A Semantic Web based approach to expertise finding at KPMG

Master's Thesis, August 31 2010

Ernst Alexander Jansen

A Semantic Web based approach to expertise finding at KPMG

THESIS

submitted in partial fulfillment of
the requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE
TRACK INFORMATION ARCHITECTURE

by

Ernst Alexander Jansen
born in ’s-Gravenhage, the Netherlands

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

A Semantic Web based approach to expertise finding at KPMG

Author: Ernst Alexander Jansen
Student id: 1129945
Email: alexanderjansen@gmail.com

Abstract

An important requirement for realizing high organizational effectiveness within large organizations is the existence of an expertise finding application which enables rapid discovery of expertise inside the organization. The main goal of this research was to investigate how Semantic Web techniques could be used to increase the findability and accessibility of the employee expertise information inside organizations. Techniques from the Semantic Web were chosen because they are particularly suitable for integrating different data sets and allow the application to be easily extended in the future. A skills ontology has been created which defines skills from the IT Advisory domain and relates them to each other. The skills ontology is used in a proof of concept expertise finding application to illustrate that this background knowledge can be used to increase the findability and accessibility of the employee information. The employee information is expressed with RDF and the skills ontology is constructed with OWL 2 DL and contains SWRL rules which are used to infer relations between employees and skills.
To demonstrate how the constructed application can be extended with external knowledge sources, a second skills ontology has been constructed and linked to the first ontology. The two ontologies can now be used jointly to enhance the discovery of expertise in the organization

Keywords: expertise finding, skills ontology, semantic web.

Preface

This document has been produced in the master thesis project as part of the Master program Information Architecture at Delft University of Technology (TU-Delft). The work on the project took place at KPMG in Amstelveen.

KPMG is one of the largest professional services firms in the world and employs over 140,000 people. KPMG has three lines of services: audit, tax, and advisory. I was situated at the IT Advisory department were I shared a room with junior advisors which were able to provide answers to all my questions about KPMG, related and unrelated to this project. I really enjoyed doing my thesis project in this stimulating environment and I can recommend an internship at KPMG to everyone.

This thesis project could not have been completed without the help and support of the members of the graduation committee which I would like to thank now. First of all, Professor Geert Jan Houben who was always there to read through the draft versions of the report and was able to provide detailed comments and critical notes about my ideas. My daily supervisor, assistant professor Laura Hollink, who was willing to spend hours discussing about the pro's and con's of the different variants of the data model and was furthermore providing extensive feedback in the last stage of the project. Ir. Hans Geers, who was willing to read the thesis and provide feedback. I would also like to thank my daily supervisor at KPMG, Ruben de Wolf, who arranged all the resources that I needed to complete this project. I really appreciate the time and effort he has invested in the discussions, explanations, and the evaluation of my thesis report.

I hope you enjoy reading my work!

Ernst Alexander Jansen
Delft, the Netherlands
August 31, 2010

Contents

Chapter 1 - Introduction

This chapter introduces the problem situation which is addressed by this thesis. The problem is demarcated and a main research goal and research questions are formulated. Finally the structure of the report is discussed.

1.1 - Motivation

KPMG is a knowledge intensive organization where the network of the individual employee plays an important role in the process of expertise finding (defined in section 2.1). Because KPMG extends rapidly and because the growing amount of international projects there is need for an application which helps employees locate expertise inside the organization.
The expertise finding applications that are currently available inside the organization of KPMG are barely used by the employees. KPMG presumes that this is related to the fact that these applications contain mostly outdated information. It is the responsibility of the employees to keep their personal expertise information up to date but the experience is that they do not devote time on maintaining their expertise information. KPMG thinks this is probably related to the full agenda's of the employees. They would like a solution for this problem and indicate that an ideal expertise finding application would be an application which:

- contains complete and up to date (expertise) information about each employee in the organization,
- is able to assist employees with maintaining their personal information,
- and is able to help employees in their search for colleagues with certain expertise.

The next section discusses the approach used in this research.

1.2 - Approach

The approach used in this project is visualized in the image shown on the right. KPMG identified a problem related to their expertise finding applications and first of all this problem should be investigated. Chapter 1 discusses this exploratory phase where the problem situation is analyzed by inspecting the expertise finding applications and by discussing the shortcomings of these applications with the users (the employees of KPMG). After this stage, at the end of chapter 1, project goals can be formulated. Relevant literature can be studied to understand the tools, techniques and related cases. These results are discussed in chapter 2.

With the knowledge gathered from the literature the proof of concept (PoC) application can be designed. Whenever subjects are not clear answers should be searched in literature. This process is discussed in chapter 3. Chapter 4 discusses the implementation phase of the application. When the application has been build there should be checked if the project goals have been met. If not, the application should be adjusted / improved. These steps are discussed in chapter 5.

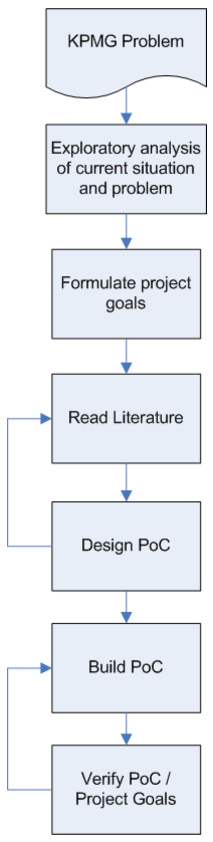

KPMG Problem
Exploratory analysis of current situation and problem
Formulate project goals
Read Literature
Design PoC
Build PoC
Verify PoC / Project Goals

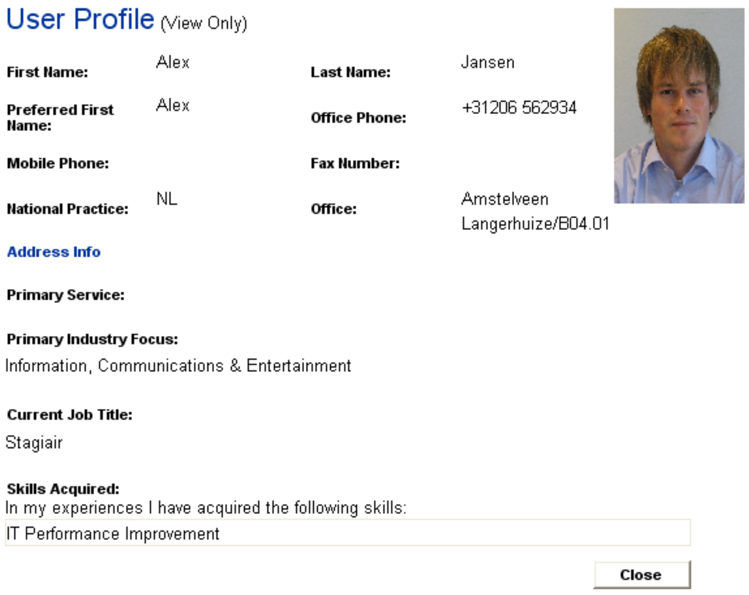
Figure 1.1 - Approach

1.3 - Problem Situation & Analysis

After KPMG explained the problems related to their expertise finding applications an exploratory research was conducted in order to understand the problem situation thoroughly. This exploratory phase started with an inspection of KPMG's expertise finding applications. There are two expertise finding applications available and they can be accessed on the Intranet. They are called 'Skills & Experience' and 'My Site'.

