associating a note with a concept, Catch Notes[8], and SpringPad[9].

4.6 Conclusion

In this chapter, we presented a solution to the challenges of designing applications for the Semantic Desktop. We described one such application, SemNotes, a semantic note-taking tool for the Nepomuk-KDE Semantic Desktop. It provides a real-world, functional

[6] http://www.zemanta.com
[7] http://www.jenganote.org
[8] http://www.catch.com
[9] http://www.springpadit.com

use case for fully exploiting the capabilities of the Semantic Desktop: interlinking, organisation and management of personal information, efficient search and browsing. It supports the entire life-cycle of the semantic data represented by notes, with emphasis on the creation of links between the new data and the existing network of linked information on the desktop.

Through SemNotes we present a possible solution to each of the challenges presented in the begining of the chapter. To the first question of creating semantic data, we describe how we create new semantic notes, using the existing vocabularies and creating links to the existing Semantic Desktop data available. To the second question of designing the human computer interaction, we describe the efforts towards an easy to use yet rich user interface design for SemNotes. With the help of usability experts and user testing the interface went so far through three iterations, each described in this chapter.

For the third challenge, that of evaluating a semantic tool, we described a task-based user evaluation comparing SemNotes to the popular note-taking application Evernote. The results show that the extra effort (measured in time spent) required for the annotation of notes with links to related resources is not significant. However, the benefit (measured in time saved) is significant for one of two complex search tasks.

SemNotes embodies a simple and user-friendly way of generating new semantic data on the desktop, and integrating it with the already existing data. However, the personal information that the users work with is not restricted to the desktop. Further, semantic data exists in large quantities outside of the desktop, on the Web and in organisational repositories. Thus, in the next chapter we continue by describing how the linked information from the desktop can be meaningfully and safely connected to the outside world.

Chapter 5

Bridging the Gap between the Semantic Desktop and the Web of Data

Based on "Linking Semantic Desktop Data to the Web of Data" [Dragan et al., 2011a] published at the 10th International Semantic Web Conference (ISWC 2011)

The previous chapters showed how the Semantic Desktop enables better personal information management on our computers, using semantic technologies to explicitly connect information which is naturally connected in the real-world, and thus matching the representation to a user's mental model. However, our personal data is often a reflection of our subjective view, or limited knowledge. Thus, although it may seem huge when faced with the task of organising it, the information on the desktop is small compared to the amount of information available online. The Web of Data contains a linked network of information similar to the one on the desktop, only at a much larger scale. Connecting the desktop to the Web of Data would enrich and complement desktop information.

This chapter presents a solution to the second research question **Q 2.** from Section 1.2, *how to expand the scope of the Semantic Desktop into the realm of the Web of Data, to enhance the user experience and benefit?*, more specifically, to the first sub-question **Q 2.1.** described there — *how to find instances representing the same real-world thing described by a Semantic Desktop resource?*

Our solution uses a semantic search engine for the Web of Data, such as Sindice[1], to find and retrieve a relevant subset of entities from the Web. We present a matching framework, using a combination of configurable heuristics and rules to compare data graphs, that achieves a high degree of precision for the linking decision. We evaluate our methodology with real-world data — we create a gold standard from relevance judgements by experts; and measure the performance of our system against it. We thus show that it is possible to automatically link desktop data with Web data in an effective way.

5.1 Introduction

The Semantic Desktop aims to enable better organisation of the personal information on our computers, by applying semantic technologies on the desktop. Just as Linked Data connects distributed data on the Web, creating a network of interlinked information, the Semantic Desktop connects personal data across application boundaries on the desktop, creating a network of personal information. However, information on our desktop is often incomplete, as it is based on our subjective view, or limited knowledge about an entity.

On the other hand, the Web of Data contains information about virtually everything, generated by multiple sources, and theoretically unlimited. Connecting the desktop to the Web of Data would thus enrich and complement desktop information. Bringing in information automatically from the Web of Data would release the user from the burden of searching for information.

Connecting the two networks of information opens up the possibility of personal services on the desktop, which use external data but in the personal context of the user, highly connected to his personal data and focused on his interests. One such example is a service that finds implicit links between the publications that the user has on the desktop, and provides recommendations to other publications on the same topics, by the same authors, or related in another way. Another desktop service could use information from the Web of Data to notify the user of new concert dates in his area, based on the latest or most popular artists played on the desktop. Web data can also be used as a point of reference when working collaboratively, e.g., documents linked by the user to people, projects, or other resources from his semantic desktop can be shared together

[1] `http://sindice.com/`

with the annotations, which can be accessed and reused outside of the Semantic Desktop where they were generated.

From the perspective of interlinking information, and using the frameworks provided by the Semantic Desktop and the Web of Data, we have separate islands of knowledge, both containing similar data, related to the same topics of interest to the user, but disconnected from each other.

The disconnection appears in two forms:

- The data on the desktop, although similar to that on the Web of Data, is described using specific *desktop ontologies*, which are different from the ones found on the Web of Data. This schema mismatch makes interlinking data from the two datasets difficult.

- Identifiers (URIs) on the desktop are local to the desktop data space, they are not globally unique and cannot be dereferenced as normal Linked Data URIs are. Hence, it is hard to access and connect to local data from the Web.

To tackle this disconnection, it is necessary to create links between desktop identifiers and Web identifiers that refer to the same real-world thing. This means we need to compare the data graph describing an entity on the desktop with the data graph of an entity on the Web. Leaving aside the use of different terminology within the data, the Web of Data is large, with billions of entities across hundreds of thousands of datasets. From this vast amount of information we must find and retrieve a relevant subset of entities, that are potential candidates with the desktop entity. Then we must decide if the candidates are similar enough with the desktop entity to create a link between the two. Because we wish to make the interlinking automatic, we must be able to decide with a high degree of precision which candidates among this subset are in fact referring to the same entity.

Our solution tackles the problems raised above by using a semantic search engine for the Web of Data, such as Sindice, to find and retrieve a relevant subset of entities from the Web. We then present a matching framework, using a combination of configurable heuristics and rules to compare data graphs, that achieves a high degree of precision in the linking decision.

To evaluate our methodology with real-world data, we create a gold standard from relevance judgements by experts, and we measure the performance of our system against it.

Our solution proves that interlinking the two environments is feasible, and even more, it yields good results. Connecting desktop data with the Web enables the system to bring Web data to the users, instead of the users having to go find it by themself.

5.2 Related Work

The problem of entity linking is well known across various research communities with a variety of different names, such as record linkage [Fellegi and Sunter, 1969], entity resolution [Benjelloun et al., 2006], reference reconciliation [Dong et al., 2005] or object consolidation [Hogan et al., 2007]. A wide variety of algorithms has been developed for resolving the co-reference problem, but record linkage between distributed databases is still considered a difficult problem.

Recent initiatives within the Semantic Web community address the problem of linking entities across data sources. For instance, [Jaffri et al., 2007] describe the phenomenon of proliferation of URIs and propose a Consistent Reference Service to manage URI equivalences. The OKKAM project [Bouquet et al., 2007] proposes an infrastructure for assigning global identifiers at web scale. These approaches are more focussed towards the management of entity identity on the Web, but do not provide an easy means to create new links between data sources.

Similar to our approach, [Raimond et al., 2008] describe an algorithm and its implementation GNAT, for linking a personal music collection to corresponding MusicBrainz resources. The approach recursively measures the similarity of the resource graphs from the two datasets, with the limitation that the same vocabularies must be used in both. By contrast, using property paths in our mappings, we eliminate the need for recursion while still propagating the measures from connected resources. Silk is a framework for linking multiple entities between two datasets [Bizer et al., 2009]. It relies on user-defined rules and various string matching algorithms to measure the similarity between two entities. In this case it is necessary to know a priori which specific dataset to link to and to perform manual configuration of the matching algorithms, something that requires a high degree of expertise. [Hogan et al., 2007] and [Saïs et al., 2007] propose logic-based methodologies for merging identifiers of equivalent entities across multiples knowledge sources. While being precise, these techniques do not have a very good recall and are computationally demanding.

The most relevant approach related to ours is the Silk framework. We provide a generic matching process that the user can configure based on their own expertise in order to get more precise results. However, our approach differs by the fact that the matching process is not restricted to linking data between two predefined information sources. On the contrary, our approach makes it possible to link desktop data with an arbitrary number of external data sources. This makes the problem harder since we are generally unaware of the data structure or schema of these data sources.

We therefore need to first find potential entities of interest among a vast number of data sources, then retrieve a partial description of these entities and rely on more complex entity matching algorithms.

This first step of our algorithm can be seen as a blocking pass to reduce the information space before executing complex matching algorithms [Elmagarmid et al., 2007]. The blocking step is implemented on top of a boolean query model for centralised search systems such as Sindice [Tummarello et al., 2007] and on top of the SPARQL query language for specific data sources providing a SPARQL endpoint.

[Nikolov et al., 2012] propose the use of a genetic algorithm to achieve unsupervised discovery of the similarity parameters needed for data linking between two datasets. Similarly to our algorithm, the approach proposed uses a blocking pass using a SPARQL query, to reduce the computation time. As with Silk, the algorithm considers only two datasets to be matched.

5.3 The Process of Finding Web Aliases

The goal of our algorithm and system is to find Web aliases for desktop resources. A Web alias is a Web entity identifier, i.e., URI, that represents the same real-world thing as the desktop entity to which it was matched. To find Web aliases, we use the information available on the desktop, like the contact information from the address book for people, or metadata of music files for songs, albums and artists. We also use the knowledge about the desktop ontologies and the way data is organised and used on the desktop. The desktop data drives the entire process, and is used throughout the steps:

1. Candidate selection — blocking pass
    - Query and identify candidate entities from various Web of Data sources
    - Retrieve data for each candidate from the appropriate source[s].

2. Candidate filtering — scoring based on similarity
    - Compute similarity score based on the data of the entities.
    - Filter the candidates based on the similarity score.

The first step requires identifying a list of candidate entities and obtaining the data available about them. There are two options for this:

- through a small set of sources that we know have the data we need, and can query each of them independently for possible candidates, or
- through a search engine for the Web of Data, like Sindice [Tummarello et al., 2007] which indexes millions of documents containing semantic mark-up.

Each option has use cases where it is more suitable than the other, thus we designed our system to support both. Querying specific sources is preferred for instance, if the data is from a very specific domain, like cancer research, or when we are interested only in results from an organisation's internal repository. Using a search engine is best when the information sources to query are not known a priori. It also has the advantage of covering a large number of information sources with only one query, and of selecting the most relevant data sources and candidates with respect to the query via the search engine ranking system. However, in the case of ambiguous entities, the latter option has the disadvantage of returning too many unrelated results, thus making the entity selection more difficult.

Once a list of candidates is available, we compute a similarity score for each of them with respect to the desktop entity. The desktop data is considered authoritative, and the matching process is driven by it. This has the apparent advantage of control, since the desktop data is more strictly under the control of the user. However, there are cases when the desktop data is simply insufficient, or has uncaught errors due to the user's incomplete information, or the Semantic Desktop's extraction process.

The matching algorithm checks first whether the types of the candidate entities correspond to the type of the desktop entity, and discards the ones that do not. Only then are the data of the entities examined and the properties and corresponding values compared. If required, the algorithm looks at other related entities and their properties. The values of the properties are compared using either exact string matching or string similarity techniques. As mentioned above, the ontologies used on the desktop are different from those used on the Web, thus both type checking and property matching require mappings between the two sets of vocabularies.

After the matching algorithm computes the scores for each of the candidates, the list is filtered for values above a given threshold. References to those Web resources which passed are saved in the desktop repository, as `pimo:hasOtherRepresentation` links from the initial desktop resource.

5.4 Implementing the Process

We implemented the process described above in a desktop daemon that finds Web aliases for desktop entities. It sequentially searches for aliases for all resources that have no alias listed, and for resources that have changed since the last time aliases were determined for them. In the case when a resource is revisited after being modified, the previously found aliases are discarded and new ones are searched for.

The result of the process is the creation of new links on the Semantic Desktop between local resources and their Web aliases. They can be used to enhance the desktop data about the entities, or as entry points to access further information online.

The implementation has two major components, each handling one step of the matching process: A query component that initiates the search and identifies the candidates, and a matching component that filters the candidates based on similarity measures. We next discuss these components in turn.

5.4.1 The Query Component

The query component was designed to be able to use either generic search engines or specific data sources. Therefore, we chose to make the query module plugin-based, thus allowing various new sources to be connected if needed. The query modules are responsible for finding the initial list of candidates, as well as for retrieving the data for each candidate. The maximum number of candidates to retrieve from a data source can be set as a parameter in the configuration. We allow three types of plugins:

SWSE — connect to semantic search engines, through their APIs. We provide a plugin of this type for Sindice.

Sparql — connect to sources that provide a SPARQL endpoint. We provide plugins of this type for DBpedia and the Semantic Web Conference Server.

Custom — connect to other sources, possibly ones that do not expose any data as RDF (e.g., relational databases or third-party APIs like last.fm).

Most major Linked Data providers (e.g. DBpedia) are indexed by Sindice, therefore the Sindice plugin is the only one enabled by default.

In the Sindice module, the initial query, which determines the list of candidates, is constructed using all the value properties of the desktop entity, combined using the boolean conjunction operator "OR". Multiple word terms are also tokenised and the tokens are added to the query. This may result into a longer query which contains duplicate words. This is however due to the fact that we rely on the search engine to interpret the query and rank higher the results that match most of the terms. The search engine (in this case Sindice) algorithm ranks higher results which match more of the strings connected by "OR". Building the query with just the strings without quotes would result in automatic tokenisation by the search engine, which might lead to higher ranking of unsuitable results, especially if the words searched for are common. Thus we chose to prevent the tokenisation by the search engine by using quotes around the full string searched for, but still mimic the default behaviour of the algorithm by doing the tokenisation ourselves, as a fail-safe in case the query yields no results.

For the music album "One Night Only" by the Bee Gees from 1998, the query constructed is:

> *"Bee Gees" OR "One Night Only" OR "1998" OR "Bee" OR "Gees" OR "One" OR "Night" OR "Only"*

5.4.2 The Matching Component

The matching module computes a similarity score for each pair (*desktop entity—web candidate entity*). The way the score is computed depends on a set of parameters:

String matching (SM) — If this parameter is set to `true`, the matching module will use string similarity measures where appropriate. Currently the system supports Monge-Elkan [Monge and Elkan, 1996] and Chapman[2] distances. If the value is set to `false`, the matching module uses exact matching of property values.

Weighted properties (WP) — If `true`, the matching module will use weights for the properties compared, otherwise, all properties contribute equally to the final score.

[2] `http://sourceforge.net/projects/simmetrics/`

Multi-valued properties (MVP) — If `true`, properties that have more than one matching value will contribute to the score proportionally to the number of values.

These parameters are set by default to true. However, the measurements we did for the evaluation (see Section 5.5) show that in some cases they can be disabled without impacting the quality of the result, while requiring less processing; or even make the algorithm perform slightly better when disabled, for some restrictions.

```
{
   "http://www.semanticdesktop.org/ontologies/2007/11/01/pimo#Person": {
      "mapping":[
         "http://xmlns.com/foaf/0.1/Person",
         "http://xmlns.com/foaf/0.1/Agent",
         "http://dbpedia.org/ontology/Person",
         "http://www.w3.org/2000/10/swap/pim/contact#Person",
         "http://rdf.data-vocabulary.org#Person"
      ]
   }
}
```

Listing 5.1: Type mapping for pimo:Person.

```
{
   "http://www.semantcdesktop.org/ontologies/2009/02/19/nmm#performer##http://www.
       semanticdesktop.org/ontologies/2007/03/22/nco#fullname": {
      "mapping": [
         "http://dbpedia.org/property/artist",
         "http://dbpedia.org/ontology/artist##http://xmlns.com/foaf/0.1/name",
         "http://xmlns.com/foaf/0.1/maker##http://xmlns.com/foaf/0.1/name"
      ],
      "approx":"true",
      "thresholds": [
         "MongeElkan:0.7",
         "Chapman:0.8"
      ],
      "weight":"0.7"
   }
}
```

Listing 5.2: Property mapping for nco:fullname of nmm:performer.

The algorithm uses a set of mappings from the desktop ontologies to some of the more popular Web vocabularies, like FOAF. There are two kinds: *type mappings* (see Listing 5.1 for an example) and *property mappings*, each described in a separate file. The property mapping supports paths of properties. For example, Listing 5.2 shows a path composed of the property `dbpedia:artist` and `foaf:name`. The mappings are relatively static configurations of the system. We have created a set of mappings for the

most common ontologies, which can be used out of the box by the end users. Power users can edit the mapping files according to their needs. We envision that as more users find the system useful, more mappings will be created and shared.

The scoring and matching algorithm

The algorithm for computing the score works as follows. Given a pair of entities to be compared e_d and e_w, it first determines the sets of types T_{e_d} and T_{e_w} for each entity, and the set of types $Map[T_{e_d}]$ to which the elements of T_{e_d} are mapped. If no types are matching, i.e., $T_{e_w} \cap Map[T_{e_d}] = \phi$, it gives a score $score(e_w) = 0$, and stops the matching. Otherwise, it continues the process by evaluating the properties.

The evaluation of the properties is driven by the relations and properties of the desktop entity e_d. For each property p_{e_d}, the algorithm retrieves the list of values $V(p_{e_d}) = \{v \ : \ \{e_d \ p_{e_d} \ v\}\}$. Based on the list of property mappings $Map[p_{e_d}]$, it determines the set of values $V(p_{e_w} \cap Map[p_{e_d}])$ that the properties from $Map[p_{e_d}]$ have in common with e_w. If there is no value in common, i.e., $V(p_{e_d}) = \phi$ or $V(p_{e_w} \cap Map[p_{e_d}]) = \phi$, the pair is skipped and nothing is added to the score. Otherwise, it continues the process by measuring the similarity between values.

The evaluation of values is performed using string similarity between each pair of values $(v_d, v_w) \in V(p_{e_d}) \times V(p_{e_w} \cap Map[p_{e_d}])$. The algorithm creates a sparse matrix where the value of a cell contains a string similarity score between 0 and 1. Let $sum_{p_{e_d}}$ be the sum of the best score for each row of the matrix. The final score is computed as follows:

$$score(e_w) = \frac{\sum_{p_{e_d}} (w_{p_{e_d}} * sum_{p_{e_d}})}{\sum_{p_{e_d}} (w_{p_{e_d}} * \left|V(p_{e_d})\right|)}$$

where $w_{p_{e_d}}$ is the weight assigned to a certain property mapping. If the score is above 0.5[3], the entity is accepted as a Web alias for the desktop entity.

5.5 Evaluation

To evaluate our system, we wanted to measure the accuracy of the matches, in a real-world set-up, with real data. We only evaluate the matching component of the system, since the query component is straightforward and its performance depends on external

[3]We found that the threshold 0.5 provided the best results in our experiment.

factors, like the availability of the services used and the network connection. To assess the results of the matching, we created two entity corpora, one with desktop data and one with Web data. On these corpora, we first created a baseline from relevance judgements made by human experts. Then, we ran our entity matching algorithm and we computed precision, mean average precision (MAP), and normalised discounted cumulative gain (NDCG) to measure its performance.

5.5.1 Data Collection

We created two corpora for the evaluation, one containing desktop entities, and one containing possible matching entities from the Web of Data. The Web corpus was obtained by using the query component of the system, with the only active plugin being the Sindice one.

Desktop data entity corpus

The desktop data used in the evaluation was collected from a real, in-use Nepomuk-KDE Semantic Desktop. It was generated by Nepomuk applications, and extracted from the desktop repository.

We restricted the entities selected to three types:

- people — of type `nco:PersonContact`,
- publications — of type `nfo:PaginatedTextDocument`, and
- music albums — `nmo:MusicAlbum`.

From each type we collected fifty distinct resources, resulting in a corpus of 150 seed desktop entities, and other entities related to them. Examples of auxiliary entities are the authors of publications, which may or may not be already in the corpus as contacts, the tracks of the albums and the artists. In total the desktop data corpus has 11,917 triples.

We used information from our desktops, therefore the people are colleagues or other researchers we collaborate with; the publications are related to our research interests, and generally related to semantics and information extraction. The music albums data was gathered from several colleagues, for variety of genres.

The contact data is extracted by Nepomuk from the default KDE address book application, and we made no changes to it. The correct way to use the `nco:PersonContact` resources extracted automatically, is to link each of them to a corresponding `pimo:Person` representing the person who has the contact information. However, the current tools do not make the distinction, therefore we also used the "raw" `nco:PersonContact` resources, for simplicity. The algorithm makes no distinction between types, so it would yield identical results if we had used the "proper" `pimo:Person`.

The information related to music albums is extracted automatically by Nepomuk from the ID3 tags of music files.

For publications we used Sclippy[4], an existing tool described in [Groza et al., 2009a], to perform shallow metadata extraction from files to obtain the title and the authors of the publications, when the metadata of the documents was not set.

Web of Data entity corpus

We used the query module of our system to generate the second corpus, containing Web of Data entities. More precisely, we used the Sindice plugin to retrieve the first twenty results returned by Sindice, for each desktop entity, thus making a total of 3,000 URIs. The queries used in Sindice were constructed as presented in Section 5.4.1, a combination based on the properties of each desktop entity. For each URI we obtained all the triples extracted by Sindice — explicit and implicit. In total this corpus has 1,530,686 triples.

In this dataset we did not explicitly retrieve Sindice data for the auxiliary entities related to the result URIs. We assumed that this data will be available when and if required — in the relevance judgements by experts, and in the matching process by the algorithm.

5.5.2 Relevance Judgements from Experts

We collected the relevance judgements from experts through an online experiment, in which we asked participants to decide if pairs of desktop and Web URIs identify the same real-world object or person. We evaluated in this way all 3,000 pairs from the two corpora. Each pair was judged by three different experts. Eighteen people participated in the experiment, all researchers in the area of Semantic Web.

[4] `http://smile.deri.ie/projects/sclippy`

To simplify the task, we presented the two entities side by side, with all the information which was available about them in the corpora (see Figure 5.1). The desktop entity is shown on the left, and the Web entity on the right. On the Web side we included hyperlinks to the related entities, for further exploration in case the information given was not sufficient for making the decision. For convenience, on the Web side, we have separated and brought to the top the triples which partially matched any of the values from the desktop side.

Figure 5.1: The Web interface of the experiment for collecting relevance judgements.

There were only two decisions possible: *Yes* or *No*, with a *Skip* option, in case of uncertainty. Once a pair was judged or skipped, another one was shown to the participant. The pairs were randomly chosen from the remaining set. To add a gamification element to the experiment, we kept count of the number of pairs judged by each participant, and displayed it on the page. We found that even such a small addition generated ad-hoc competition and made the dull task more interesting.

The results of the experiment show an average agreement and its standard deviation, computed with Fleiss's κ, of 0.638 ± 0.214, over all three types of entities, suggesting substantial agreement between annotators. Table 5.1 shows the Fleiss's κ and its standard deviation σ per type, as well as the average pairwise percent agreement. We observed that for music albums, there was only moderate agreement between annotators, visibly

	κ	σ	Avg
All	0.638	0.214	92.252
People	0.661	0.257	88.2
Publications	0.786	0.127	98.067
Albums	0.442	0.233	90.523

Table 5.1: Inter-annotator agreement measures

lower than the average, while for publications it is visibly higher. We believe the difference is caused by the fact that the data about publications is generated and curated by experts in the field — even more so, as the publications were largely from the domain of Semantic Web — while the music data comes from much more heterogeneous sources.

5.5.3 Quantitative Results

To evaluate the performance of the algorithm, we evaluate each of the matching parameters described in Section 5.4.2, activated either separately or using a combination of them, against a baseline which is the matching framework with all the parameters disabled. In the following, the String Matching parameter is denoted by SM, the Weighted Properties by WP and the Multi-Valued Properties by MVP.

We used the *trec_eval* tool[5] to compute standard information retrieval measures. The precision at k ($P@k$) with k=1,2,3,4,5, mean average precision (MAP) and normalised discounted cumulative gain (NDCG) are reported in Table 5.2 for music albums, Table 5.3 for people and Table 5.4 for publications. We report also the interpolated precision at recall cut-off points when all matching parameters are enabled. The goal for the system is high precision, i.e., achieving a maximum at P@1. Recall is not a target, as it is generally impossible to determine the entire set of correct results available in the Web of Data.

In Table 5.2, we can observe that only the SM parameter is enhancing the results compared to the baseline. The other two parameters do not improve the results at matching certain candidates. Also, in term of MAP and NDCG, the system achieves

[5] `http://trec.nist.gov/trec_eval/`

	MAP	NDCG	P@1	P@2	P@3	P@4	P@5
SM WP MVP	0.2464	0.5117	1	0.625	0.4167	0.3125	0.25
SM WP	0.2464	0.5117	1	0.625	0.4167	0.3125	0.25
SM MVP	0.2464	0.5117	1	0.625	0.4167	0.3125	0.25
WP MVP	0	0	0	0	0	0	0
SM	0.2464	0.5117	1	0.625	0.4167	0.3125	0.25
WP	0	0	0	0	0	0	0
MVP	0	0	0	0	0	0	0
Baseline	0	0	0	0	0	0	0

Table 5.2: Evaluation results for albums, when varying configuration parameters.

the lowest performance on the albums corpus. This can be explained by the fact that the Web resources returned by the query module for albums are mostly e-commerce products, which are not defined as a type of interest, and therefore are rejected by the matching module. However, some of the annotators have considered that the corresponding candidates are indeed the same as the album, while some have disagreed — this is reflected in the agreement measure, as for the albums we have the lowest value for Fleiss's κ. Whether or not such candidates should have been kept by the system is open to discussion and left for future work.

	MAP	NDCG	P@1	P@2	P@3	P@4	P@5
SM WP MVP	0.4212	0.6354	0.9302	0.8953	0.7597	0.6337	0.5442
SM WP	0.4174	0.6321	0.9286	0.8929	0.746	0.6131	0.5286
SM MVP	0.4212	0.6354	0.9302	0.8953	0.7597	0.6337	0.5442
WP MVP	0.2916	0.5338	1	0.8243	0.6036	0.473	0.3838
SM	0.4212	0.6354	0.9302	0.8953	0.7597	0.6337	0.5442
WP	0.2916	0.5338	1	0.8243	0.6036	0.473	0.3838
MVP	0.2877	0.53	1	0.8243	0.6036	0.4662	0.3784
Baseline	0.2877	0.53	1	0.8243	0.6036	0.4662	0.3784

Table 5.3: Evaluation results for people, when varying configuration parameters.

In Table 5.3, we can observe that the baseline, the WP and the MVP parameters are each able to match good candidates with high precision at P@1, with WP providing slightly better MAP and NDCG. However, the system does not get significant advantage by combining them. The SM parameter alone provides slightly lower precision at P@1 but significantly better MAP and NDCG. By combining the three parameters, the system does not get significant advantage and it seems that using SM prevails.

	MAP	NDCG	P@1	P@2	P@3	P@4	P@5
SM WP MVP	0.7773	0.8651	1	0.625	0.4167	0.3125	0.25
SM WP	0.8032	0.8609	0.9062	0.5781	0.3958	0.3047	0.2438
SM MVP	0.7175	0.7986	0.9231	0.5769	0.3846	0.2885	0.2308
WP MVP	1	1	1	0.5	0.3333	0.25	0.2
SM	0.7265	0.7883	0.8235	0.5294	0.3627	0.2868	0.2294
WP	0.6893	0.7347	1	0.55	0.3667	0.275	0.22
MVP	0	0	0	0	0	0	0
Baseline	0.7175	0.7588	1	0.5455	0.3636	0.2727	0.2182

Table 5.4: Evaluation results for publications, when varying configuration parameters.

In Table 5.4, the baseline provides good results from the start for publications. The system is not able to return any candidates when MVP alone is activated. However, when WP and MVP are combined, the system achieves much better results (in term of MAP and NDCG) than the baseline or than the WP parameter alone. When the system combines SM with the two previous ones, the system achieves a lower MAP and NDCG but an improved precision with a larger cut-off rank. While on the two previous types of entities, the SM parameter seemed to be the most important matching feature, this corpus shows that the WP and MVP are important matching features in certain cases.

Overall, the results are satisfying for our use cases where high precision prevails over recall. However, given the results shown in Figure 5.2, we can see that the system could be configured to return more than one entity in order to achieve better recall while keeping good precision. It might prove useful to implement a semi-automatic system which presents the top n candidates to the user for manual selection.

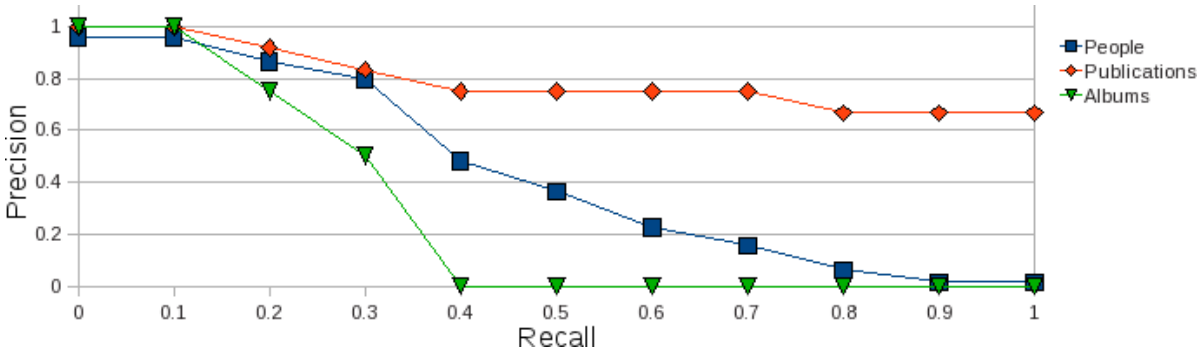

Figure 5.2: Interpolated precision at recall cut-off points.

5.5.4 Performance

To determine the performance, we measured the time spent on each step of the algorithm. We note that these results come from a prototype implementation, still to be subject to technical optimisations. Table 5.5 shows the average times overall, and for each resource type separately, when all three parameters (SM, WP, MVP) are active. We find only small variations in the measurements when the parameter values are changed. We do not consider the time spent on retrieving data from Sindice, as this depends on external factors, like network speed and server availability.

	Overall	People	Publications	Albums
Pair total	375.04	52.19	977.87	53.18
Types check	0.23	0.26	0.21	0.23
Per property check	6.66	0.92	13.2	22.06
All properties	2026.22	7.17	5478.87	1963.88

Table 5.5: Time performance (milliseconds).

The checking of types is the only value that on average does not depend on the type of resource, as it must be performed for all pairs. The time spent in average per property check is low, but it varies by type, and by the complexity of the properties (e.g. it takes longer if several resources in the graph must be traversed, for long property paths like the name of the artist of an album). The "All properties" row shows the average time required for checking all the properties of an entity, and the computation of the final

score[6]. These values depend on the type of resources as well, and on the complexity of the resource graph. We found that longer times correspond to very big graphs for online entities, which must be loaded for checking even if in most cases are not found to represent valid candidates[7].

5.6 Discussion

The scope of the system presented here is limited to finding Web of Data aliases for desktop resources. We leave the use of the aliases found to future work, but the use cases include personalised desktop services like those described in Section 5.1 and enhancement of desktop information from online sources like the one described in [Groza et al., 2009a]. We plan to develop a semi-automatic service that retrieves information from the Web aliases and updates the local resources, while saving provenance information for the imported data and allowing synchronisation when the Web data changes.

Existing Web applications already provide similar services via specific APIs (e.g., last.fm). However this is not the goal of our work. Instead, we wish to leverage information across all public information sources accessible on the Web of Data. In addition, such third-party APIs are seen as an additional information sources on the Web, and are supported by our system.

Within the system, we make use of existing semantic technologies, including semantic search engines such as Sindice. In the process of determining the aliases we focus on selecting the most appropriate URI from the list of candidates returned by the search engine. In this case, the issue of which data sources to trust is left to the search engine, which usually employs advanced techniques [Delbru et al., 2010] for making the decision. This is however not a requirement we impose on the users, who can choose to query other trusted data sources suitable for their use case.

The system we presented is automatic, as no user interaction is required for it to work. Once set up it will find and save aliases to desktop resources. Although the mappings were created manually, they are part of the system and do not need to be modified by end users. Power users can however tweak the settings to fit their specific needs by

[6]The "All properties" row has values higher that the "Pair total" row because the average time is computed only for those pairs that passed the type check, thus fewer, but with longer computation times.

[7]e.g., the graph for `http://webconf.rkbexplorer.com/models/iswc-aswc-2007-complete.rdf`

enabling or disabling modules, changing threshold values or modifying mappings. While new mappings can only be created manually, expert users can take advantage of the openness resulting from the use of SPARQL-based search mechanism and update the mappings as they need, or create new ones. We envision for the future, a way of allowing power users to publish their own mappings and let other users install new mappings in a way similar to installing add-ons to Web browsers.

5.7 Conclusion

In this chapter, we presented a framework to automatically link entities from the Semantic Desktop to the Web of Data. This is the logical next step after the creation of semantic data on the desktop, presented in the previous chapter.

The framework uses existing technologies such as semantic search engines and public SPARQL endpoints for retrieving a set of candidates. Each candidate is then evaluated more precisely based on a collection of matching components using string matching, heuristics and rule-based mechanisms. We evaluate the system qualitatively, using real-world data retrieved from a Nepomuk-KDE Semantic Desktop and the Sindice search engine. The evaluation is based on relevance judgements from a group of experts. We show that the system in its current form provides satisfactory results in term of precision for automatic linking of entities.