1.3.1 - Skills & Experience

The 'Skills & Experience' application is an expertise finding application which was launched a couple of years ago. This application can be accessed on the Intranet and allows employees to define their skills and experience. Employees have to select their skills and experience from predefined lists and there are some 'free-text sections' which allow for giving a more detailed description of their expertise. A search module enables employees to find their colleagues by searching for certain properties like skills, department, industry focus, etc. Because the application cannot find information that has not been supplied by the employees, all employees were asked to fill out their skills and experience when the application was launched. Still, the success of this application was limited, mainly because employees did not start to use the application.



User Profile (View Only)

First Name:	Alex	Last Name:	Jansen
Preferred First Name:	Alex	Office Phone:	+31206 562934
Mobile Phone:		Fax Number:	
National Practice:	NL	Office:	Amstelveen Langerhuize/B04.01

Address Info

Primary Service:

Primary Industry Focus:
Information, Communications & Entertainment

Current Job Title:
Stagiair

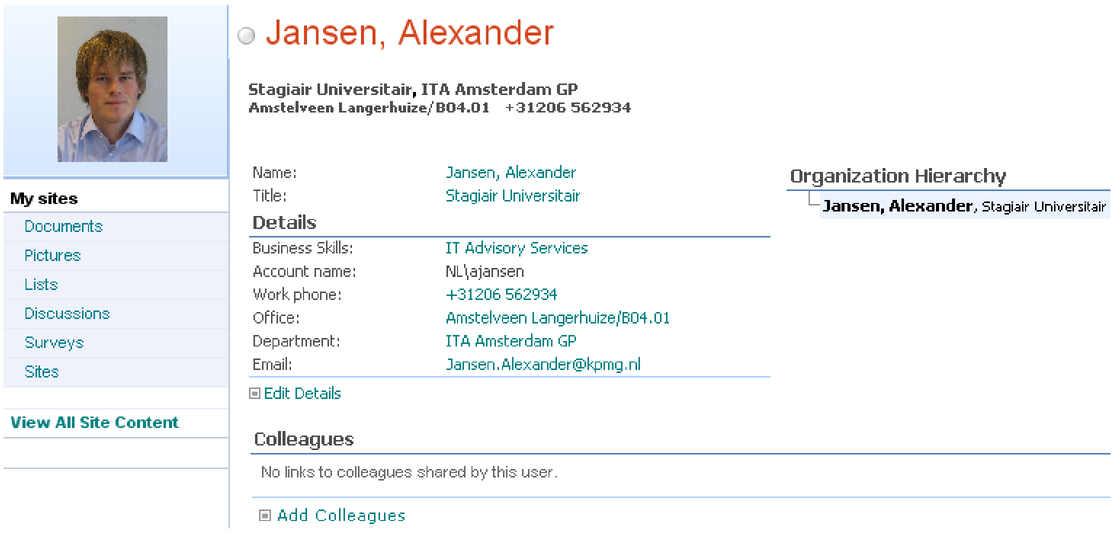
Skills Acquired:
In my experiences I have acquired the following skills:
IT Performance Improvement

Figure 1.2 – KPMG Skills & Experience

1.3.2 - My Site

A few months ago the 'My Site' application was launched. My Site can be seen as the new Skills & Experience application with a social networking approach (like LinkedIn). All employees have their personal page on the Intranet where they can describe themselves. This description includes expertise information like skills, experience, project participation, and colleagues. Their personal page can be completed by adding a photo and an external CV document. My Site comes with a search functionality which allows you to find colleagues with expertise, similar to the search module of the (older) Skills & Experience application. The way in which employees have to define their skills and experience is also similar to the Skills &

Experience application, they have to select their expertise from predefined lists or describe it using free text fields.



Jansen, Alexander

Stagiair Universitair, ITA Amsterdam GP
Amstelveen Langerhuize/B04.01 +31206 562934

My sites
- Documents
- Pictures
- Lists
- Discussions
- Surveys
- Sites

View All Site Content

Name:	Jansen, Alexander
Title:	Stagiair Universitair

Details

Business Skills:	IT Advisory Services
Account name:	NL\ajansen
Work phone:	+31206 562934
Office:	Amstelveen Langerhuize/B04.01
Department:	ITA Amsterdam GP
Email:	Jansen.Alexander@kpmg.nl

Edit Details

Organization Hierarchy

Jansen, Alexander, Stagiair Universitair

Colleagues

No links to colleagues shared by this user.

Add Colleagues

Figure 1.3 – KPMG My Site

1.3.3 - Limitations of KPMG's Expertise Finding Applications

The skills & experience application was never a success because employees did not start to use the application. The new My Site application has just been launched and its functionality is very similar to the skills & experience application. Will My Site, with its social networking approach, be more successful? While this will become clear over time it seemed interesting to look more closely at the reasons why the skills & experience application was not used by the employees of KPMG. From discussions with employees it became apparent that the main reasons for not using the application were:

1. The application contained outdated information because employees did not update their expertise information. This meant that when you searched for someone with certain expertise, only outdated information could be retrieved.
2. The performance of the search module was bad. Even when you knew what you were searching for, it was hard to get the right results.
3. The way in which you had to describe your expertise information was pretty unclear: What does a certain skill in a predefined list actually mean?

Points 2 and 3 are both problems which can be improved by the designers of the application. The search of the new My Site application performs better than the search of the old Skills & Experience application, but it can still use some improvements (especially on user friendliness). The way in which employees define their expertise information (skills, experience, etc.) in the new My Site application is pretty much the same as in the old Skills and Experience application. Employees have to select their skills and experience from predefined lists. These lists are alphabetically ordered and contain over 2500 skills which do not have a clear meaning and can thus be interpreted in multiple different ways. Employees indicate that this is a point which has to be improved too.

To improve point 1, the outdated information problem, the behavior of employees should change. When employees were asked for the reason that they did not update their personal information in the expertise finding applications, their reaction was unanimous:

- Updating personal information costs valuable time.
- Employees who keep their expertise information up-to-date do not receive more requests to participate in client projects.

The first argument does not need any explanation, but the second one does.
Finding the right employee for the tasks at hand requires a thorough understanding of the expertise of the employee. The resource planning systems currently in use within KPMG fall short here. Until now, the personal networks of the employees were sufficient to locate the persons with the appropriate expertise. It was therefore not necessary to devote time on updating your personal information in an expertise finding application. In times like these where the organization is expanding rapidly and the amount of international projects is increasing even faster, personal networks are expanding quickly too. At the same time people are developing new skills and experience rapidly. It is therefore important that employees are stimulated to define their expertise in expertise finding application because these applications are perfectly able to keep track of the skills and experience of the people in personal networks and they can be of great value whenever certain expertise is needed which is not available in the personal network.

This section discussed the problem situation and it should be clear that the problems around the expertise finding applications of KPMG are not just related to the applications, but also to the culture of the organization. The next section discusses where the focus of this research will be.

1.4 - Focus & Demarcation

The previous section explained that the current expertise finding applications contain outdated information, confusing ways to describe expertise, and a bad search performance. As a result, nobody uses the applications.
To change this situation all three of the mentioned points should be attended. First of all the application should have a very intuitive input module with the meaning of the expertise descriptors unambiguously defined. It should furthermore allow employees to define their expertise in a time-inexpensive way and it should have an easy to use search module that is able to retrieve the information that employees are looking for. Last but not least, the information which can be retrieved should be correct, complete and up to date.

Due to time limitations this project cannot create this ideal application. Therefore the choice has been made to focus on increasing the search performance while at the same time creating a solid base which allows for further improvements on the other points.

One of the reasons that the search performance of the current expertise finding applications is limited is the fact that they only use syntactic keyword search. Improved search performance could be achieved by extending this to also incorporate semantics. Semantic Web tools and techniques can be used for this purpose. This can result in improved search performance of the expertise finding applications. To

accomplish this background knowledge about the employee information will be added to the system. The background knowledge that will be added in this project consists of the skills which are available inside the organization. The meaning of skills, and the relations between skills, will be defined in an ontology. Whenever the ontology defines that certain skills are similar, or related, to each other the system will be able to exploit these relations when searching for employees with certain skills. With this kind of background knowledge it is possible that a search for employees with a certain skill returns employees who did not tell the system (explicitly) that they possessed this specific skill. The system was able to infer that they posses this specific skill based on the skills they did supply and the skills ontology which defines the skills and relations between them (chapter 3 discusses this in more detail).
This background knowledge (the skills ontology) can only be exploited whenever the employee information is represented in a way that allows for this exploitation, such as RDF (Resource Description Framework, see section 2.2.4). Currently all the employee information is stored inside relational databases and CV documents. This information has to be expressed with RDF. Expressing the employee information with RDF creates an extreme flexible and extendable situation. While relational databases try to capture the world inside tables, RDF expresses its data as a graph. When RDF is used for data storage, data can be used for all kinds of different goals (applications) and no conversions are needed, while relational databases are designed for a certain goal (application) and whenever the data inside these databases is needed for another goal (application), it could very well be that the databases are not suitable (designed) for this new goal, and thus need to be converted.

RDF and ontologies are techniques from the Semantic Web. Chapter two discusses the Semantic Web and its techniques in more detail and chapter three explains how these techniques are used to improve the search performance.

The goal of this research is thus to improve the findability and accessibility of the employee information by expressing the employee information in RDF, which allows for adding background knowledge. A proof of concept application will be created to illustrate the improved search performance. The flexibility and extendability of the approach will be demonstrated by showing the easiness of coupling and combining different background knowledge repositories (ontologies).

Some further demarcations on the proof of concept application that will be created are shown below.

- My Site contains over 2500 skills which can be used to describe expertise. The skills ontology that will be created will be focused on the IT Advisory department and will contain around 150 skills.
- KPMG has various systems in use that contain employee information. The two data sources that will be used for the proof of concept application are the Active Directory, which contains basic information about each employee, and the IT Advisory CV database, which contains CV documents of the employees of the IT Advisory department.
- Information extraction from text files is a complex problem. A separate research could be entirely dedicated to the subject of creating reliable expertise extraction from the CV documents. Therefore this research makes assumptions about the format and language used in the CV documents.

- The information that can be retrieved with the application will consist of the information that is available inside the two mentioned data sources. Something that should be mentioned is that most of the expertise information that is available inside the CV documents is outdated.

1.5 - Research Goal

From the previous section the following main research question and sub questions can be derived.

Main Research Question:
How can Semantic Web techniques improve the accessibility and the findability of KPMG employee information, in the area of expertise finding?

Sub Questions:
1. *How can background knowledge about the skills possessed by the KPMG employees be exploited to improve expertise finding within KPMG.*

The employee instance data that is stored in the KPMG data sources will be combined and expressed in RDF. This can be seen as a translation from one storage format into the other. How can the skills ontology improve the accessibility and findability of this employee information (expressed in RDF).

2. *Is the chosen approach flexible and extendable?*

Because the discussed proof of concept application addresses only one of the problems KPMG experiences with its expertise finding applications, the constructed application should be extendable and flexible so that it can easily be extended in the future. Semantic Web techniques are intended for integrating various datasets which means that they should create a flexible and extendable solution. It should therefore be investigated to what extent the resulting application is indeed extendable and flexible.

1.6 - Report Structure

Chapter one introduces the problem situation, the goals of the research project, and the approach used. Chapter two discusses relevant literature and chapter 3 discusses the design of the application. Chapter four covers the implementation phase and discusses all the problems and discoveries done during implementation. Chapter five presents the results. Can the constructed proof of concept application help in the process of expertise finding and to what extent is it flexible and extendable. The conclusion and recommendations can be found in chapter 6.

Chapter 2 – Expertise Finding with Semantic Web Techniques

This chapter discusses the most important literature that has been studied during this research project. The discussed topics will be used throughout the research project.

2.1 - Knowledge Management & Expertise Finding

The term expertise finding was used throughout the introduction chapter (chapter one) and this section explains what expertise finding is.

“knowledge is power” (scientia potentia est), as originally stated by the Persian poet Ferdowsi (940-1020), is a statement that still holds nowadays. It especially holds for large organizations that contain lots of knowledge. When people in organizations need to solve complex problems they rely on their own knowledge and experience, information available in the organization, and knowledge and experience of others in the organization. [1] Therefore it is important that the knowledge available inside the organization can easily be located. The larger and more geographically distributed an organization is, the more difficulties there will be with locating the desired knowledge. [2] Maybury confirms these statements in [3] and adds that the ability to rapidly discover individual experts or documented knowledge is an essential element of organizational effectiveness. Balog, Azzopardi, and de Rijke state that the ability to find the persons in an organization with the appropriate skills and knowledge is crucial for the success of a project. [4] It is therefore important that organizations try to achieve efficient and effective expertise finding.
This section discusses the concept of expertise finding, which is part of the broader concept knowledge management.