Once the two networks of linked data are connected, we can build semantic applications and services using personal data from the desktop and enriched with information from the Web of Data. In the next chapter we present one such application.

Chapter 6

Transforming Semantic Notes into Semantic Blog Posts

Based on "Linking Semantic Personal Notes" [Dragan et al., 2010] published at the Workshop on Knowledge Injection into and Extraction from Linked Data (KIELD 2010 co-located with EKAW 2010)

In the previous two chapters we showed how Semantic Web technologies are available both on the desktop and of course on the Web, and how the two networks of linked data can be connected. SemNotes, a semantic note-taking tool for the Semantic Desktop, described in Chapter 4, shows how new data can be created and seamlessly integrated with existing semantic information from the desktop. Chapter 5 describes an algorithm for automatically connecting desktop resources to their Web aliases, bridging the gap between the desktop and the Web in regards of semantic data, opening up the possibility of augmenting desktop data and building personalised desktop services.

In this chapter we present a use case, and a proof of concept system which builds on our previous work in order to support easy publishing and sharing of semantic personal notes taken with SemNotes as Linked Data. Our approach can be used to publish any kind of information from the desktop to the Web, enabling integration of small chunks of personal knowledge into the Web of Data. This use case illustrates a solution to the second research question **Q 2.** from Section 1.2, *How to expand the scope of the Semantic Desktop into the realm of the Web of Data, to enhance the user experience and benefit?*, more specifically, to the second and third sub-questions **Q 2.2.** — *How to use the Web*

information found related to a desktop resource?, and **Q 2.3.** — *How to make desktop data available online safely?*

6.1 Introduction

Semantic Web technologies are deployed in various domains and applications, both on the desktop and on the Web. While these two domains share compatible representation models (RDF(S)/OWL), there is still a gap between data from the Web and the desktop. We showed in the previous chapter how finding Web aliases for Semantic Desktop resources enables us to bridge this gap. This solution is however unidirectional — it only solves connection from the desktop to the Web, and not the reverse, from the Web to the desktop. It can be argued that accessing desktop information from the Web is not necessary, or not safe, or that the risks exceed the benefits. However, sharing desktop information online has many possible use cases, and does not necessarily involve making private information public.

One such use case of sharing desktop information and realising better integration of personal data with online data, is described in this chapter. We present an approach for publishing personal notes from the desktop (using SemNotes and Semantic Desktop technologies) to the Web of Data (using the Linked Data principles). Our goal is to publish this data online without losing the personal context established on the desktop through the links from the notes and the related desktop resources. Our approach consists of two main steps:

1. preparing the desktop data for sharing,
2. publishing it online.

In addition, it requires two prerequisite steps, which are supported by previous work:

- the note-taking process and annotation of the note (adding the context) which was presented in Chapter 4, and
- the identification of Web URIs which represent the same real-world thing as the desktop resources that belong to the context of a note, which was presented in Chapter 5.

Our contribution consists of the process for publishing personal information from the desktop to the Web following the Linked Data principles [Berners-Lee, 2006a], and a

system implementation that allows sharing of semantic personal notes as semantic blog posts, interlinked with existing information from the Web of Data. The publication process must follow the Linked Data principles, while at the same time protecting sensitive private data from being shared unwillingly, and maintaining the meaningful relations in the process.

The semantic notes taken with SemNotes constitute the personal information that we want to publish as Linked Data. They are semantically linked to local resources mentioned in their content, like people, projects or other notes. If the notes were directly published online, the relations that enrich them would be lost and the links they contain would be broken, because they point to local URIs not available outside of the desktop. The same objects often exist within the Web of Data, therefore the note links to them can be recreated in the publishing process. Before the note is published, our system searches for existing public Web aliases for the local resources. They are then used to substitute the local private data referenced in the text — the links are changed to point to the Web resource. In this way, the published links are valid, the meaning of the link is preserved (as referring to the same entity) and the private information is protected. The system then publishes the notes online as blog posts taking care of the transformations required from desktop ontologies to Web ones, according to the mappings devised in Chapter 5.

In line with the two-step process mentioned above, our system has two modules, one working on the desktop and the second one online. First, the desktop part handles the preparation of the notes for publishing, using the Web aliases identified by the service described in the previous chapter, substituting them for the local ones, as well as the transformation required from desktop ontologies to Web ones. This module is invisible to the user, except for the *Publish* button shown in the user interface of SemNotes. Then, the Web part publishes the transformed notes as Linked Data in accordance with the Linked Data publishing principles.

The solutions provided by most existing systems fall into two categories: (i) desktop applications that involve publishing the actual local resource information together with the note, or (ii) online applications that do not have access to desktop data relevant to the user. The first category, represented by tools like SemiBlog [Möller et al., 2005] or SemBlog [Takeda and Ohmukai, 2005], is not optimal when dealing with resources that contain sensitive private information. Services like Zemanta[1], from the second category,

[1] `http://www.zemanta.com`

have access to various online resources, but not to the personal context of the user. Also they miss the personal touch of desktop-based applications.

6.2 From Note-Taking to Weblogging

Two characteristics of blog posts which are relevant to this use case are:

- their topics are of interest to the author and thus are very likely to have references to things also present on the desktop (e.g. people, events);
- the context consists of the references made in their content, such as places, projects, or other blog posts.

However, not all blog posts start by being a blog post. Some are just ideas or impressions jotted down for later, in one's preferred desktop note-taking application. Nevertheless, some of these notes do become posts after polishing and refining, especially as some notes might require further investigation before they can be published (e.g. when writing on a technical subject). Examples include notes taken at presentations, ideas on topics of interest written down at times when blogging is not possible (attending a meeting or lack of Internet connection).

Tools from the Semantic Desktop, like SemNotes, provide means to enhance these notes locally, by interlinking them with other desktop data — the contacts in the address book, the events from the calendar application, the projects worked on, the music listened to. For example, a note about an upcoming concert can be linked to the performing artist which is in turn linked to the music files of that artist and pictures from earlier shows stored in a desktop photo application. Such annotations give context to the note and should be preserved when the note is published as a blog post on the Web, since it enables serendipitous browsing and information discovery, through the relevant additional links they contain.

Currently, personal notes, even the ones semantically enriched using Semantic Desktop applications, must be published as blog posts by being manually copied into a blogging tool. In this way, any additional semantic information available on the desktop is lost or, if copied, leads to broken references as they point to the local resources which are not accessible outside of the desktop. The note-taking to publishing process is sometimes cut short by using the drafting functionality offered by some systems like WordPress or Blogger, so that users can directly take the notes in the blogging tool,

usually online, thus replacing the desktop note-taking application completely. However, using online tools deprives the user from having the personal context added to the blog post, since desktop information cannot be easily integrated in Web-based services.

In order to enable a better transition from personal notes to blog posts, or simply to Web-based information available to others (for example, meeting notes published in a company intranet or lecture notes shared between students of the same class), we defined a list of requirements that a system for publishing semantic personal data online should fulfil:

R1. Publish the complete desktop data on the Web without losing any relevant information, including metadata and context (e.g. tags, relations, identifiers);

R2. Protect any machine readable and private data that might be unwillingly included in the context being transferred[2];

R3. Publish the note according to the Linked Data principles and describe it using popular ontologies;

R4. Enable object-centred sociality by establishing connections between data published by different users.

6.3 Overview of the Process and Prerequisite Steps

We propose an approach that enables the publishing and sharing of personal notes by extending the functionality provided by SemNotes, and by using the Web of Data aliases found for desktop resources. The process consists of two steps:

1. transformation, and
2. publication.

In the first step, the note is transformed locally for publication, and private local data is replaced with public references. In the second step, the transformed note is published online.

The system requires a dedicated server, where the resources referenced and the tags assigned are shared between the notes of all users, to add a social aspect. Also, rather than trying to determine which of the many possible Web aliases found for a desktop

[2]The protected data does not include the actual text to be shared as it is a conscious decision to publish it, taken explicitly by the user

resource is authoritative, we let the server create equivalent public resources and connect all the aliases to them. We also used the server for publishing the notes, although this can be replaced by any other blogging platform.

The first prerequisite step — note-taking and annotation of the note with the relevant desktop resources — must be performed before any actual sharing of notes can be done. The annotation is done semi-automatically and is an existing feature in SemNotes.

The second prerequisite step consists in finding Web resource for each of the desktop entity linked to the note that is about to be published. This step is executed by a desktop service which implements the two-part algorithm detailed in the previous chapter, the blocking pass using Sindice to retrieve possible candidates and filtering the list by the score returned by the matching algorithm. The local service has access to, and uses all the information available on the desktop about a resource to identify only exact matches for it. The aliases which are found for the desktop resources are saved by creating a `pimo:hasOtherRepresentation` link in the local repository between the desktop resource and the Web one.

6.4 System Implementation Details

Based on the process described above, the system is divided between its local part and its remote part, as shown in Figure 6.1. The local part handles local *private* data, while the remote one handles online *public* data. The separation between them extends over three layers: ontology, data and application. On the *ontology* layer, the NEPOMUK desktop ontologies are used locally, while popular Web vocabularies are used on the server-side. These ontologies are used to describe the *data* exchanged between the applications. Desktop data is stored in the local NEPOMUK repository, while Web data is distributed. Finally, on the *application* layer, the local component is an extension to SemNotes that provides publishing functionality for notes, and the remote component is a server that hosts and publishes online the notes received.

The first step of the process — transformation — is executed on the local side, by an extension of the SemNotes application. Then, the publication step is done by the server, which receives information from the desktop and publishes the note, as we will describe next. The two application components, the communication between them, and the data translation process are described in detail below.

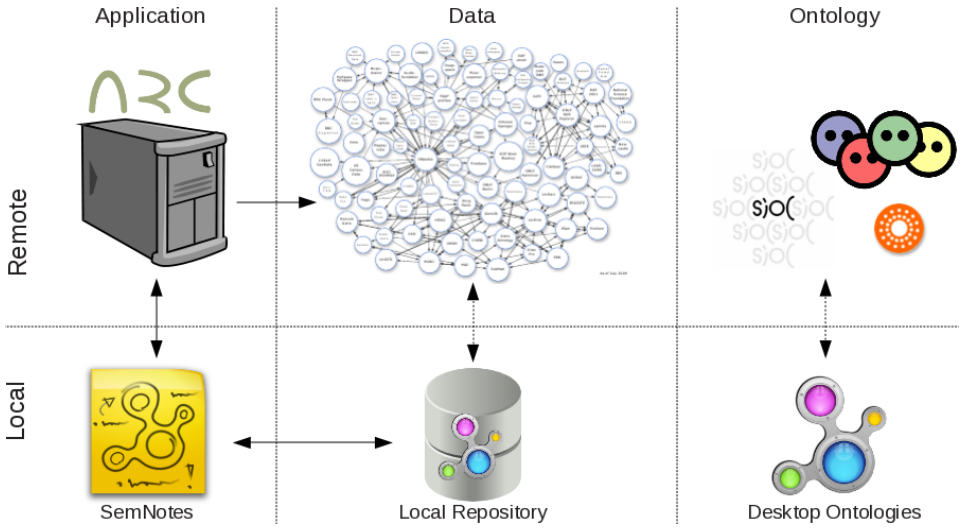

Figure 6.1: Overview of the semantic publishing system.

6.4.1 Ontologies

Although both the Semantic Desktop and the Semantic Web use the same representation languages, i.e. RDF(S)/OWL, they use different vocabularies to describe their data. This vocabulary gap makes data integration difficult.

The NEPOMUK project defines a set of desktop ontologies to describe its data. SemNotes uses these ontologies to represents personal notes as instances of `pimo:Note` and are linked to the `pimo:Things` they mention by the relation `pimo:isRelated`.

When a desktop resource is found to represent the same real world entity as a Web resource, the relation is stored on the desktop as `pimo:hasOtherRepresentation`. This property is recommended by the PIMO specification as desktop equivalent to the `owl:sameAs` relation, although without the formal semantics that the latter provides. We also use the property `pimo:hasOtherRepresentation` to store the remote URL of a note when it is published. If the note changes on the desktop after publication, the property is replaced with `pimo:hasDeprecatedRepresentation`.

While well-suited to represent desktop information, these ontologies are not used, so far, on the Web. However, numerous vocabularies have emerged for describing semantic data published online. Such ontologies have now been widely adopted and are recommended as best practices when publishing data on the Web [Bizer et al., 2007].

Consequently, while representing similar objects, the two sets of vocabularies must be aligned so that on the one hand, desktop information can be moved to the Web and

understood by usual Semantic Web applications that rely on the aforementioned vocabularies, and on the other hand, Web information could be understood and imported by Semantic Desktop applications. In order to enable interoperability between the desktop and the Web, we defined mappings between the sets of ontologies. The mappings create appropriate subclasses or subproperties of the relevant concepts from the chosen vocabularies.

SIOC is probably the most widely used vocabulary for interlinking social media within the Web of Data. There are already many tools for creating and using SIOC data [Bojars et al., 2008]. This is why we chose to represent the `pimo:Note`s as `sioc:Post`s when they are published online with our system. The rest of the desktop resources are also transformed into concepts from the vocabularies listed above (see Table 6.1), the mappings being published at `http://rdfs.org/sioc/nepomuk`. The note's properties,

Class	Subclass of
`pimo:Note`	`sioc:Post`
`nao:Tag`	`sioct:Tag`
`pimo:Person`	`foaf:Person`
`pimo:Project`	`doap:Project`
`pimo:Event`	`ical:Vevent`

Table 6.1: Sample of the mapping between classes.

like title, creation and last modification time, are translated to the appropriate Dublin Core properties: `dcterm:created`, `dcterms:modified` and `dcterms:title`. The tags associated locally with the notes are transformed into `sioct:Tag`s associated with the published post using the `sioc:topic` property. Table 6.2 lists the proposed mappings for properties[3].

[3] Although `nao:lastModified` and `dcterms:modified` do not have the same semantics, defining subproperty relations between them is acceptable.

Property	Subproperty of
nao:prefLabel	rdfs:label
nao:created	dcterms:created
nao:lastModified	dcterms:modified
nao:hasTag	sioc:topic
pimo:isRelated	sioc:related_to

Table 6.2: Sample of the mapping between properties.

6.4.2 Server Schema

In order to publish the resources with a consistent URI scheme, we defined patterns for naming the objects published from the desktop on the Web. In the schema definition, we apply several Linked Data patterns described in [Dodds and Davis, 2010]:

- patterned URIs — for all the entities, to make them more human readable;
- proxy URIs — for the server URIs, to group multiple Web aliases;
- equivalence links — for the resources related to the notes, to unify various sources;
- natural keys — in the tag URIs.

For each note the server generates a new unique identifier `id` which is used to create the note's URI in the form: `http://notes.server/note/id`.

According to the proxy URIs identifier creation pattern, we generate new URIs for the resources related to the notes. This ensures that the publishing process is consistent and avoids having to choose among several Web aliases a resource could have. Like the notes, each resource has a unique identifier on the server, which is used to create the resource URI according to the following format: `http://notes.server/resource/id`. Resources are shared by all the notes that link to them, which increases the interlinking and the consistency of the data. For each resource, the server keeps internally a list of Web aliases using `owl:sameAs` links.

Tags are considered a particular type of resources, and are also shared on the server. The specific format for the URI: `http://notes.server/tag/label`, differentiates them from regular resources. The label of the tag acts as a unique identifier, and is case sensitive. They are created on the fly, and are persisted when they are used for the first time.

6.4.3 Transformation of the Notes for Sharing

The first step of the process consists of preparing the note for publishing. In this phase all the relevant information about the note is included in the content, specifically the title, creation and last modification time, the tags and the referenced resources. This transformation is necessary, so that less information, specifically only the HTML content of the note is sent to the server, and not the entire RDF graph describing the note. Although the content is already stored as HTML, to include all the metadata about the note, it has to be enriched with RDFa before being posted to the server.

The preparation step is done on the desktop side, by the added extension to SemNotes. To include the referenced resources in RDFa, we need to know their server URIs. Therefore the application needs to communicate with the server to retrieve several URIs:

- the URI for the new note to be published, and

- the server URI for each resource referenced by the note.

In case the note has already been published, the user can overwrite the old post (on the Web) or create a new one. Depending on this choice, a new URI is requested from the server, or the existing one is used (that was saved in the local repository when the note was previously published).