In [1], Fazel-Zarandi and Yu state that knowledge management systems can help with the problem of locating knowledge inside organizations by suggesting employees who possess the required knowledge. They explain that these systems work with profiles containing competences and experiences of the employees. The difference between a knowledge management system and an expertise finding system is explained by Yimam-Seid and Kobsa in [5], where they argue that a knowledge management system provides access to knowledge in all forms, while an expert finding system locates experts who possess certain knowledge. The expert finding functionality is thus part of the knowledge management system.

While the term expert was introduced in the section above, no definition was given. It is noteworthy that many organizations refrain from using the term expert because of the legal consequences that come with stating that someone is an expert. In [6], McDonald and Ackerman define expertise as:
The knowledge and skills an individual has

This is in line with the definition of the Oxford English Dictionary:
Great skill or knowledge in a particular field

These definitions illustrate that an expert in a particular area has expertise in this particular area. What an expert finding system tries to do is locate the persons who possess knowledge and skills in a certain area.

Chapter one explained the problem situation and introduced the goals of this research project. There was explained that the main research goal is to improve the accessibility and findability of the employee information, which can be achieved by expressing the employee information with RDF and enriching this information with an ontology. RDF and ontologies are techniques from the Semantic Web which is discussed in more detail below.

2.2 – The Semantic Web

The original World Wide Web, invented by Sir Tim Berners-Lee, allows humans to find information by browsing documents which are linked to other documents. People can add documents to the Web and add links between the documents. The content of this web of documents is hard to understand for machines. In 2001 Tim Berners-Lee defined the Semantic Web, an extension of the current Web aimed at machine-processable information. In the Semantic Web information is given well-defined meaning enabling computers to understand the information. [7] This results in a more powerful Web because computers are able to perform complex tasks like finding, sharing, and combining information. [8] The World Wide Web consortium (W3C) defines the Semantic Web, also called the web of data, as follows [9]:

"The vision of the Semantic Web is to extend principles of the Web from documents to data. Data should be accessed using the general Web architecture using, e.g., URI-s; data should be related to one another just as documents (or portions of documents) are already. This also means creation of a common framework that allows data to be shared and reused across application, enterprise, and community boundaries, to be processed automatically by tools as well as manually, including revealing possible new relationships among pieces of data."

So, the Semantic Web is an extension of the current World Wide Web in which computers are able to understand the data and reason with it. It provides a framework for syntax independent structuring of data and it offers mechanisms to define relationships between structures. The Semantic Web is build up from separate layers which together form the Semantic Web stack.

2.2.1 - The Semantic Web Stack

The figure below was created by Tim Berners-Lee and shows the layers of the Semantic Web, also known as the Semantic Web Stack. [10]

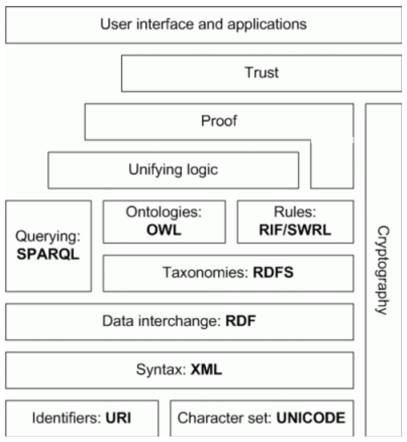


Figure 2.1 - Semantic Web Stack [10]

In this hierarchy each layer uses the capabilities from the layers below. It shows that the Semantic Web is an extension of the current Web and that Semantic Web applications are build upon the Semantic Web stack. [10]
The bottom layers contain techniques from the Hypertext Web. The Unicode layer makes sure that an international character set is used and the URI layer enables identification of objects. The XML layer provides syntax for content structuring of documents. [11] The middle layers contain technologies that enable the building of Semantic Web applications. With the RDF (Resource Description Framework) layer it is possible to represent information about objects (resources) and their relationships. RDFS (RDF Schema) is used to create taxonomies by defining classes and properties of RDF-based objects. OWL (Web Ontology Language) extends RDFS by adding more advanced vocabulary for describing classes and properties. OWL is based on description logic and allows for reasoning over data. SPARQL is a language for querying RDF data and is used to retrieve information. Rules can be used to extent the expressivity of OWL and can be defined with RIF (Rule Interchange Format) or SWRL (Semantic Web Rule Language). [11; 12; 13]
The web principle "anyone can say anything about anything" makes the web a unique information source, but it is important that there are some mechanisms to verify the source of information to be able to decide whether or not to trust the information. A proof mechanism is needed to verify that a certain claim is valid. The top layers of the Semantic Web stack, unifying logic, proof, and trust, together with the cryptography layer, enable these functionalities but are not standardized yet. These functionalities are especially important for the development of Semantic Web applications, because they are needed to enable machines to decide which information source to trust, and which to choose whenever multiple sources with varying quality are available. [14]

Now follows a more detailed description of the techniques from the Semantic Web stack.

2.2.2 URI and Unicode

The bottom layer of the Semantic Web stack contains the building blocks of the Semantic Web; Unicode and URI. Unicode is the standard for computer character representation and contains all the symbols used by modern writing systems. [15] A URI (Uniform Resource Identifier) is used to uniquely identify resources, where a resource can be a document, a person, a skill, a department, or any other entity. Each

URI points to a unique location where the resource is described. [16; 17] Within the proof of concept application URI's will be used to identify different resources like skills and employees.