The referenced resources are shared by all the published notes, therefore the server must create the URI for a resource only if it has not been created before. To decide whether a local resource already has a server URI created, the list of Web aliases found for it on the desktop in the second prerequisite step of the process, is sent to the server (see JSON data in Listing 6.1). If a resource with a matching type and an overlapping list of aliases exists, the server reuses it, otherwise it creates a new one and saves the information about it in its own RDF repository. On the server, the URI aliases are saved as `owl:sameAs` as it is customary for Linked Data. The server URIs for the note and the resources are also stored on the desktop for reuse, as `pimo:hasOtherRepresentation`.

```
{
   "id" : "",
   "resources": [
   {
      "id": "nepomuk:/res/bfcdcd1a-4898-492f-940b-4cc4c67799a7",
      "type": "mo:MusicArtist",
      "uris": [
         "http://dbpedia.org/resource/Scorpions_(band)",
         "http://musicbrainz.org/artist/c3cceeed-3332-4cf0-8c4c-bbde425147b6"
      ]
   }
   ]
}
```

Listing 6.1: JSON formatted message sent to the server.

```
{
   "note":{
      "uri":"http://notes.server/note/4baccab834e20",
      "resources":[
         {
            "local":"nepomuk:/res/bfcdcd1a-4898-492f-940b-4cc4c67799a7",
            "uri":"http://notes.server/resource/4bacca84ca8bb"
         }
      ]
   }
}
```

Listing 6.2: Server reply with the server URIs for the resource aliases sent.

In the case when no Web aliases are found for a desktop resource that is related to a note, and thus the list of aliases sent to the server is empty, a new resource is created on the server with the specified type, but without any information attached to it. The server URI is saved on the desktop as `pimo:hasOtherRepresentation` of the resource, and will be available for reuse when other notes related to the same object are published by the same user. However, this resource will not be shared between notes published by different users. If at a later stage a Web alias is found for the desktop resource, it will be added to the resource already created on the server, thus enabling it to be linked to by multiple users.

The communication between SemNotes and the server is done with a single REST call, in order to minimise network delays. The reply contains the newly created URI for the note, if one was required, as well as a list of server URIs for the resources (see

JSON data in Listing 6.2). The communication between the desktop side and the server is shown in Figure 6.2.

Using the information received from the server, the note content is enriched with RDFa. The metadata about the note, like type, creation and last modification times and the tags, is added in `meta` tags in the `head` of the HTML page. RDFa is added to the `title` tag and in the `body`, to the links. Listing 6.3 shows the content of a note prepared for publishing.

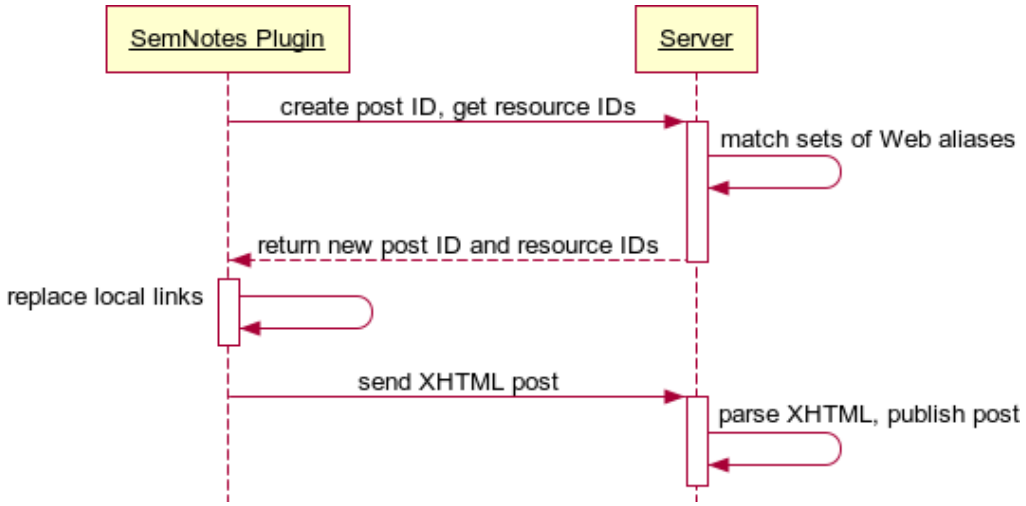

Figure 6.2: Sequence diagram for the communication with the server.

```
<!DOCTYPE html PUBLIC '-//W3C//DTD XHTML RDFa 1.0//EN'
                      'http://www.w3.org/MarkUp/DTD/xhtml-rdfa-1.dtd'>
<html about="http://notes.server/note/4baccab834e20">
    <head>
        <meta content="sioc:Post" property="rdf:type"/>
        <meta rel="sioc:topic" href="http://notes.server/tag/concert"/>
        <title property="dc:title">concert sunday</title>
    </head>
    <body>
        <a rel="sioc:is_related"
             href="http://notes.server/resource/4bacca84ca8bb">scorpions</a> concert on
                  sunday was great ...
    </body>
</html>
```

Listing 6.3: RDFa-annotated XHTML content of note.

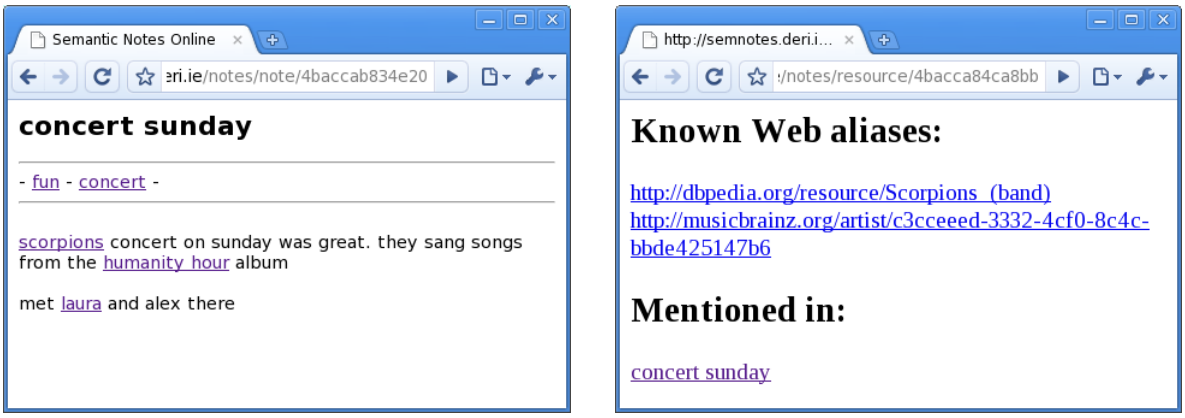

Figure 6.3: Online view of a note (i) and a resource (ii).

6.4.4 Publication Step

After the preparation step, which takes place on the desktop side, the RDFa enriched content is sent to the server via another REST call. The publication step of the process only handles public data. When the content is received it is parsed and the server extracts the contained RDF triples and stores them in its repository. The content (as it is received) is also stored.

The server implementation uses ARC2[4], as it provides out of the box RDFa parsing and an RDF repository. It is easily deployable due its minimal setup requirements (a PHP enabled Web server and a MySQL database), thus making our system easily deployable as well.

All server URIs are dereferenceable, as required by the Linked Data principles. For notes, the URI redirects to the RDFa annotated HTML page containing the note itself (as shown in Figure 6.3 (i)), the URI of the note being the URL of this page. For the linked resources, the URI is also dereferenceable and provides RDFa information about itself, linking to the known existing Web aliases of the same resource. The description also includes a list of backlinks to all the notes that reference the resource (see Figure 6.3 (ii)). The page for a tag will contain backlinks to all the notes tagged with it.

The RDFa annotated page for the note is generated on the user's desktop by the SemNotes plugin, as we have seen in the previous step, while the page describing each resource and tag is generated on the fly, by the server, when the URI is requested.

[4] `http://arc.semsol.org`

6.5 Conformance with the Initial Requirements

When establishing the specifications of the framework, we identified four main requirements (Section 6.2). Our proposed implementation conforms with them as follows.

***R1**: Publish the complete desktop data on the Web without losing any relevant information, including metadata and context (e.g. tags, relations, identifiers).*
By translating existing desktop data surrounding the note to RDF and putting it online, available as RDFa, the entire information available on the desktop side is made available on the Web for further reuse. In addition, all information from the original note-taking tool, including title, tags, etc. is publicly made available on the Web.

***R2**: Protect any machine readable and private data that might be unwillingly included in the context being transferred;.*
By replacing the private desktop data with equivalent public Web data, we protect the former. On the desktop there is a lot of private information stored about resources, like the email address or telephone number for people, or the list of attendees of an event. When the person or event is linked to a note that is afterwards published online, such private information is not exported, because the reference to the local resource is replaced by a reference to already public Web data representing the same thing. In this manner, the context of the note being published is preserved, but the private details are not exposed.

***R3**: Publish the note according to the Linked Data principles and describe it using popular ontologies.*
Our system publishes notes on the Web using the Linked Data principles. Each note and connected resource, has its own URI, which is made dereferenceable, while distinguishing information resources and non-information resources. In addition, while original desktop data is provided using "desktop ontologies", the published information is made available using FOAF, SIOC, Dublin Core, etc. and the mappings have been validated through Vapour[5].

***R4**: Enable object-centred sociality by establishing connections between data published by different users.*
Since resources and tags are shared between users, notes can be browsed serendipitously through shared connections, or tags. This enables "object-centred sociality" [Knorr-Cetina, 1997], since people can interact around these shared tags and topics,

[5] `http://vapour.sourceforge.net/`

such as projects or people that they have in common. Depending on the destination of the publishing process, there is the possibility of having private notes, yet still accessible online by registered users only.

6.6 Related Work

Semantic blogging was introduced by [Cayzer and Shabajee, 2003], and since then has received much interest. Later [Karger and Quan, 2004] described semantic blogging in the context of the Semantic Web with Haystack. So far, existing systems for semantic blogging fall into two categories:

- desktop applications that involve publishing the actual local resource information together with the blog post, or
- online applications that do not have access to desktop data relevant to the user.

The tools in the first category, like SemBlog [Takeda and Ohmukai, 2005] and SemiBlog [Möller et al., 2005], have the advantage that users have better access to the relevant data from the desktop. However, both tools require that the resources that contain sensitive private information are published together with the blog posts, which might lead to privacy issues. The SemBlog project allows users to add data from personal ontologies to their blogs. SemiBlog allows integration of personal data in the posts by drag and drop from various desktop applications like the address book. SemBlog and SemiBlog are used for exchange of personal information in the blog posts, which differs from our approach of using already published Web data as to protect the privacy of the personal information. SemiBlog's process implies manually adding the metadata, while our approach relies on automatic export. Both tools comply with our first requirement, but not with the last three.

Online services like BlogAccord [Cayzer, 2006] for music information, and Zemanta[6] blogging assistant, belong to the second category. They have access to various online resources to create the context of a blog post and enhance the blogging experience, but they do not use the personal context of the user.

[6] `http://www.zemanta.com`

6.7 Conclusion

In this chapter we presented an approach for publishing semantic information from the desktop as Linked Data on the Web. The approach is realised by a system which takes semantic notes from the desktop and makes them part of the Web of Data, as semantic blog posts. The goal of the system is to provide a way for publishing and sharing complete information by preserving the personal context of the notes without compromising privacy. While the use case presented is focused on notes and semantic blogging, the same approach can be applied to publishing of any interlinked information.

We defined a publishing process that comprises of two steps:

1. preparation — the note is transformed into a SIOC-based Web representation; and
2. publication and sharing — the note is published online following the Linked Data principles.

In addition, we provided an implementation of the process, and tested it against a set of requirements regarding publishing personal content from the desktop to the Web as Linked Data.

While we do not authentication issues in this current release, we are considering SW-compliant systems such as FOAF+SSL [Story et al., 2009] for future versions of the system. We also plan to develop additional integration modules for publishing platforms, e.g. Drupal, WordPress, and to investigate approaches for authentication and notification tailored towards the people mentioned in the published data.

Chapter 7

Additional Extensions and Applications

In this chapter we describe several applications developed in our SmILE[1] group at DERI, and to which we contributed. They are extensions to the work presented in the preceding pages, providing or serving as test use cases and validation scenarios.

We start with a list of extensions to SemNotes' functionality, which employ Natural Language Processing techniques. They focus on specific use cases of note-taking, like ontology engineering, meeting minutes or status reports; as well as text analytics tailored to short text.

The work presented in Chapter 5 was initially motivated by the increasingly difficult task of managing publications on the desktop and on the Web, and the need of connecting the information available in both worlds. *Sclippy* is a tool created to avail of the rich information available on the Web about publications, and bring it to the user's desktop. The algorithm described in Chapter 5 is a generalisation of the algorithms used in Sclippy[2], extending the scope from publications to any type of desktop resources.

We continue by presenting Konduit, a desktop-based platform for visual scripting with RDF data. Based on the idea of the Semantic Desktop, non-technical users can create, manipulate and mash-up RDF data with Konduit, and thus generate simple applications or workflows, which are aimed to simplify their everyday work by automating repetitive tasks. The platform allows to combine data from both the Web and the desktop and integrate it with existing desktop functionality.

[1] `http://smile.deri.ie`
[2] `http://smile.deri.ie/projects/sclippy`

7.1 Natural Language Extensions for SemNotes

During its development phases, SemNotes has served as a sandbox environment and testing bench for several technologies and prototypes developed within our group. Three of the most notable extensions are related to Natural Language Processing (NLP) of the notes, to add functionality. These include:

- a text analytics extension for keyphrase extraction [Schutz, 2008]
- two controlled language extensions for semantic annotation of meeting minutes and status reports [Davis et al., 2008, Davis et al., 2009]

7.1.1 Keyphrase Extraction

Tagging is one of the most used basic features offered by Nepomuk-KDE and by SemNotes. Tags are important for categorisation and organisation of resources, especially for personal notes. Tagging requires users to make a decision on what is relevant about the resource being tagged and how they might reuse the annotation in the future — i.e. is the tag expressive enough to be later user for filtering the information and finding the required resource. To support users in their choice of tags, and partially automate it, we collaborated on a keyphrase extraction extension to SemNotes. It is based on the TextAnalytics service developed by [Schutz, 2008].

Keyphrase extraction is just one of the NLP functionalities provided by the TextAnalytics desktop service, alongside information extraction and speech act detection. The service can be accessed by all desktop applications and is available also as an Eclipse plugin for the Java implementation of Nepomuk.

Standard information extraction tools usually extract information which is already known — like detecting resources in the text. This is the norm in SemNotes as well, where only known existing desktop resources are suggested as annotations of notes, even when other resources might be referred to, but are unknown to the system. The TextAnalytics service is different and complementary, by extracting terms and instances from text, which are not already known to the system. The new terms can become new desktop resources, once identified.

We have developed an extension to SemNotes, which uses the service to extract keyphrases from the text of the personal notes and suggest them to the user as tags.

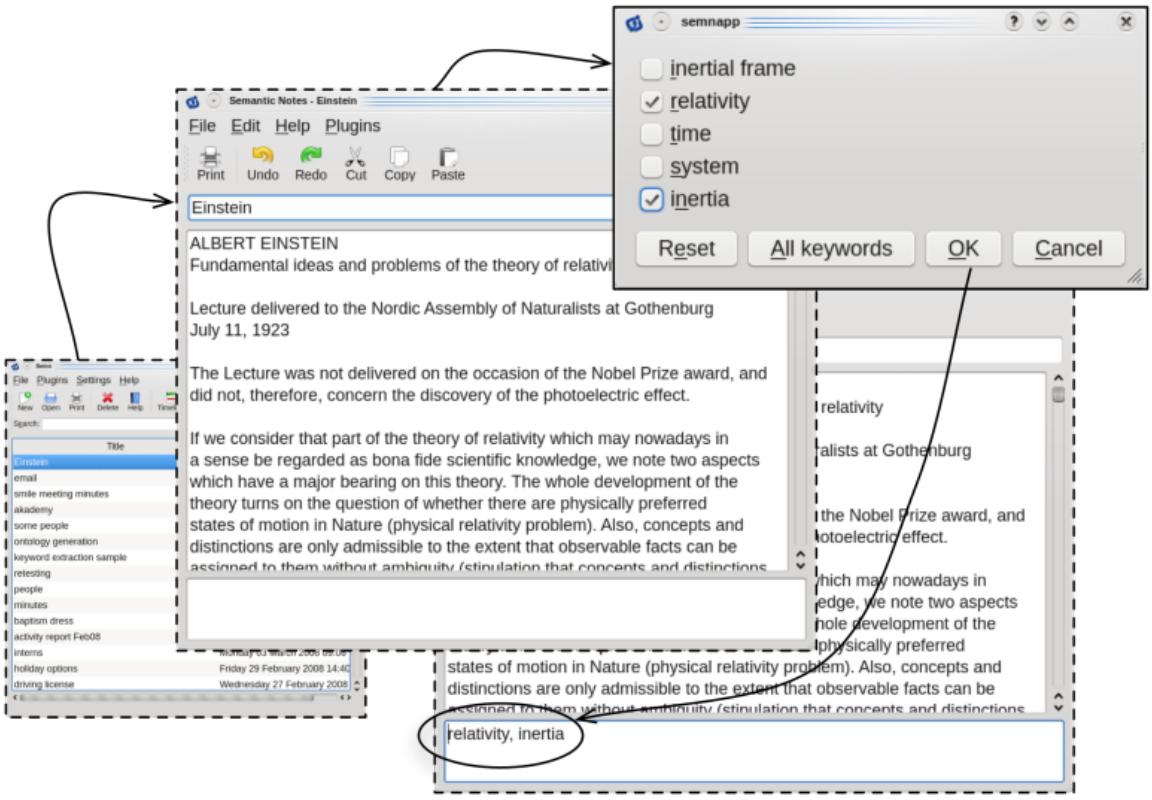

Figure 7.1: User interface to keyphrase extraction in SemNotes.