2.2.3 XML

The next layer in the Semantic Web stack is the XML layer. XML stands for eXtensible Markup Language and is a free and open standard designed for data structuring, storage, and transport. It is easy to use, flexible, self-descriptive, and machine-readable. In contrast to HTML which is about displaying information, XML is about carrying information. Because XML allows data exchange in a standard format, independent of storage, it has become the de-facto standard for representing metadata descriptions of resources. [18; 19]
The author of the XML document is free to define the tags and structure (in contrast to HTML which has predefined tags). The problem with XML is that it is a syntactic and structural language, but that there is no possibility to describe the semantics. How is a one supposed to know if two identical tags refer to the same thing, or that two different tags refer to the same thing? [19]

2.2.4 - RDF

RDF is intended for describing information about resources on the World Wide Web. It is a collection of triples and each triple consists of a subject, a predicate, and an object, which together form a statement. Usually, both the subject and the object are URIs of certain resources, but the object can also be a literal value. [13; 20] The subject denotes the resource, the predicate denotes the relationship between the subject and the object, and the object is the object of the predicate. For example: "Alex is member of the IT Advisory department" would be expressed as the triple:
- subject: "Alex" (resource)
- predicate: "is member of" (relation)
- object: "the IT Advisory department" (resource)

XML authors are free to define their own data structure, but with RDF data is always stored as triples. This results in more easy construction of mappings because only statements need to be mapped (XML requires you to consider the nested structure of the files invented by the authors).
While RDF helps to give structure to web content, it does not define very strong semantics to describe the content itself. Ontologies are needed to describe content. [7; 21] RDFs and OWL are ontology languages built upon RDF which offer extended functionalities like class hierarchies.

2.2.5 - RDFS

RDF Schema (RDFS) allows simple class hierarchies (taxonomies) and relations to be defined between classes in order to structure the RDF resources. This results in clearer semantics. A problem with RDFS is that there is no way to declare whenever two statements are equal. Also, with RDFS it is not possible to define more advanced relations between classes. [22]

2.2.6 - OWL

Ontologies are an important part of the Semantic Web. Ontologies capture the knowledge about a certain domain of the real world and are used to unambiguously define these concepts and the relationships between them. [23] They provide a common understanding of information and enable reuse of domain knowledge. [7; 24]
The Web Ontology Language (OWL) builds on RDF and RDFS and has a greater expressive power. It offers a larger vocabulary and stronger syntax. In OWL, more advanced relations between classes (disjoint, union, intersection, and equality) and more advanced statements about properties can be made (symmetry, transitive, inverse). OWL allows other ontologies to be imported, while RDFS only allows referencing them. [23]
There are three variants (species) of OWL; Lite, DL, and Full. OWL Lite offer easier reasoning against less expressive power, while OWL Full offers the most expressive power with harder reasoning. [25] OWL DL will be used for the skills ontology (section 4.4).

2.2.7 - Rules

Ontologies and rules are the two key components of the Semantic Web. Rules can be used in ontology languages, either as an alternative too, or in conjunction with, description logics to extend the expressivity. Rules can also be used as a means to draw inferences, express constraints, react to events, and transform data. [26; 27]
In the Rules layer of the Semantic Web stack both RIF and SWRL are showed. SWRL is the Semantic Web Rule Language which was designed as an extension to OWL. In 2004, SWRL was proposed as W3C standard. It combines RuleML and OWL (DL and Lite). [28] RuleML (Rule Markup Language) is a language which uses XML to express rules. It enables the construction of queries and inferences in an ontology and it can be used to create mappings between ontologies. [26]
RIF (Rule Interchange Format) is another XML rule language. It was designed as an interchange format for exchanging rules between rule systems (for example, systems that implement SWRL). [29] Currently, RIF is a W3C standard in development. [30]
In the proof of concept application, SWRL will be used to extent the expressive power of OWL in the skills ontology. Rules will be used to infer relations between employees and skills.

2.2.8 - Reasoners

For the interpretation of ontology data and rules an appropriate inference mechanism is needed. These reasoners perform various inference services like computing class hierarchy and determining consistency of a class. They can be used to fill in incomplete or missing data by means of deduction and automated reasoning, making implicit data explicit. They can also be used for the mapping of concepts between ontologies. [7] This requires the use of a formal specification like description logic. This formal specification enables different reasoners to provide the same results when processing the same ontologies. [31] In the project the pellet reasoner is used to construct the inference model, based on the skills ontology with SWRL rules and the employee instance data (which defines which employee possesses what skills) (discussed in section 5.1).

2.2.9 - SPARQL

SPARQL stands for ‘SPARQL Protocol And RDF Query Language’. SPARQL is standardized by the W3C as the language to query RDF datasets. SPARQL usually refers to the RDF query language which is a syntactically-SQL-like language for querying RDF graphs. The SPARQL protocol is a method which can be used to remotely invocate SPARQL queries and it provides an interface to issue SPARQL queries against some endpoint (‘SPARQL endpoint’). [32] In the project SPARQL is used in the process of expertise finding, i.e., to discover which employees posses a certain skill(s).

Now that the tools and techniques have been discussed, there will be looked at some related cases from literature.

2.3 - Related Research

This section discusses two cases which have a similar goal as this research project. It is interesting to look at their approach to the problem.

2.3.1 – Skills management at Swiss Life

The implementation of an ontology based skills management system at the Swiss Life Group, the largest life insurance company of Switzerland, is described in [33]. The goal of this project was to construct a skills management system that would be able to find people with a certain skill profile. The system could then be used for staffing new projects or identifying experts who might help to solve a certain problem. RDFS and OIL were used to construct the skills ontology. At time of publication the skills ontology of the system was only a taxonomy without more advanced functionality. An interesting observation made in the publication is that the implementation of a skills management system requires one to address the technical, the content, and the cultural dimension. The technical dimension emphasizes the functionality of the system. The content dimension deals with setting up an automatic process for keeping the contents up-to-date, and the cultural dimension is concerned with ensuring a climate of trust and openness in which employees are motivated to make their skills known.

The ontology discussed in this publication only contains hierarchical relations. They mention that they are planning extensions to this hierarchy, but this was never published. The skills ontology that will be developed for KPMG will contain more relations than only the hierarchical ones. Furthermore, the ontology discussed here was constructed with a predecessor of OWL, which has a vocabulary that is not as powerful. Another remark is that this approach did not make attempts to reuse existing vocabularies for expressing the information. This concept is discussed below.

2.3.2 – Expertise Finding with Semantic Web Vocabularies

In 2007 Aleman-Meza et al. investigated how the most widely used vocabularies in the Semantic Web could be combined to create an expert finding framework. A combination of the FOAF, SIOC, and SKOS vocabularies were used for describing experts, their expertise, and their relations with other experts. [34]

The reason that existing vocabularies are used instead of constructing a new vocabulary from scratch is that they already attracted a considerable user community, which results in low entry barriers and direct applicability. More importantly, existing vocabularies cover a wide range of necessary features to adequately describe the expert finding domain, so there is no need to develop a new ontology. [34]
The components needed to describe an expert are: general descriptions, relations (to other experts), educational aspects, past and present activities and projects, and skills. Other components which can be used to describe experts, but which do not relate to a unique person, are: events, publications, opinions, ratings, recommendations and references. The popular vocabularies FOAF (Friend of a Friend) and SIOC (Semantically-Interlinked Online Communities) cover most of the components described above and are therefore selected in this approach.