The text of personal notes can be small, thus adding difficulty to the extraction task. [Schutz, 2008] details the way this challenge is tackled in the service. In SemNotes, the service can be called through the menu, and once the note is parsed the relevant terms found are presented to the user as tag suggestions. Figure 7.1, taken from [Schutz, 2008], shows a screenshot of the interface in an older version of SemNotes. If the user accepts any of the suggestions, they become tag resources and are stored in Nepomuk in the usual way.

7.1.2 Controlled Language Extensions

Controlled Natural Languages are well-defined subsets of a natural language with a restricted grammar and lexicon [Schwitter, 2005], in order to reduce ambiguity and complexity. In his Ph.D. thesis [Davis, 2012], researches the use of Controlled Languages for

ontology engineering and semantic annotation. We used SemNotes for prototype testing of both directions, while also providing a feasible use case for the services.

SemNotes, as a note-taking application, is well suited for Controlled Language use, since the main type of input from the users is textual. As a Semantic Desktop application it is also particularly well suited to ontology engineering due to the direct access provided by the framework to the underlying ontologies and desktop resources. As an annotation tool it is also a good testing ground for the semantic annotation using Controlled Language.

Round Trip Ontology Authoring

Controlled Natural Language proved efficient for creating ontologies in a user friendly way, for naive users who do not wish or do not need to learn how to use full-fledged ontology editors, nor want to dive into the complexities of ontology engineering. However simplified, Controlled Languages still require that the users be familiar with the vocabulary and syntax rules in order to use the language properly. The Round-trip Ontology Authoring environment described in [Davis et al., 2008] aims to reduce the learning curve required to use Controlled Languages for ontology authoring. It combines and builds on the CLOnE controlled language for ontology editing [Funk et al., 2007], and Natural Language Generation from existing ontologies, to provide a simplified process: starting with an existing ontology, either imported or produced using CLOnE, generate the Controlled Language, edit the text, then parse it back into the modified ontology. Since the user only has to work with the textual representation of the ontology, we created a prototype extension for SemNotes to test the process.

Our extension allowed all the steps of the process — import of existing ontologies, editing and exporting back into an ontology. It also had the added benefit of having access to all the ontologies loaded on the Semantic Desktop, and all the instances extracted from the personal data available on the desktop. Thus SemNotes allowed reuse of existing instances from the desktop in the ontology editing, and more importantly to create instances directly into the desktop repository, in a simple and user friendly way. It also meant that through this extension we could provide a way of easily creating or removing a relation between existing instances just by typing a sentence about it.

Controlled Language Annotation

As we described before, SemNotes provides semi-automatic annotation of notes, by checking the available desktop resources and suggesting instances which are relevant to the note. Semi-automatic annotation is a mechanism often used to simplify the process of semantic annotation, and support the user. By employing Controlled Natural Language techniques, we can improve further the annotation process, in a user-friendly way. [Davis et al., 2009] presents two approaches to applying Controlled Language to Semantic Annotation, called CLANN — Controlled Language ANNotation. The two approaches were used to measure and evaluate the balance needed between expressiveness of the language and ease of use. CLANN is based on CLIE (Controlled Language for Information Extraction) and CLOnE.

Semantic annotation with Controlled Language is better suited for some annotation scenarios than for others. CLANN is applied to two use cases, where the use of a controlled vocabulary is implicit — meeting minutes and status reports. However, other possible use cases include the eHealth and business domains. The two use cases benefit from a semantic note-taking tool like SemNotes, thus creating an extension that uses CLANN was the next logical step.

The CLANN extension of SemNotes allows users to simultaneously create and annotate meeting minutes or status reports in Controlled Natural Language. It goes beyond the initial interlinking capabilities of SemNotes, by allowing the creation of different relations between the notes and desktop resources, as well as the creation of new resources and relations. CLANN uses a specially designed ontology for the meeting minutes and status reports, called MEMO, which is based on the desktop ontologies described in Section 3.2.3. The notes must follow a specific template, which is then parsed by the service. In the parsing, the extension reuses SemNotes' mechanism for detecting related desktop resources. The information resulted from the parsed note is stored in the desktop repository.

7.2 Linking Publication Data with Sclippy

The work presented in Chapter 5 is based on previous work [Groza et al., 2009a] on linking publication data from the Web of Data and from the Semantic Desktop. It consists of a three step process (extraction, expansion, integration) that starts from a

file with no metadata and incrementally enrich it to a comprehensive semantic model around the given publication, linked and embedded within the personal information space. The process is implemented by Sclippy.

The three steps are:

- extraction — metadata is automatically extracted from the publication;
- expansion — the extracted metadata is used to search the Web of Data and find relevant information which is then connected to the original metadata; and
- integration — the metadata is further enriched by embedding it within the Semantic Desktop, where it is automatically linked with the existing personal metadata.

The work done by [Groza et al., 2009b] is broader in a sense than our work, as it also includes in the first step of the process — extraction — containing complex algorithms for shallow (i.e. title, authors, abstract) and deep (i.e. discourse knowledge items like claims, positions or arguments) extraction of metadata from documents. Our work relies on the metadata already extracted by external applications or by the underlying Semantic Desktop, and instead focuses on the expansion and integration steps. We also aimed to provide a generic model for the integration of Web of Data sources and the Semantic Desktop, broadening the scope from the world of publications and authors.

The linking of publications done by Sclippy was motivated by the growing difficulty for early stage researchers to determine relevant work in a domain. The existing efforts are limited in the online world, where there are many publishers, each providing access to disjunct corpora of publications, and regardless of how well interconnected and easy to search they are, they do not cover all possible sources. The Semantic Desktop, being the implicit place where researchers would store documents, and already providing means of interlinking and better management of information, is the obvious choice for connecting to the similar but richer information available on the Web.

While Sclippy served as basis for our work, it also provided our first real-world use case where the interlinking of the Semantic Desktop data with the Web of Data would be useful, as well as the first example of a customised desktop service benefiting from it.

on the desktop, by means of usual desktop applications (e.g. file andWeb browser). while the expansion is achieved via the Semantic Web Dog Food Server and the Faceted DBLP 7 linked data repositories.

7.3 Building User-Applications with Konduit

An added benefit of structured information is its potential for reuse: being able to integrate existing data, relieves users from creating it again. While a large part of the data is found online, when it comes to working with information, users still rely on the familiar environment of desktop-based applications. On the other hand, Web-based applications can only access Web data, and do not integrate well with desktop information.

Konduit [Dragan et al., 2009] offers a way of accessing structured Web data from the desktop, integrating it with existing desktop data and applications, and working with both in a unified way. It precedes the system for finding Web aliases for semantic resources from the desktop, presented in Chapter 5.

The goal of Konduit was to allow the users to easily design their own personalised applications, customised to their needs and based on their workflows, without requiring prior knowledge of programming languages, or even semantics. Our approach with Konduit is based on a combination of the visual programming paradigm and the idea of UNIX pipes, and allows the casual users to build simple programs in order to perform and automate everyday tasks on RDF data. In other words, it is visual programming for RDF.

7.3.1 Components and Workflows

Konduit provides a collection of useful components ready for immediate use. The components offer individual units of functionality and are represented visually as blocks. They are connected through input and output slots, and in this way the flow of the program is defined. In order to keep simple the task of connecting components, the only data that flows through the workflow is RDF. This condition ensures that each component always fulfils the minimal requirement for dealing with its input. Obviously, components may be specialised with respect to the actual vocabulary on which they can operate and will decide at runtime if and how they deal with the incoming RDF. By neither allowing different kinds of data (e.g., text, numbers, lists, images, etc.), nor typing the RDF data with respect to the vocabularies they use, we stay very close to the original UNIX pipes concept, where data is always an untyped stream of bytes, and where it is up to each process or program how to handle it.

The architecture of Konduit is modular, each component is realised as a plugin which can be independently installed or removed. We expect that new plugins will be developed and shared by external power users, as the need for them arises. Formally, a component is defined by the following parameters:

$$Component = (I, O, P, F)$$

- a set of RDF input slots I,
- a set of RDF output slots O,
- a set of parameters P which allow for user input in the workflow,
- a function F, which works on the input I and generates the output O.

The parameters P influence the behaviour of F.

The number of input and output slots is not fixed and can be 0 or more. Depending on the number of slots, components can be grouped in three categories: sources, sinks, and ordinary components. Sources are components that do not have any inputs. They supply the workflow with data. There is always at least one source at the start of any workflow. Because data graphs can be merged, there can be more than one source for any workflow. Typical examples of sources are connectors to RDF stores, file (URL) input components, or converters from other, non-RDF formats. Sinks are components that do not have any outputs. They represent the final point(s) of any workflow. Examples of sink components are application adaptors, serialisers (file output components) and visualisers. In Konduit, workflows are activated from a sink component, usually by clicking on an activation button.

Several connected components make up a workflow, which is defined by specifying

$$Workflow = (C, f), where f : inputs(C) \rightarrow outputs(C) \cup \{\ nil\ \}$$

- a set of components C,
- a function f defined from the set of all the inputs of the components of C to the set of all the outputs of the components of C and the *nil* output.

The function f shows how the components of C are connected. The inputs that are not connected have a *nil* value of f; the outputs that do not represent a value of f are not connected.

Workflows can be saved and reused. Saving a workflow implies saving all the components that have at least one connection to it, as well as their existing connections, parameters and layout. There is no restriction that the components should be completely connected, so there can be input or output slots that remain open. A saved workflow can be reopened and modified by adding to it or removing components, or by changing connection or parameters and thus obtaining different workflows with minimum effort.

An important aspect of Konduit is directly tied to the fact that all inputs and outputs are RDF graphs. As a result, any workflow can itself become a component, meaning that workflows can be built recursively. In this way, it is possible to create specialised components, which we call black boxes, based on the combination of other components. The blackboxes are added to the existing library for reuse. An example of saved workflows and blackboxes is shown in Figure 7.2. The workflow is taken from `http://smile.deri.ie/konduit/discography`, where an entire example use case is presented.

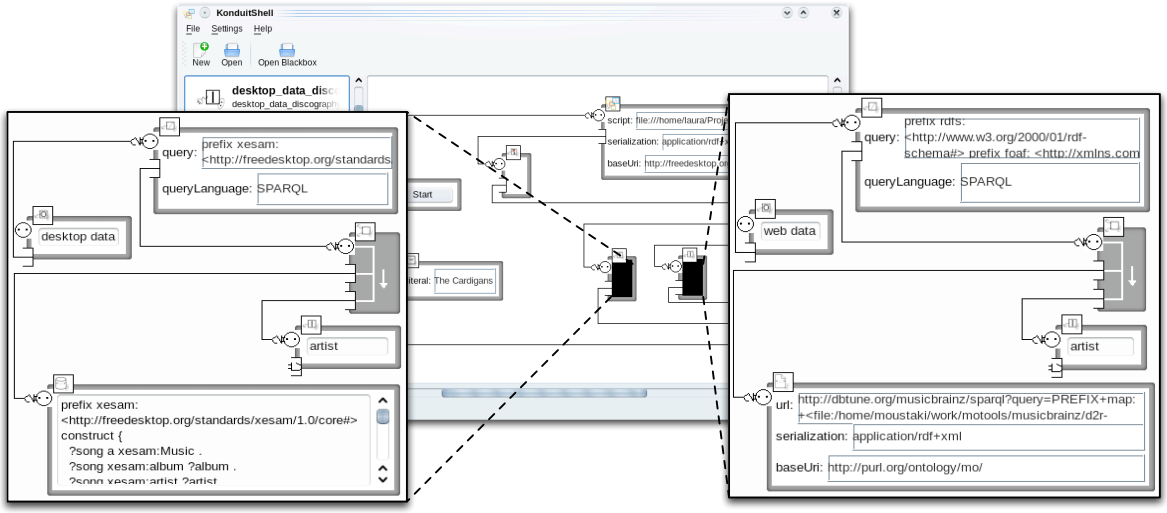

Figure 7.2: The entire discography generator workflow.

[Dragan et al., 2009] describes in more detail the different types of sources, sinks, and ordinary component types, as well as the blackbox mechanism. Some of the basic components available for Konduit require previous knowledge of writing SPARQL queries. Since the queries can influence the performance of the entire workflow, we recognise the need for a smart query editor that is suitable for naive users. [Ambrus et al., 2010] describes the means for making Konduit more user friendly, by adding a smart wizard with suggestions and SPARQL autocompletion.

Part IV

Conclusion

Chapter 8

Conclusion and Outlook

The Semantic Desktop is a framework for semantic interlinking of desktop data. We ground our work in it, using the building blocks it provides — the foundations of data representation and the layered service oriented architecture —, to enhance it further, through good user-facing semantic applications, and to connect it to the large source of Linked Data that is the Web.

The approach we took in this thesis is data-centric. We focused on maintaining and enriching the network of linked personal data that the Semantic Desktop enables. The path we took is two-fold, with both directions working towards interlinking semantic personal data. The first direction is internal, on the Semantic Desktop, through the means of new semantic applications, designed to support, and even more, encourage interlinking. Then second direction is external, connecting the desktop to the Web of Data. Both directions lead to a better interconnected environment, regardless of whether the data resides on the desktop or online.

This chapter summarises the work presented in this thesis, reiterating the contributions and presenting a general discussion and insights gathered. We conclude with a list of open questions and directions for future research and a final summary of the work.

8.1 Contributions

This thesis presents three main contributions, as mentioned in Section 1.3. The first contribution sets the scene for the rest of the work, by surveying existing Semantic

Desktop systems and applications, in Chapter 3. The survey presents an extensive list of systems, and compares and contrasts their features, architectures, data representation and handling. The rest of the work focuses on interlinking personal semantic data, following the two complementary directions.

The first direction tackles interlinking of personal data within the Semantic Desktop. Because the framework that the Semantic Desktop provides is mostly invisible to the end users, the benefits it brings must be reflected through the applications that use the framework. Thus, creating new semantic data inside the Semantic Desktop, including making new connections between resources, focuses on enabling the users to do so through semantic applications. We describe the challenges of developing good semantic applications in Chapter 4, and we present our solutions through an example. We use SemNotes, our semantic note-taking tool for the Nepomuk-KDE Semantic Desktop, to describe the design process and implementation. Although it is a relatively small application, it covers all the life-cycle phases of semantic data, and does so in a domain which is not specific or restricted — note-taking.

SemNotes supports the integration of new semantic information, the notes, with the network of existing information available on the desktop. It encourages interlinking, by making it very easy for users to connect the notes to the relevant resources mentioned in the text. Through SemNotes we describe the importance, and difficulty, of information visualisation when working with semantic data. We also present a user study conducted to compare SemNotes with Evernote in terms of the effort spent on annotation versus effort spent searching for information.

The second direction looks outside the Semantic Desktop, to the Web of Data. It capitalises on the common representation and structure of the data in the two spaces. Because personal information from the desktop is rarely disconnected from the rest of the information available on the Web, and most entities from the desktop appear online as well, we defined and implemented an algorithm which finds and connects matching entities from the desktop and the Web. We describe the bridging of the two spaces in Chapter 5. We evaluated the algorithm against a gold standard of relevance judgements by experts, and proved that it produces good results according to our requirement for high precision.

We weave the two threads of interlinking within and outside the desktop into a use case presenting semantic blogging. The use case, described in Chapter 6 presents a system where notes taken with SemNotes on the Semantic Desktop, and connected with

the relevant local entities, are published safely on the Web of Data as blog posts, while preserving the context created around them on the desktop, and following the principles of publishing Linked Data.

8.2 Directions for Future Research

We have presented in this thesis, two directions for interlinking personal semantic data. The purpose of enabling and encouraging interlinking of personal data is to create and maintain an explicit network of connected information that reflects the way we think about that information, and to use this network to improve the way we work with the information.

We described the challenges faced when developing an application for the Semantic Desktop, and exemplified potential solutions to them through SemNotes design and development. However, we believe there is still much improvement to be made in the repertoire of applications for the Semantic Desktop.

We plan to further investigate Information Extraction algorithms and methods, to support:

- the creation of multiple types of relations based on the text of the notes taken with SemNotes,
- the extraction of new entities from text and connecting them to the notes,
- the extraction of links between entities mentioned in the notes — instead of creating the links between the note and the mentioned resource.

Some of this functionality is already supported through the use of controlled language, but we would like to experiment further with extracting information from free text, possibly using something like the pidgin language processor [Kleek et al., 2007].

In Chapter 3 we described many Semantic Desktop systems, and one of the recurring applications that they provide is a browser for the semantic data, resources and connections. Such a browser is an essential tool to allow users to peek at their data without it being filtered by any particular application, but so far, exploring semantic information has not been an easy task, as most visualisations for generic data are either graph based or tabular, and generally not pretty. One of the new and interesting visualisations that we are planning to test on personal information is the Atom interface [Samp et al., 2008].

Finally, a third direction for the future is devising and running a long term, large scale user study, to gather insights into how users really use the current functionalities offered by their Semantic Desktop. We have started work in this direction, targeting users of the Nepomuk-KDE Semantic Desktop, because of the large user base that KDE has. We plan to look into what kind of semantic information is used, and what are its dynamics. We hope that such a study would help us focus our research on things which have the most impact on the way the Semantic Desktop is used.

8.3 Summary

The main contributions of this thesis focus on supporting interlinking of personal semantic data on a Semantic Desktop and beyond.

Conceptually, we present the challenges of designing semantic applications on top of the framework provided by the Semantic Desktop, and we discuss options and possible solutions. We also detail an algorithm for bridging the gap between the connected network of information formed on the desktop with the much larger network of Linked Data on the Web, through resource matching and finding *Web aliases* for desktop entities.

From the implementation point of view, we support the conceptual contributions with corresponding software.

SemNotes is a note-taking semantic application for the Nepomuk-KDE Semantic Desktop. In Chapter 4 we describe the design and implementation of SemNotes, as an illustration of possible solution to the challenges found.

Desktop service for Web aliases is a service for Nepomuk-KDE Semantic Desktop which automatically finds Web aliases for desktop entities and saves them locally on the desktop. In Chapter 5 we describe the algorithm as well as the implementation, which allows for various modes of utilisation, depending on the use case.

We evaluated both implementations and the results are positive:

- A comparative task-based user evaluation of SemNotes against Evernote showed that although using SemNotes for complex annotations requires more mouse clicks, there is no significant difference in time spent, while SemNotes requires significantly less effort (in time spent) for some complex searches.

- The matching algorithm of the desktop service for Web aliases was evaluated against a gold standard of relevance judgements from experts, which we also constructed. The results were positive, our algorithm proving to have high precision, which was our initial goal and requirement, due to the automatic function of the service.

Finally, we combined the two directions into a coherent use case for semantic blogging, and described a system which makes use of both SemNotes and the framework for finding Web aliases for desktop resources. It can be generalised as a publishing platform for personal semantic data from the desktop to the Web, following the Linked Data principles, maintaining the context, while at the same time not exposing sensitive information.

Appendix A

SemNotes evaluation – Questionnaire

Page 1

Thank you for doing this evaluation. This is the last part, where we would like to find out your impressions about SemNotes and the tasks of the evaluation.

Which environment did you test first? *

- ○ EverNote on Windows
- ○ SemNotes on Linux

Page 2 — Personal information

You can choose not to answer any of the questions in this section if you feel they are too personal. We tried to keep them to a minimum, and every answer will help.

You are ...

- ○ a Msc. student
- ○ a Ph.D. student
- ○ a post-doctoral researcher
- ○ a researcher
- ○ other

You are ...

- ○ under 20 years old
- ○ 20 to 24 years old
- ○ 25 to 29 years old
- ○ 30 to 34 years old
- ○ over 35 years old

You are ...

- ○ female
- ○ male

Page 3 — Information about your work environment

Your digital work environment that is!

What operating system do you use in your everyday work? *

- ○ Windows
- ○ OSX
- ○ Linux
- ○ other

Do you use any note-taking tools on your computer? *

- ○ Yes
- ○ No

Page 4 — Note-taking

What note-taking tool(s) do you use? *

- ☐ EverNote
- ☐ Microsoft OneNote
- ☐ Basket
- ☐ Tomboy
- ☐ Notepad
- ☐ Notepad++
- ☐ other

Why did you choose that application(s)? (Select all that apply) *

- ☐ It was the first application that I found.
- ☐ I had used it before, and I knew it.
- ☐ It was recommended to me.
- ☐ It was the only one for my setup.
- ☐ It is the only one that had the features I was looking for.
- ☐ other

What is the feature you like the most when note-taking? *

What is the feature you use the most when note-taking? *

Do you use text formatting when note-taking? *

(Never)	**1**	**2**	**3**	**4**	**5**	(Always)
	○	○	○	○	○	

Do you tag your notes? (If this feature is not available in the application you use, please skip this question.)

(Never)	**1**	**2**	**3**	**4**	**5**	(Always)
	○	○	○	○	○	

How many of the notes you took did you revisit at least once? *

(None)	**1**	**2**	**3**	**4**	**5**	(Most)
	○	○	○	○	○	

What do you use note-taking for? (Select all that apply.) *

- ☐ Shopping list
- ☐ Todos
- ☐ Meeting minutes
- ☐ Brainstorming
- ☐ Idea vault
- ☐ other

Page 5 — What did you think about the tasks?

How similar are the tasks to your daily activity? *

(Not at all similar)	**1**	**2**	**3**	**4**	**5**	(Very similar)
	○	○	○	○	○	

How familiar was the data provided? *

(Not at all familiar)	**1**	**2**	**3**	**4**	**5**	(Familiar)
	○	○	○	○	○	

How easy / difficult did you find the tasks? *

(Very easy)	**1**	**2**	**3**	**4**	**5**	(Very difficult)
	○	○	○	○	○	

How fast did you feel you could finish the tasks? (Compared to your speed in your own environment) *

(Much slower)	**1**	**2**	**3**	**4**	**5**	(Much faster)
	○	○	○	○	○	

Did you encounter any problems while working on the tasks? (If No, leave blank.)

Page 6 — How did you like SemNotes?

What is your overall impression of SemNotes? *

(Very bad)	**1**	**2**	**3**	**4**	**5**	(Very good)
	○	○	○	○	○	

How useful do you consider the following features of SemNotes? *

	(Not useful)	**1**	**2**	**3**	**4**	**5**	(Very useful)
Tagging		○	○	○	○	○	
Rating		○	○	○	○	○	
Formatting		○	○	○	○	○	
Full text search		○	○	○	○	○	
Tag cloud		○	○	○	○	○	
Resource panel		○	○	○	○	○	
Timeline		○	○	○	○	○	
Linking of notes		○	○	○	○	○	
Filtering		○	○	○	○	○	
Sorting		○	○	○	○	○	

Which feature of SemNotes did you like the most? (if any)

Which feature of SemNotes did you like the least? (if any)

What other features would you like SemNotes to have? (if any)

Which features from SemNotes did you miss in EverNote? (if any)

Which of the two applications do you feel that helped you finish the tasks faster? *

- ○ EverNote
- ○ SemNotes
- ○ I didn't feel any difference

Which of the two applications do you feel that helped you finish the tasks better? *

- ○ SemNotes
- ○ EverNote
- ○ I didn't feel any difference

Page 7 — That was all! THANK YOU!

Do you have anything else to tell us?

Bibliography

[Adar et al., 1999] Adar, E., Karger, D. R., and Stein, L. A. (1999). Haystack: Per-User Information Environments. In *Proceedings of the 8th International Conference on Information and Knowledge Management*.

[Adida et al., 2008] Adida, B., Birbeck, M., McCarron, S., and Pemberton, S. (2008). RDFa in XHTML: Syntax and Processing — A collection of attributes and processing rules for extending XHTML to support RDF. W3C Recommendation, W3C. `http://www.w3.org/TR/rdfa-syntax/`.

[Ambite et al., 2005] Ambite, J. L., Chaudhri, V. K., Fikes, R., Jenkins, J., Mishra, S., Muslea, M., Uribe, T., and Yang, G. (2005). Integration of Heterogeneous Knowledge Sources in the CALO Query Manager. Technical report.

[Ambite et al., 2006] Ambite, J.-l., Chaudhri, V. K., Fikes, R., Jenkins, J., Mishra, S., Muslea, M., Uribe, T., and Yang, G. (2006). Design and Implementation of the CALO Query Manager. In *Proceedings of the Innovative Applications of Artificial Intelligence (AAAI2006)*.

[Ambrus et al., 2010] Ambrus, O., Möller, K., and Handschuh, S. (2010). Konduit VQB: a Visual Query Builder for SPARQL on the Social Semantic Desktop. In *Proceedings of the Workshop on Visual Interfaces to the Social and Semantic Web (VISSW 2010)*.

[Aumueller and Auer, 2005] Aumueller, D. and Auer, S. (2005). Towards a Semantic Wiki Experience — Desktop Integration and Interactivity in WikSAR. In *Proceedings of the 1st Workshop on The Semantic Desktop*.

[Barreau, 1995] Barreau, D. K. (1995). Context as a Factor in Personal Information Management Systems. *Journal of the American Society for Information Science*, 46(5).

[Barreau and Nardi, 1995] Barreau, D. K. and Nardi, B. A. (1995). Finding and Reminding: File Organization From the Desktop. *SIGCHI Bulletin*, 27(3).

[Beckett, 2004] Beckett, D. (2004). RDF/XML Syntax Specification (Revised). W3C Recommendation, W3C. http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/.

[Beckett, 2007] Beckett, D. (2007). Turtle — Terse RDF Triple Language. http://www.dajobe.org/2004/01/turtle/.

[Bell, 2001] Bell, G. (2001). A Personal Digital Store. *Communications of the ACM*, 44(1).

[Bell, 2007] Bell, G. (2007). A Digital Life. *Scientific American Magazine*.

[Bell et al., 2004] Bell, G., Gemmell, J., and Lueder, R. (2004). Challenges in Using Lifetime Personal Information Stores. In *Proceedings of the 27th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR2004)*.

[Benjelloun et al., 2006] Benjelloun, O., Garcia-Molina, H., Jonas, J., Su, Q., and Widom, J. (2006). Swoosh: A Generic Approach to Entity Resolution. Technical report, Stanford University.

[Bernardi et al., 2008] Bernardi, A., Decker, S., Van Elst, L., Grimnes, G. A., Groza, T., Handschuh, S., Jazayeri, M., Mesnage, C., Moeller, K., Reif, G., and Sintek, M. (2008). The Social Semantic Desktop A New Paradigm Towards Deploying the Semantic Web on the Desktop. In *Semantic Web Engineering in the Knowledge Society*.

[Berners-Lee, 1989] Berners-Lee, T. (1989). Information management: A proposal. http://www.w3.org/History/1989/proposal.html.

[Berners-Lee, 2000] Berners-Lee, T. (2000). *Weaving the Web—The Past, Present and Future of the World Wide Web by its Inventor*. Harper.

[Berners-Lee, 2006a] Berners-Lee, T. (2006a). Linked data. http://www.w3.org/DesignIssues/LinkedData.html.

[Berners-Lee, 2006b] Berners-Lee, T. (2006b). Notation 3. http://www.w3.org/DesignIssues/Notation3.

[Berners-Lee et al., 1994] Berners-Lee, T., Cailliau, R., Luotonen, A., Nielsen, H. F., and Secret, A. (1994). The World-Wide Web. *Communications of the ACM*, 37(8).

[Berners-Lee et al., 2001] Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The Semantic Web. *Scientific American*, 284(5).

[Bernstein et al., 2008] Bernstein, M., Van Kleek, M., Karger, D., and schraefel, m. c. (2008). Information Scraps: How and Why Information Eludes Our Personal Information Management Tools. *ACM Transactions on Information Systems*, 26(4).

[Bernstein et al., 2007] Bernstein, M., Van Kleek, M., Karger, D. R., and schraefel, m. (2007). Wicked Problems and Gnarly Results: Reflecting on Design and Evaluation Methods for Idiosyncratic Personal Information Management Tasks. Technical report, Electronics and Computer Science, University of Southampton, UK.

[Berry et al., 2011] Berry, P., Gervasio, M., Peintner, B., and Yorke-Smith, N. (2011). PTIME: Personalized Assistance for Calendaring. *ACM Transactions on Intelligent Systems and Technology*, 2(4).

[Berry et al., 2004] Berry, P., Gervasio, M., Uribe, T., Myers, K., and Nitz, K. (2004). A Personalized Calendar Assistant. In *Proceedings of the AAAI Spring Symposium*.

[Berry et al., 2005] Berry, P. M., Gervasio, M. T., Uribe, T. E., and Yorke-Smith, N. (2005). Mixed-Initiative Issues for a Personalized Time Management Assistant. In *Proceedings of Workshop on Mixed-Initiative Planning and Scheduling (ICAPS2005)*.

[Bizer et al., 2007] Bizer, C., Cyganiak, R., and Heath, T. (2007). How to Publish Linked Data on the Web. `http://sites.wiwiss.fu-berlin.de/suhl/bizer/pub/LinkedDataTutorial/`.

[Bizer et al., 2009] Bizer, C., Volz, J., Kobilarov, G., and Gaedke, M. (2009). Silk - A Link Discovery Framework for the Web of Data. In *Proceedings of the 18th International World Wide Web Conference (WWW2009)*.

[Blanc-Brude and Scapin, 2007] Blanc-Brude, T. and Scapin, D. L. (2007). What Do People Recall About Their Documents?: Implications for Desktop Search Tools. In *Proceedings of the 12th International Conference on Intelligent User Interfaces (IUI2007)*.

[Boardman, 2004] Boardman, R. (2004). *Improving Tool Support for Personal Information Management*. Phd thesis, Imperial College, University of London.

[Boardman and Sasse, 2004] Boardman, R. and Sasse, A. M. (2004). "Stuff Goes into the Computer and Doesn't Come Out": A Cross-tool Study of Personal Information Management. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

[Bojars et al., 2008] Bojars, U., Passant, A., Cyganiak, R., and Breslin, J. (2008). Weaving SIOC into the Web of Linked Data. In *Proceedings of the Linked Data on the Web Workshop (LDOW2008)*.

[Bouquet et al., 2007] Bouquet, P., Stoermer, H., and Giacomuzzi, D. (2007). OKKAM: Enabling a Web of Entities. In *Proceedings of the Workshop I3: Identity, Identifiers, Identification, Entity-Centric Approaches to Information and Knowledge Management on the Web.*

[Breitman and Truszkowski, 2005] Breitman, K. and Truszkowski, W. (2005). Agent-mediated Pro-active Web-sites. In *Proceedings of the Second International Conference on Radical Agent Concepts: Innovative Concepts for Autonomic and Agent-Based Systems.*

[Breitman and Truszkowski, 2006] Breitman, K. and Truszkowski, W. (2006). The Autonomic Semantic Desktop: Helping Users Cope with Information System Complexity. In *Proceedings of the Third IEEE International Workshop on Engineering of Autonomic & Autonomous Systems (EASE2006).*

[Breslin et al., 2005] Breslin, J. G., Harth, A., Bojārs, U., and Decker, S. (2005). Towards Semantically-Interlinked Online Communities. In *Proceedings of the 2nd European Semantic Web Conference (ESWC2005).*

[Brickley and Guha, 2004] Brickley, D. and Guha, R. (2004). RDF vocabulary description language 1.0: RDF schema. W3C Recommendation, W3C. `http://www.w3.org/TR/rdf-schema/`.

[Brickley and Miller, 2005] Brickley, D. and Miller, L. (2005). FOAF vocabulary specification. `http://xmlns.com/foaf/0.1/`.

[Broekstra and Kampman, 2003] Broekstra, J. and Kampman, A. (2003). The SeRQL Query Language. Technical report, Aduna.

[Brunkhorst et al., 2006] Brunkhorst, I., Chirita, P. A., Costache, S., Gaugaz, J., Ioannou, E., Iofciu, T., Minack, E., Nejdl, W., and Paiu, R. (2006). The Beagle++ Toolbox: Towards an Extendable Desktop Search Architecture. In *Proceedings of the Semantic Desktop and Social Semantic Collaboration Workshop (SemDesk 2006).*

[Buckland, 1992] Buckland, M. K. (1992). Emmanuel Goldberg, Electronic Document Retrieval, and Vannevar Bush's Memex. *Journal of the American Society for Information Science*, 43.

[Bush, 1945] Bush, V. (1945). As we may think. *The Atlantic Monthly*, 3(176):101–108.

[Carroll et al., 2005] Carroll, J. J., Bizer, C., Hayes, P., and Stickler, P. (2005). Named graphs, Provenance and Trust. In *Proceedings of the 14th International World Wide Web Conference (WWW2005).*

[Cayzer, 2006] Cayzer, S. (2006). What Next for Semantic Blogging? In *Proceedings of the SEMANTICS Conference*.