The FOAF ontology describes people, what they create and do, and how they interact with others. The FOAF project started in 2000 as an "experimental linked information project". The FOAF website describes FOAF as a simple technology that makes it easy to share and use information about people and their activities, to transfer this information between Web sites, and to automatically extend, merge, and re-use it. [35] The authors state that expertise can be defined using the interests, publications, documents, and current or past projects of people.

Aleman-Meza et al. argue that an overview of the interests, and thus related expertise, of an individual can be created by aggregating related discussion posts of the individual across a number of websites. [34] SIOC, a project which was started in 2004, provides methods for connecting and interchanging information from several discussion methods such as blogs, forums, and mailing lists. [36]

The third used ontology is SKOS (Simple Knowledge Organization System), which can be used to describe general terms and concepts, and their properties. A major feature of SKOS is that it allows for declaring that a concept is broader / narrower than another concept. [34; 37] Aleman-Meza et al. use SKOS to define and relate skills, areas of interest (expertise), and discussed topics.

The approach taken in this research is a good practice as they reuse and combine concepts from already available vocabularies. Both the FOAF vocabulary and the SKOS vocabulary will be used for constructing the proof of concept application. This is discussed in section 4.3.1 and 4.3.4 respectively.

This chapter discussed the most important literature that has been studied during this project. The techniques from the Semantic Web that will be used in the project have been discussed. RDF will be used to store the employee data, RDFS and OWL to create the skills ontology, and SWRL rules to extent the expressivity of OWL. The Pellet reasoner will be used for interpretation of the SWRL rules. The SPARQL query language can be used to retrieve the information. Related research taught us that vocabularies used to represent data should be reused where possible to create an interoperable solution. From the vocabularies discussed in literature the FOAF vocabulary will be reused in the project for expressing personal information of the employees and the SKOS vocabulary will be reused to relate concepts in the skills ontology.

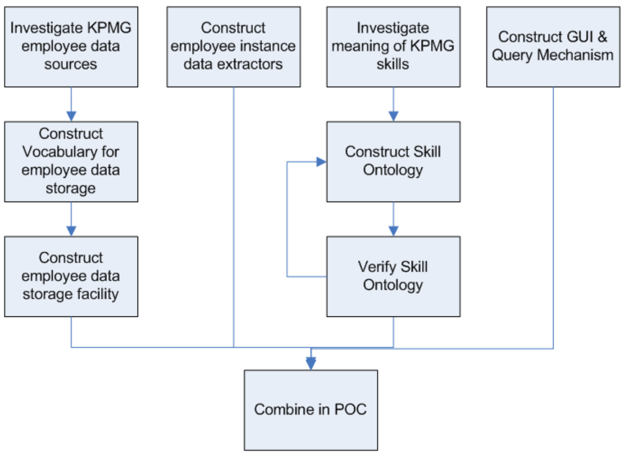
Chapter 3 – Design

Chapter one introduced the problem situation were the problems with the expertise finding applications of KPMG were explained. A direction towards a possible solution was given and research questions were formulated accordingly. Chapter two gave an overview of relevant literature around the subjects addressed in this project. This chapter starts with discussing the approach taken to build the proof of concept application, followed by an overview of the architecture of the application. The most important parts of the application will be discussed in more detail. The last section of this chapter explains how the discussed application will contribute to the project goal of improving the accessibility and the findability of KPMG employee information.

3.1 – Approach

The activities needed to build the proof of concept application are shown in figure 3.1. There are four main activities which have to be completed before their deliverables can be combined in the application.

- The employee instance data, which is currently stored in the KPMG databases and CV documents, should be expressed with RDF using suitable vocabularies. Appropriate classes and properties should thus be selected from existing vocabularies and whenever no suitable vocabularies can be found, a vocabulary with suitable classes and properties should be created. A facility has to be created which can store the employee instance data using the selected and created vocabularies.
- The process of extracting the employee instance data from the KPMG data sources is a separate activity. This comprises the construction of wrappers which are able to extract instance data from a database and CV documents.
- The construction of the skills ontology starts with defining the meaning and relations of the KPMG skills. When these are clear, this knowledge should be stored in the skills ontology and this ontology should be verified by KPMG employees. KPMG employees are the ones which posses the skills and they should thus be used to verify the correctness ontology.
- The last activity is the construction of a graphical user interface which allows employees to search for colleagues with certain expertise. Construction of the search functionality implies the construction of query functionality (SPARQL).


Investigate KPMG employee data sources
Construct employee instance data extractors
Investigate meaning of KPMG skills
Construct GUI & Query Mechanism
Construct Vocabulary for employee data storage
Construct Skill Ontology
Construct employee data storage facility
Verify Skill Ontology
Combine in POC
Figure 3.1 - Approach

All these building blocks are combined in the PoC application. The next section discusses these functional building blocks that are needed to create the PoC.

3.2 – Architecture

The image below shows the architecture of the PoC application wherein KPMG data is enriched with background knowledge to enable better findability and accessibility of this data.