[Cayzer and Shabajee, 2003] Cayzer, S. and Shabajee, P. (2003). Semantic Blogging and Bibliography Management. In *Proceedings of BlogTalk Conference*.

[Cheyer et al., 2005] Cheyer, A., Park, J., and Giuli, R. (2005). IRIS: Integrate. Relate. Infer. Share. In *Proceedings of the 1st Workshop on The Semantic Desktop*.

[Chirita et al., 2005] Chirita, P.-A., Ghita, S., Nejdl, W., , and Paiu, R. (2005). Semantically Enhanced Searching and Ranking on the Desktop . In *Proceedings of the 1st Workshop on The Semantic Desktop*.

[Civan et al., 2008] Civan, A., Jones, W., Klasnja, P., and Bruce, H. (2008). Better to Organize Personal Information by Folders or by Tags?: The Devil is in the Details. In *Proceedings of the American Society for Information Science and Technology*.

[Conley and Carpenter, 2007] Conley, K. and Carpenter, J. (2007). Towel: Towards an Intelligent To-do List. In *Proceedings of the AAAI Spring Symposium on Interaction Challenges for Artificial Assistants*.

[Craik, 1943] Craik, K. (1943). *The Nature of Exploration*. Cambridge University Press.

[Cutrell et al., 2006] Cutrell, E., Robbins, D. C., Dumais, S. T., and Sarin, R. (2006). Fast, Flexible Filtering with Phlat — Personal Search and Organization Made Easy. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

[d'Aquin et al., 2010] d'Aquin, M., Elahi, S., and Motta, E. (2010). Personal Monitoring of Web Information Exchange: Towards Web Lifelogging. In *Proceedings of the WebSci10: Extending the Frontiers of Society On-Line*.

[d'Aquin et al., 2011a] d'Aquin, M., Elahi, S., and Motta, E. (2011a). Semantic Technologies to Support the User-Centric Analysis of Activity Data. In *Proceedings of the 4th Workshop on Social Data on the Web (SDoW2011), co-located with the 10th International Semantic Web Conference (ISWC2011)*.

[d'Aquin et al., 2011b] d'Aquin, M., Rowe, M., and Motta, E. (2011b). Self-Tracking on the Web: Why and How. In *Proceedings of the W3C Workshop on Web Tracking and User Privacy*.

[Davis et al., 2010] Davis, B., Dantuluri, P., Handschuh, S., and Cunningham, H. (2010). Towards Controlled Natural Language for Semantic Annotation. *International Journal on Semantic Web and Information Systems*, 6(4).

[Davis et al., 2008] Davis, B., Iqbal, A., Funk, A., Tablan, V., Bontcheva, K., Cunningham, H., and Handschuh, S. (2008). Roundtrip Ontology Authoring. In *Proceedings of the 7th International Semantic Web Conference (ISWC2008)*.

[Davis et al., 2009] Davis, B., Varma, P., Handschuh, S., and Dragan, L. (2009). On designing Controlled Natural Languages for Semantic Annotation. In *Proceedings of the Workshop on Controlled Natural Language (CNL2009)*.

[Davis, 2012] Davis, B. P. (2012). *On Applying Controlled Natural Languages for Ontology Authoring and Semantic Annotation*. PhD thesis, National University of Ireland, Galway (NUIG).

[Dawson and Howes, 1998] Dawson, F. and Howes, T. (1998). vCard MIME Directory Profile. IETF RFC 2426 (Proposed Standard).

[Decker and Frank, 2004] Decker, S. and Frank, M. (2004). The Social Semantic Desktop. Technical report.

[Delbru et al., 2010] Delbru, R., Toupikov, N., Catasta, M., Tummarello, G., and Decker, S. (2010). Hierarchical Link Analysis for Ranking Web Data. In *Proceedings of the 7th Extended Semantic Web Conference (ESWC 2010)*.

[Dodds and Davis, 2010] Dodds, L. and Davis, I. (2010). *Linked Data Patterns*. `http://patterns.dataincubator.org`.

[Dong and Halevy, 2005a] Dong, X. and Halevy, A. (2005a). A Platform for Personal Information Management and Integration. In *Proceedings of the Second Biennial Conference on Innovative Data Systems Research (CIDR2005)*.

[Dong and Halevy, 2005b] Dong, X. L. and Halevy, A. (2005b). Malleable Schemas: A Preliminary Report. In *Proceedings of the Eight International Workshop on the Web & Databases (WebDB2005)*.

[Dong et al., 2004] Dong, X. L., Halevy, A., Nemes, E., Sigurdsson, S. B., and Domingos, P. (2004). SEMEX: Toward On-the-fly Personal Information Integration. In *Proceedings of the 4th International Workshop on Information Integration on the Web (IIWeb2004)*.

[Dong et al., 2005] Dong, X. L., Halevy, A. Y., and Madhavan, J. (2005). Reference Reconciliation in Complex Information Spaces. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

[Dourish et al., 2000] Dourish, P., Edwards, W. K., LaMarca, A., Lamping, J., Petersen, K., Salisbury, M., Terry, D. B., and Thornton, J. (2000). Extending Document Man-

agement Systems with User-specific Active Properties. *ACM Transactions on Information Systems*, 18(2).

[Dourish et al., 1999] Dourish, P., Edwards, W. K., Lamarca, A., and Salisbury, M. (1999). Presto: An Experimental Architecture for Fluid Interactive Document Spaces. *ACM Transactions on Computer-Human Interaction*, 6.

[Dragan and Decker, 2012] Dragan, L. and Decker, S. (2012). Knowledge Management on the Desktop. In *Proceedings of the 18th International Conference on Knowledge Engineering and Knowledge Management (EKAW2012)*.

[Dragan et al., 2011a] Dragan, L., Delbru, R., Groza, T., Handschuh, S., and Decker, S. (2011a). Linking Semantic Desktop Data to the Web of Data. In *Proceedings of the 10th International Semantic Web Conference (ISWC2011)*.

[Dragan and Handschuh, 2009] Dragan, L. and Handschuh, S. (2009). SemNotes — Note-taking on the Semantic Desktop. In *Proceedings of the 6th European Semantic Web Conference (ESWC 2009) Poster Session*.

[Dragan et al., 2011b] Dragan, L., Handschuh, S., and Decker, S. (2011b). The Semantic Desktop at Work: Interlinking Notes. In *Proceedings of the 7th International Conference on Semantic Systems (I-SEMANTICS2011)*.

[Dragan et al., 2009] Dragan, L., Moeller, K., Handschuh, S., Ambrus, O., and Trueg, S. (2009). Converging Web and Desktop Data with Konduit. In *Proceedings of the 5th Workshop on Scripting for the Semantic Web (SFSW2009)*.

[Dragan et al., 2010] Dragan, L., Passant, A., Groza, T., and Handschuh, S. (2010). Linking Semantic Personal Notes. In *Proceedings of the 1st Workshop on Knowledge Injection into and Extraction from Linked Data (KIELD2010)*.

[Dumais et al., 2003] Dumais, S., Cutrell, E., Cadiz, J., Jancke, G., Sarin, R., and Robbins, D. C. (2003). Stuff I've Seen: A System for Personal Information Retrieval and Re-use. In *Proceedings of the 26th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR2003)*.

[Einsfeld et al., 2005] Einsfeld, K., Ebert, A., Agne, S., and Klein, B. (2005). DocuWorld — A 3-D User Interface to the Semantic Desktop. In *Proceedings of the 1st Workshop on The Semantic Desktop*.

[Elmagarmid et al., 2007] Elmagarmid, A. K., Ipeirotis, P. G., and Verykios, V. S. (2007). Duplicate Record Detection: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 19(1).

[Elsweiler and Ruthven, 2007] Elsweiler, D. and Ruthven, I. (2007). Towards Task-based Personal Information Management Evaluations. In *Proceedings of the 30th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR2007)*.

[Engelbart, 1962] Engelbart, D. C. (1962). *Augmenting Human Intellect: A Conceptual Framework*.

[Engelbart and English, 1968] Engelbart, D. C. and English, W. K. (1968). A Research Center for Augmenting Human Intellect. In *Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, part I (AFIPS1968)*.

[Fellegi and Sunter, 1969] Fellegi, I. P. and Sunter, A. B. (1969). A Theory for Record Linkage. *Journal of the American Statistical Association*, 64(328).

[Fertig et al., 1996a] Fertig, S., Freeman, E., and Gelernter, D. (1996a). "finding and reminding" reconsidered. *SIGCHI Bull.*, 28(1):66–69.

[Fertig et al., 1996b] Fertig, S., Freeman, E., and Gelernter, D. (1996b). "Finding and reminding" reconsidered. *SIGCHI Bulletin*, 28(1).

[Fertig et al., 1996c] Fertig, S., Freeman, E., and Gelernter, D. (1996c). Lifestreams: An Alternative to the Desktop Metaphor. In *Conference Companion on Human Factors in Computing Systems: Common Ground*.

[Franz, 2008] Franz, T. (2008). On the Evaluation of Personal Knowledge Management Solutions: Evaluating Tools of the X-COSIM Semantic Desktop. In *Proceedings of Workshop-Woche: Lernen, Wissen & Adaptivitat (LWA2008)*.

[Franz et al., 2009] Franz, T., Scherp, A., and Staab, S. (2009). Are Semantic Desktops Better?: Summative Evaluation Comparing a Semantic Against a Conventional Desktop. In *Proceedings of the 5th International Conference on Knowledge Capture (K-CAP2009)*.

[Franz and Staab, 2005] Franz, T. and Staab, S. (2005). SAM: Semantics Aware Instant Messaging for the Networked Semantic Desktop. In *Proceedings of the 1st Workshop on The Semantic Desktop*.

[Franz et al., 2007] Franz, T., Staab, S., and Arndt, R. (2007). The X-COSIM Integration Framework for a Seamless Semantic Desktop. In *Proceedings of the 4th International Conference on Knowledge Capture (K-CAP2007)*.

[Freeman and Fertig, 1995] Freeman, E. and Fertig, S. (1995). Lifestreams: Organizing Your Electronic Life. In *Proceedings of the AAAI Fall Symposium: AI Applications in Knowledge Navigation and Retrieval*.

[Freeman and Gelernter, 2007] Freeman, E. and Gelernter, D. (2007). Beyond Lifestreams: The Inevitable Demise of the Desktop Metaphor. In *Beyond the Desktop Metaphor Designing Integrated Digital Work Environments*.

[Funk et al., 2007] Funk, A., Tablan, V., Bontcheva, K., Cunningham, H., Davis, B., and Handschuh, S. (2007). CLOnE: Controlled Language for Ontology Editing. In *The Semantic Web*, Lecture Notes in Computer Science.

[Gemmell et al., 2006] Gemmell, J., Bell, G., and Lueder, R. (2006). MyLifeBits: a Personal Database for Everything. *Commununications of the ACM*, 49(1).

[Gemmell et al., 2002] Gemmell, J., Bell, G., Lueder, R., Drucker, S., and Wong, C. (2002). MyLifeBits: Fulfilling the Memex Vision. In *Proceedings of the 10th ACM International Conference on Multimedia (MULTIMEDIA2002)*.

[Gemmell et al., 2003] Gemmell, J., Lueder, R., and Bell, G. (2003). Living with a Lifetime Store. In *Proceedings of the ATR Workshop on Ubiquitous Experience Media*.

[Gifford et al., 1991] Gifford, D. K., Jouvelot, P., Sheldon, M. A., and O'Toole Jr., J. W. (1991). Semantic File Systems. In *Proceedings of the 13th ACM Symposium on Operating Systems Principles*.

[Goldberg, 1988] Goldberg, A., editor (1988). *A History of Personal Workstations*.

[Goldschmidt and Otlet, 1906] Goldschmidt, R. and Otlet, P. (1906). On a New Form of the Book: the Microphotographic Book. In *International Organisation and Dissemination of Knowledge : Selected Essays of Paul Otlet*.

[Grant and Beckett, 2004] Grant, J. and Beckett, D. (2004). RDF test cases. W3C Recommendation, W3C. `http://www.w3.org/TR/2004/REC-rdf-testcases-20040210/`.

[Grimnes et al., 2009] Grimnes, G. A., Sauermann, L., and Bernardi, A. (2009). The Personal Knowledge Workbench of the NEPOMUK Semantic Desktop. In *Proceedings of the 6th European Semantic Web Conference on The Semantic Web (ESWC2009)*.

[Groza et al., 2009a] Groza, T., Dragan, L., Handschuh, S., and Decker, S. (2009a). Bridging the Gap between Linked Data and the Semantic Desktop. In *Proceedings of the 8th International Semantic Web Conference (ISWC2009)*.

[Groza et al., 2011] Groza, T., Handschuh, S., and Decker, S. (2011). Capturing Rhetoric and Argumentation Aspects within Scientific Publications. *Journal on Data Semantics XV*, 6720.

[Groza et al., 2009b] Groza, T., Handschuh, S., and Hulpus, I. (2009b). A Document Engineering Approach to Automatic Extraction of Shallow Metadata from Scientific Publications. Technical report, Digital Enterprise Research Institute, National University of Ireland, Galway.

[Gruber, 1993] Gruber, T. R. (1993). Towards Principles for the Design of Ontologies Used for Knowledge Sharing. In *Formal Ontology in Conceptual Analysis and Knowledge Representation*.

[Heitmann et al., 2012] Heitmann, B., Cyganiak, R., Hayes, C., and Decker, S. (2012). An Empirically Grounded Conceptual Architecture for Applications on the Web of Data. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 42(1).

[Hogan et al., 2007] Hogan, A., Harth, A., and Decker, S. (2007). Performing Object Consolidation on the Semantic Web Data Graph. In *Proceedings of the Workshop I3: Identity, Identifiers, Identification, Entity-Centric Approaches to Information and Knowledge Management on the Web*.

[Hull and Hart, 2001] Hull, J. J. and Hart, P. E. (2001). Toward Zero-Effort Personal Document Management. *Computer*, 34.

[Huynh et al., 2003] Huynh, D., Quan, D., and Karger, D. R. (2003). Haystack's User Experience for Interacting with Semistructured Information. In *Proceedings of the 12th International World Wide Web Conference (WWW2003)*.

[Huynh and Karger, 2002] Huynh, D. F. and Karger, D. R. (2002). Haystack: A Platform for Creating, Organizing and Visualizing Information Using RDF. In *Proceedings of the Workshop on the Semantic Web (SWW2002)*.

[Iga and Shinnishi, 2006] Iga, S. and Shinnishi, M. (2006). SnapShoot: Integrating Semantic Analysis and Visualization Techniques for Web-based Note Taking System. In *Proceedings of the Asia-Pacific Symposium on Information Visualisation (APVis'06)*.

[ISO, 2006] ISO, I. O. f. S. (2006). Iso 9241-110:2006 ergonomics of human-system interaction – part 110: Dialogue principles. http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?csnumber=38009.

[Jaffri et al., 2007] Jaffri, A., Glaser, H., and Millard, I. (2007). URI Identity Management for Semantic Web Data Integration and Linkage. In *Proceedings of the 3rd International Workshop On Scalable Semantic Web Knowledge Base Systems*.

[Järvelin and Wilson, 2003] Järvelin, K. and Wilson, T. D. (2003). On conceptual models for information seeking and retrieval research. *Information Research*, 9(1).

[Johnson-Laird, 1989] Johnson-Laird, P. N. (1989). *Mental models*. MIT Press.

[Jones et al., 2011] Jones, N. A., Ross, H., Lynam, T., Perez, P., and Leitch, A. (2011). Mental models: an interdisciplinary synthesis of theory and methods. *Ecology and Society*, 16(1).

[Jones, 2008] Jones, W. (2008). *Keeping Found Things Found: The Study and Practice of Personal Information Management: The Study and Practice of Personal Information Management*.

[Jones and Teevan, 2007] Jones, W. and Teevan, J. (2007). *Personal Information Management*. University of Washington Press.

[Kaptelinin and Boardman, 2007] Kaptelinin, V. and Boardman, R. (2007). Toward Integrated Work Environments: Application-centric Versus Workspace-level Design. In *Beyond the Desktop Metaphor: Designing Integrated Digital Work Environments*.

[Karger, 2007] Karger, D. R. (2007). It's All the Same to Me: Data Unification in Personal Information Management. In *Personal Information Management*.

[Karger et al., 2005] Karger, D. R., Bakshi, K., Huynh, D., Quan, D., and Sinha, V. (2005). Haystack: A Customizable General-Purpose Information Management Tool for End Users of Semistructured Data. In *Proceedings of the Second Biennial Conference on Innovative Data Systems Research (CIDR2005)*.

[Karger and Jones, 2006] Karger, D. R. and Jones, W. (2006). Data Unification in Personal Information Management. *Communications of the ACM*, 49(1).