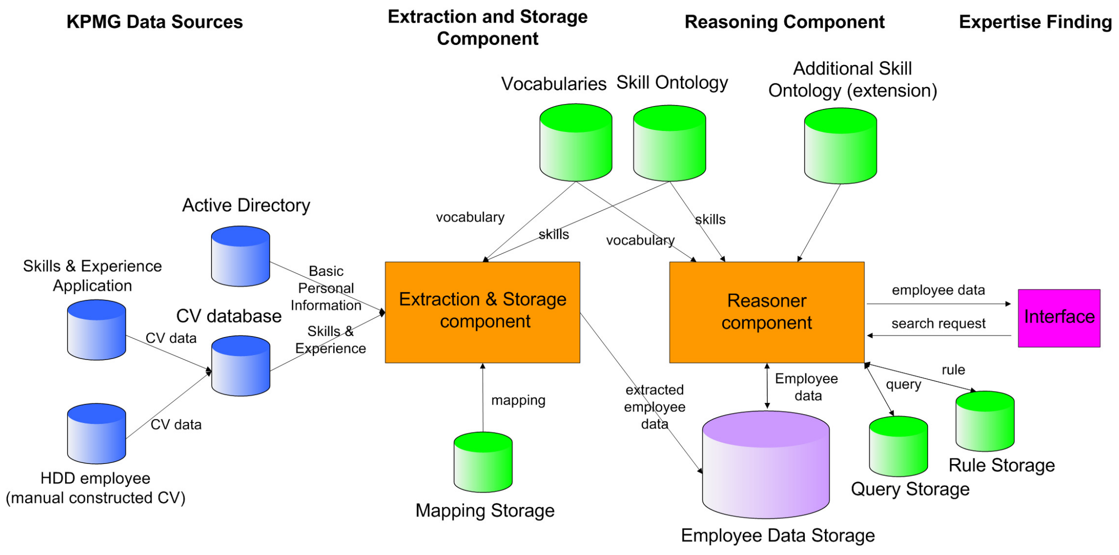


Figure 3.2 - Architecture

On the left side the KPMG employee data sources are shown (in blue). The employee instance data should be extracted and expressed using the vocabularies. The mapping repository contains the mappings that define which element of the source data should be expressed as what RDF vocabulary element. The skills that employees possess (which are stored in their CV's which are contained in the expertise finding application 'Skills & Experience' or stored on their hard drives) are matched against skills in the skills ontology and then expressed as corresponding skill instances. The employee data, expressed in RDF, is saved in the employee data store (persistent) shown in purple. The reasoner component (shown in orange) illustrates the coupling component where the employee instance data, the skills ontology, and the rules are linked together. The reasoner enriches the employee instance data by using the skills ontology and the rules which results in new fact discovery about the employee instance data. The enriched data is used in the process of expertise finding which is shown on the right side of the figure. The browser (pink) is used by an employee to find colleagues with certain expertise. The search parameters entered by the employee are converted into a query which is constructed from template queries that are stored in the query store. The reasoner component performs the query on the enriched employee instance data.
When an additional background knowledge source is available it can be linked to the application by informing the reasoner component of the existence of the knowledge source. In figure 3.2 this additional knowledge source is indicated as 'additional knowledge source (extension)'. This additional ontology can be used to enrich the employee instance data and results (could result) in even better findability and accessibility of the employee data.

3.2.1. – Employee instance data sources
Within the organization of KPMG are several different data sources available that store information about employees. For this project basic personal information and expertise information are needed. The basic personal information can be collected from the Active Directory and the expertise information can be collected from CV files.

Active Directory is Microsoft's implementation of the LDAP protocol and KPMG uses it, among other things, for storing the basic personal information of all its employees. This basic personal information consists of properties like full name, employee number, email address, telephone number, department, office location, etc.

The CV database contains CV documents of employees. These CV files are either manually constructed or automatically constructed by the Skills & Experience application (This application offers the possibility to export all the supplied expertise information to a CV file). All the CV files are in Microsoft Word or PDF format. The CV documents contain expertise information like skills, experience, projects, and education.

3.2.2 – Extraction from the data sources
The previous section discussed that the employee instance data that will be used in the application will be retrieved from the Active Directory and from employee CV documents. Extracting information from the Active Directory will not be a problem because the data is stored structured. On the other hand, extracting information from unstructured CV documents is a difficult task and a separate research project could be devoted on this subject. Therefore this project will only extract skill information from CV documents which are stored in a certain format. Existing applications will be used to support the data extraction process.

3.2.3 – Vocabularies
Concepts from vocabularies are used to express the employee instance data in RDF. As discussed in section 2.3.2 it is common practice to reuse existing vocabularies whenever possible so there should be investigated which already available vocabularies can be reused to express the employee information. Whenever concepts cannot be expressed with existing vocabularies they should be created and added to a newly created vocabulary.

3.2.4 – Skills ontology
CV documents of employees contain expertise information. For this project the skills of the employees are extracted from the CV documents. These skills will be expressed in RDF as properties of an employee. As explained in chapter one the goal of this project, improving the accessibility and the findability of KPMG employee information, can be achieved by adding background knowledge to the system. This background knowledge will be the skills ontology.

The KPMG skills ontology contains the skills, and relations between them, which are available inside the organization of KPMG. This ontology will be constructed manually and KPMG employees will be used to verify its correctness. Skills which are currently used in the My Site application (available from the alphabetically ordered dropdown menus, see section 1.2) will be used in the ontology. A separation is made between business skills (like Project Management) and technical skills (like SQL queries).

A skill in the ontology will contain a skill name, which is a copy of the name that KPMG uses on My Site, and a skill definition, which is a URL to an external (not on the KPMG intranet) website with information wherefrom employees can derive the meaning of the skill. Skills will be placed in a hierarchy based on their meaning and various relations will be defined between the individual skills to indicate how skills are related. It should for instance be possible to indicate that skills are closely related or related to each other.

3.2.5 – The GUI and SPARQL query's

The graphical user interface (GUI) can be used by employees who are in need of a colleague with certain expertise. They can search for certain employee properties like department, function, and skills. When the search button is pressed a SPARQL query will be created to retrieve the employees that match the search command.

3.3 – The application and the project goals

The previous section explained the most important building blocks of the proof of concept application but there was not explained how this application will increase the accessibility and findability of the employee information. This section explains how Semantic Web techniques like RDF and ontologies are used to achieve this goal.

The employee instance data is extracted from the KPMG data sources and expressed with the chosen vocabularies in RDF. One of the employee properties that will be gathered is the skill set of the employee. The skills ontology is the background knowledge that will be used to enrich the KPMG employee data. This is illustrated in the figure below, which is divided into three parts, namely the skills ontology (pink), the vocabulary (yellow), and the employee data storage (gray).

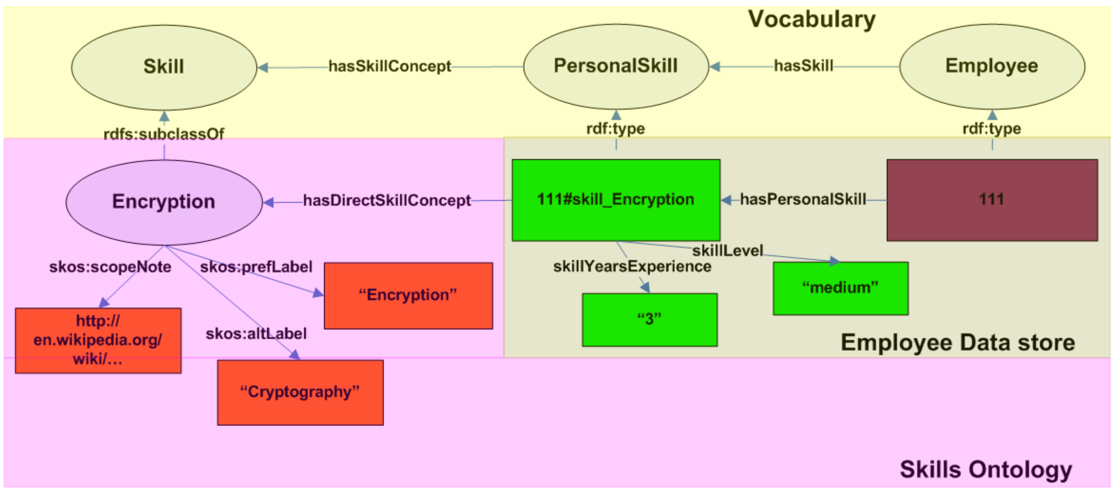


Figure 3.3 – Employee Data & Skills Ontology

In the figure circles represent classes and rectangles represent class instances. While chapter four discusses all classes and relations in more detail the goal of this section (and this figure) is to illustrate how this approach can help achieve the project goals. Therefore the shown figure is simplified and an in-depth discussion about the exact content of the figure will have to wait until chapter four where the implementation is discussed in more detail.