[Karger and Quan, 2004] Karger, D. R. and Quan, D. (2004). What Would It Mean to Blog on the Semantic Web? In *Proceedings of the 3rd International Semantic Web Conference (ISWC2004)*.

[Karvounarakis et al., 2002] Karvounarakis, G., Alexaki, S., Christophides, V., Plexousakis, D., and Scholl, M. (2002). RQL: A Declarative Query Language for RDF. In *Proceedings of the International World-Wide Web Conference (WWW2002)*.

[Katifori et al., 2005] Katifori, V., Poggi, A., Scannapieco, M., and Catarci, T. (2005). OntoPIM: How to Rely on a Personal Ontology for Personal Information Management. In *Proceedings of the 1st Workshop on The Semantic Desktop*.

[Kelly, 2006] Kelly, D. (2006). Evaluating Personal Information Management Behaviors and Tools. *Communications of the ACM*, (1):84.

[Kleek et al., 2007] Kleek, M. V., Bernstein, M. S., Karger, D. R., and m. c. schraefel (2007). GUI — Phooey!: The Case for Text Input. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology (UIST 2007)*.

[Kleek et al., 2009] Kleek, M. V., Bernstein, M. S., Panovich, K., Vargas, G. G., Karger, D. R., and m. c. schraefel (2009). Note to Self: Examining Personal Information Keeping in a Lightweight Note-Taking Tool. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

[Klyne and Carroll, 2004] Klyne, G. and Carroll, J. J. (2004). Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation, W3C. `http://www.w3.org/TR/rdf-concepts/`.

[Knorr-Cetina, 1997] Knorr-Cetina, K. (1997). Sociality with Objects: Social Relations in Postsocial Knowledge Societies. *Theory, Culture & Society*, 14(4).

[Krishnan, 2008] Krishnan, A. (2008). Semantic LS - An Approach for Personal and Private Group Information Management. *Information Storage and Retrieval*.

[Lamming and Flynn, 1994] Lamming, M. and Flynn, M. (1994). "Forget-me-not" – Intimate Computing in Support of Human Memory. In *Proceedings of the Symposium on Next Generation Human Interface, invited keynote paper at FRIEND21 Symposium*.

[Lansdale, 1988] Lansdale, M. W. (1988). The Psychology of Personal Information Management. *Applied Ergonomics*, 19(1).

[Lassila and Swick, 1999] Lassila, O. and Swick, R. R. (1999). Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation, W3C. `http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/`.

[Lauriere et al., 2006] Lauriere, S., Solleiro, A., Trueg, S., Bogdan, C., Groth, K., and Lannero, P. (2006). Mandriva Community Scenario Report. Technical report.

[Martin Dvorak, 2012] Martin Dvorak (2012). MindRaider. `http://mindraider.sourceforge.net/`.

[Miles and Bechhofer, 2009] Miles, A. and Bechhofer, S. (2009). SKOS Simple Knowledge Organization System Reference. W3C Recommendation, W3C. `http://www.w3.org/TR/skos-reference/`.

[Möller, 2009] Möller, K. (2009). *Lifecycle Support for Data on the Semantic Web*. PhD thesis, National University of Ireland, Galway (NUIG).

[Möller et al., 2005] Möller, K., Breslin, J. G., and Decker, S. (2005). semiBlog — Semantic Publishing of Desktop Data. In *Proceedings of the 14th Conference on Information Systems Development (ISD2005)*.

[Monge and Elkan, 1996] Monge, A. and Elkan, C. (1996). The field matching problem: Algorithms and applications. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*.

[Myers et al., 2007] Myers, K., Berry, P., Blythe, J., Conley, K., Gervasio, M., Mcguinness, D., and Morley, D. (2007). An Intelligent Personal Assistant for Task and Time Management. *AI Magazine*, 28(2).

[Nadeen and Sauermann, 2007] Nadeen, D. and Sauermann, L. (2007). From Philosophy and Mental-Models to Semantic Desktop research: Theoretical Overview. In *Proceedings of I-Semantics 2007*.

[Nelson, 1965] Nelson, T. H. (1965). Complex Information Processing: A File Structure for the Complex, the Changing and the Indeterminate. In *Proceedings of the 20th National Conference (ACM1965)*.

[Nelson, 1973] Nelson, T. H. (1973). A Conceptual Framework for Man-Machine Everything. In *Proceedings of the National Computer Conference and Exposition (AFIPS 1973)*.

[Nelson, 1974] Nelson, T. H. (1974). *Computer Lib / Dream Machines*. Microsoft Press, Redmond, WA, USA.

[Nelson, 2004] Nelson, T. H. (2004). A Cosmology for a Different Computer Universe: Data Model, Mechanisms, Virtual Machine and Visualization Infrastructure. *Journal of Digital Information*, 5(1).

[Nikolov et al., 2012] Nikolov, A., d'Aquin, M., and Motta, E. (2012). Unsupervised learning of link discovery conguration. In *Proceedings of the 9th Extended Semantic Web Conference (ESWC 2012)*.

[Nilsson et al., 2008] Nilsson, M., Powell, A., Johnston, P., and Naeve, A. (2008). Expressing Dublin Core metadata using the Resource Description Framework (RDF)0. DCMI Recommendation, DCMI. `http://dublincore.org/documents/dc-rdf/`.

[Noh, 2006] Noh, S. (2006). The Measurable Belief of Trust in Social Networks. In *Proceedings of the Semantic Desktop and Social Semantic Collaboration Workshop (SemDesk 2006)*.

[Norman, 1983] Norman, D. A. (1983). *Some Observations on Mental Models*. Lawrence Erlbaum Associates.

[Nunes, 2005] Nunes, D. A. (2005). HyperDE: A Framework and Development Environment Driven by Ontologies for Hypermedia Applications. Master thesis, Department of Informatics, PUC-Rio.

[Nunes and Schwabe, 2006] Nunes, D. A. and Schwabe, D. (2006). Rapid prototyping of web applications combining domain specific languages and model driven design. In *Proceedings of the 15th International Conference on World Wide Web (WWW2006)*.

[Nyce and Kahn, 1991] Nyce, J. M. and Kahn, P., editors (1991). *From Memex to Hypertext: Vannevar Bush and the Mind's Machine*. London Academic Press.

[Oren, 2005] Oren, E. (2005). SemperWiki: a Semantic Personal Wiki. In *Proceedings of the 1st Workshop on The Semantic Desktop*.

[OSA Foundation, 2012] OSA Foundation (2012). The Chandler Project. `http://chandlerproject.org/`.

[Osterfeld et al., 2005] Osterfeld, F., Kiesel, M., and Schwarz, S. (2005). Nabu — A Semantic Archive for XMPP Instant Messaging. In *Proceedings of the 1st Workshop on The Semantic Desktop*.

[Otlet, 1934] Otlet, P. (1934). *Traité de Documentation. Le Livre sur le Livre. Théorie et Pratique*. Editiones Mundaneum.

[Otlet and Rayward, 1990] Otlet, P. and Rayward, W. B. (1990). *International Organisation and Dissemination of Knowledge : Selected Essays of Paul Otlet*.

[Paivio, 1986] Paivio, A. (1986). *Mental Representations: A Dual Coding Approach*. Oxford Psychology Series. Oxford University Press.

[Park, 2006] Park, J. (2006). Promiscuous Semantic Federation: Semantic Desktops Meet Web 2.0. In *Proceedings of the Semantic Desktop and Social Semantic Collaboration Workshop (SemDesk 2006)*.

[Payne, 2003] Payne, S. J. (2003). *Users' Mental Models: The Very Ideas*, pages 135–154. Morgan Kaufman Publishers.

[Prud'hommeaux and Seaborne, 2008] Prud'hommeaux, E. and Seaborne, A. (2008). SPARQL query language for RDF. W3C Recommendation, W3C. `http://www.w3.org/TR/rdf-sparql-query/`.

[Quan and Karger, 2004] Quan, D. and Karger, D. R. (2004). How to make a Semantic Web browser. In *Proceedings of the 13th International World Wide Web Conference (WWW2004)*.

[Quan et al., 2003] Quan, D., Karger, D. R., and Huynh, D. (2003). RDF Authoring Environments for End Users. In *Proceedings of the International Workshop on Semantic Web Foundations and Application Technologies*.

[Raimond et al., 2008] Raimond, Y., Sutton, C., and Sandler, M. (2008). Automatic Interlinking of Music Datasets on the Semantic Web. In *Proceedings of the Linked Data on the Web Workshop (LDOW2008)*.

[Rayward, 1991] Rayward, W. B. (1991). The Case of Paul Otlet, Pioneer of Information Science, Internationalist, Visionary. Reflections on Biography. *Journal of Librarianship and Information Science*, 23.

[Reif et al., 2006] Reif, G., Gall, H., and Morger, M. (2006). Semantic Clipboard — Semantically Enriched Data Exchange Between Desktop Applications. In *Proceedings of the Semantic Desktop and Social Semantic Collaboration Workshop (SemDesk 2006)*.

[Reif et al., 2008] Reif, G., Groza, T., Scerri, S., and Handschuh, S. (2008). Final NEPOMUK Architecture. Technical report.

[Rhodes and Starner, 1996] Rhodes, B. J. and Starner, T. (1996). Remembrance Agent: A Continuously Running Automated Information Retrieval System. In *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi Agent Technology (PAAM '96)*.

[Richter et al., 2005] Richter, J., Volkel, M., and Haller, H. (2005). Deepamehta-a semantic desktop. In *Proceedings of the 1st Workshop on The Semantic Desktop*.

[Ringel et al., 2003] Ringel, M., Cutrell, E., Dumais, S., and Horvitz, E. (2003). Milestones in Time: The Value of Landmarks in Retrieving Information from Personal Stores. In *Proceedings of the International Conference on Human Computer Interaction (INTERACT2003)*.

[Saïs et al., 2007] Saïs, F., Pernelle, N., and Rousset, M.-C. (2007). L2R: A Logical Method for Reference Reconciliation. In *Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI'07)*.

[Samp et al., 2008] Samp, K., Basca, C., McDaniel, B., and Decker, S. (2008). Atom Interface—A Novel Interface for Exploring and Browsing Semantic Space. In *Proceedings of the Semantic Web User Interaction Workshop, Exploring HCI Challenges (SWUI2008)*.

[Sauermann, 2003] Sauermann, L. (2003). The Gnowsis — Using Semantic Web Technologies to Build a Semantic Desktop. Diploma thesis, Technical University of Vienna.

[Sauermann, 2005a] Sauermann, L. (2005a). The Gnowsis Semantic Desktop for Information Integration. In *Proceedings of the 3rd Conference Professional Knowledge Management (WM2005)*.

[Sauermann, 2005b] Sauermann, L. (2005b). The Semantic Desktop — a Basis for Personal Knowledge. In *Proceedings of I-KNOW2005*.

[Sauermann, 2008] Sauermann, L. (2008). Evaluating Long-term Use of the Gnowsis Semantic Desktop for PIM. In *Proceedings of the 7th International Semantic Web Conference (ISWC2008)*.

[Sauermann, 2009] Sauermann, L. (2009). *The Gnowsis Semantic Desktop Approach to Personal Information Management*. PhD thesis, Fachbereich Informatik der Universität Kaiserslautern.

[Sauermann et al., 2006] Sauermann, L., Grimnes, G. A., Kiesel, M., Fluit, C., Maus, H., Heim, D., Nadeem, D., Horak, B., and Dengel, A. (2006). Semantic Desktop 2.0: the Gnowsis Experience. In *Proceedings of the 5th International Conference on The Semantic Web (ISWC2006)*.

[Sauermann et al., 2009] Sauermann, L., Van Elst, L., and Möller, K. (2009). Personal Information Model (PIMO). Technical report.

[Scerri et al., 2009] Scerri, S., Davis, B., Handschuh, S., and Hauswirth, M. (2009). Semanta — Semantic Email Made Easy. In *Proceedings of the 6th European Semantic Web Conference (ESWC2009)*.

[Schaffert et al., 2006] Schaffert, S., Westenthaler, R., and Gruber, A. (2006). IkeWiki: A User-friendly Semantic Wiki. In *Proceedings of the 3rd European Semantic Web Conference (ESWC2006)*.

[Schutz, 2008] Schutz, A. T. (2008). Keyphrase Extraction from Single Documents in the Open Domain Exploiting Linguistic and Statistical Methods. Master thesis, National University of Ireland, Galway (NUIG).

[Schwabe et al., 2005] Schwabe, D., Brauner, D., Nunes, D. A., and Mamede, G. (2005). HyperSD: a Semantic Desktop as a Semantic Web Application. In *Proceedings of the 1st Workshop on The Semantic Desktop*.

[Schwitter, 2005] Schwitter, R. (2005). Controlled Natural Language as Interface Language to the Semantic Web. In *Proceedings of the 2nd Indian International Conference on Artificial Intelligence (IICAI'05)*.

[Seaborne, 2004] Seaborne, A. (2004). RDQL – A Query Language for RDF. W3C Member Submission, W3C.

[Sintek et al., 2007] Sintek, M., van Elst, L., Scerri, S., and Handschuh, S. (2007). Distributed Knowledge Representation on the Social Semantic Desktop: Named Graphs, Views and Roles in NRL. In *Proceedings of the 4th European Semantic Web Conference (ESWC2007)*.

[Smith et al., 2005] Smith, D. A., Owens, A., Russell, A., Harris, C., and Wilson, M. (2005). The Evolving mSpace Platform : Leveraging the Semantic Web on the Trail of the Memex. *Human Factors*.

[Story et al., 2009] Story, H., Harbulot, B., Jacobi, I., and Jones, M. (2009). FOAF+SSL: RESTful Authentication for the Social Web. In *Proceedings of the First Workshop on Trust and Privacy on the Social and Semantic Web (SPOT2009)*.

[Takeda and Ohmukai, 2005] Takeda, H. and Ohmukai, I. (2005). Semblog Project. In *Proceedings of the Activities on Semantic Web Technologies Workshop*.

[Tummarello et al., 2007] Tummarello, G., Delbru, R., and Oren, E. (2007). Sindice.com: Weaving the Open Linked Data. In *Proceedings of the 6th International Semantic Web Conference (ISWC2007)*.

[van Elst et al., 2008] van Elst, L., Kiesel, M., Schwarz, S., Buscher, G., Lauer, A., and Dengel, A. (2008). Contextualized Knowledge Acquisition in a Personal Semantic Wiki. In *Proceedings of the 16th International Conference on Knowledge Engineering and Knowledge Management (EKAW2008)*.

[Varadarajan et al., 2005] Varadarajan, A., Patel, N., and Grosky, W. (2005). Semantic Pen — A Personal Information Management System for Pen Based Devices. In *Proceedings of the 1st Workshop on The Semantic Desktop*.

[Veith, 2006] Veith, R. H. (2006). Memex at 60: Internet or iPod?: Research Articles. *Journal of the American Society for Information Science and Technology*, 57(9).

[Wang et al., 2005] Wang, G., Wang, B., Han, D., and Qiao, B. (2005). Design and Implementation of a Semantic Document Management System. *Information Technology Journal*, 4(1).

[Whittaker et al., 2000] Whittaker, S., Terveen, L., and Nardi, B. A. (2000). Let's Stop Pushing the Envelope and Start Addressing It: A Reference Task Agenda for HCI. *Human-Computer Interaction*, 15(2).

[Woerndl and Hristov, 2009] Woerndl, W. and Hristov, A. (2009). Recommending Resources in Mobile Personal Information Management. In *Proceedings of the Third International Conference on Digital Society (ICDS2009)*.

[Woerndl and Schulze, 2009] Woerndl, W. and Schulze, F. (2009). Location-Awareness in a Mobile Semantic Desktop Application. In *Proceedings of the Fifth International Conference on Signal Image Technology and Internet Based Systems (SITIS2009).*

[Woerndl and Woehrl, 2008] Woerndl, W. and Woehrl, M. (2008). SeMoDesk: Towards a Mobile Semantic Desktop. In *Proceedings Personal Information Management Workshop (PIM2008).*

[Xiao and Cruz, 2005] Xiao, H. and Cruz, I. F. (2005). A Multi-ontology Approach for Personal Information Management. In *Proceedings of the 1st Workshop on The Semantic Desktop.*

[Xiao and Cruz, 2006] Xiao, H. and Cruz, I. F. (2006). Application Design and Interoperability for Managing Personal Information in the Semantic Desktop. In *Proceedings of the Semantic Desktop and Social Semantic Collaboration Workshop (SemDesk 2006).*

[Yee et al., 2003] Yee, K.-P., Swearingen, K., Li, K., and Hearst, M. A. (2003). Faceted Metadata for Image Search and Browsing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.*

[Zhang and Shen, 2005] Zhang, X. and Shen, W. (2005). WonderDesk-A Semantic Desktop for Resource Shating and Management. In *Proceedings of the 1st Workshop on The Semantic Desktop.*