The part of the vocabulary shown in figure 3.3 defines that an employee has a personal skill which relates to a skill. The reason for the separation between personal skills and skills is that an employee cannot possess a skill concept. The ontology stores all skill concepts and additional information about each skill concept. An employee possesses a personal skill which can have various custom properties which vary per employee, and this personal skill relates to a skill concept in the skill ontology. This skill concept defines what the personal skill of the employee is about. This is illustrated in figure 3.3 by an employee instance '111' which has a personal skill instance '111#skill_Encryption'. The personal skill instance has two properties which define the experience that the employee has with this skill, namely the skill level and the years experience with this skill. The personal skill has a 'hasDirectSkillConcept' relation to the skill in the ontology to illustrate that this personal skill object is about the skill Encryption. The ontology stores all known facts about the Encryption skill (skill concept). The name (skos:prefLabel) contains the exact name of the skill as used within KPMG (this is stored separately because the KPMG skill names can contain special characters). Synonyms (skos:altLabel) and a definition (skos:scopeNote) of the skill are also stored. The definition is a URI to an external (non-RDF) website containing information which explains the meaning of the skill.

The interesting part is that the information contained in the employee data store is 'just' the information from the KPMG employee data sources expressed in RDF with the chosen vocabularies. The knowledge about the skills used in the KPMG employee data sources is stored in the skills ontology and can be used to enrich the employee data. While the original data in the KPMG data sources only states that a certain employee has a certain skill, the representation used in the approach described above allows for the derivation of more facts. The amount of extra facts that can be derived depends on the amount of background knowledge that is stored in the ontology. This is illustrated by an example which uses the figure below. Here, a larger part of the skills ontology is shown.

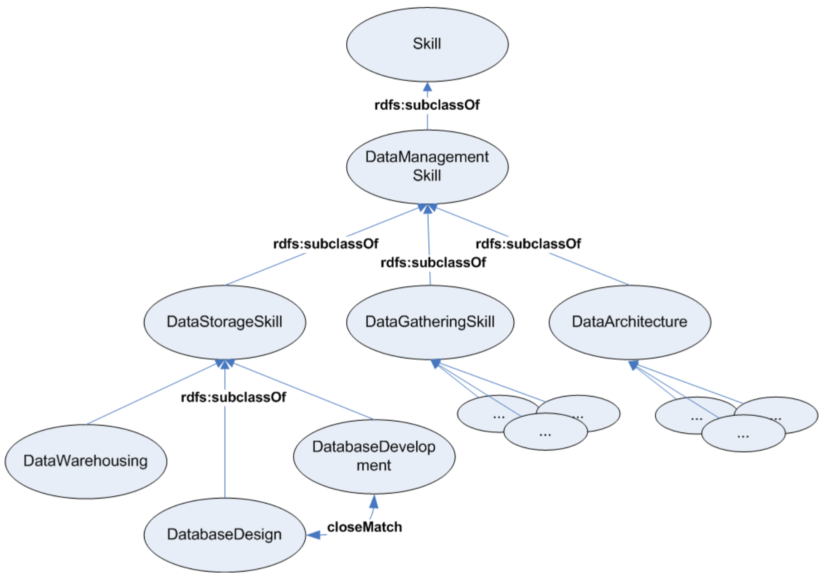


Figure 3.4 - Skills Ontology

Imagine an employee who is in need of a colleague with expertise related to database design. In the old system the employee would perform a search for all employees with the skill 'database design', and the result would be all the employees that told the system they possess this skill.
The KPMG skill list contains a skill with a closely related meaning to database design, namely the skill 'database development'. It could very well be that an employee with the skill database development also knows a lot about database design. It would therefore be very useful if the employee who is performing the search would be notified of the fact that there are also employees which possess a skill with a closely related meaning to database design. Within the skills ontology the closeMatch relation is used to express that two skills have a closely related meaning. When this knowledge would be used in the search the system would know that the employees with the skill 'database development' will probably also have expertise related to 'database design'. With this knowledge the system is thus able to provide the employee who is performing the search with all employees that have the skill database design, but also with all employees who will probably have expertise related to database design because they possess the skill database development.

This example explains how the skills ontology enables the application to improve the accessibility and findability of the employee expertise information. The goal is to create a skills ontology with many relations between the skills in order to enable a high 'intelligence'. Another relation that will be used to relate skills in the ontology is the 'related' relation which is less strong than the closeMatch relation. Synonyms of skills can also be stored to improve the findability of the skills.

Chapter 4 – Implementation

Chapter three discussed the different components needed to build the proof of concept application and how this application will help achieve the project goals. This chapter discusses the implementation process of the different components of the application.

4.1 - Frameworks and Tools

Before starting an in-depth discussion about how the components of the application are constructed, this section gives an overview of the tools and frameworks that are used in the process.

4.1.1 - Java

The Java programming language will be used to construct the application. The Java programming language is object oriented and commonly used. It is portable, which means that the application written in this language will run on all supported hardware and software platforms. There are many useful libraries available which can be used in this project for tasks like data extraction (from CV's and Active Directory), working with RDF and OWL graphs, and database access. [38]

4.1.2 - Jena

Jena is a Semantic Web framework (open source) for Java. It is an API which enables data modifications on RDF and OWL graphs via Java. It allows for performing SPARQL queries on the graphs and it has various internal reasoners. [39] Jena is used to store the java objects (an employee with its properties) as RDF, using the chosen vocabularies, in a MySQL database. Jena maintains its own format for storing the RDF data in the databases, so it is not possible to access the database directly. However, this will not be necessary because Jena can be used to retrieve all the stored data. Furthermore, it will always be possible to let Jena write all the data stored in the database to an RDF (XML) file (which can then be used by other applications).

4.1.3 – Protégé

Protégé [40] is an ontology editor (open source) which can be used to construct knowledge bases and ontologies. Because of the intuitive graphical user interface Protégé enables quick ontology building. Protégé 4.02, which has been used to construct the skills ontology, comes with the factplusplus reasoner [41] which can be used to validate the ontology.

4.1.4 – Apache UIMA

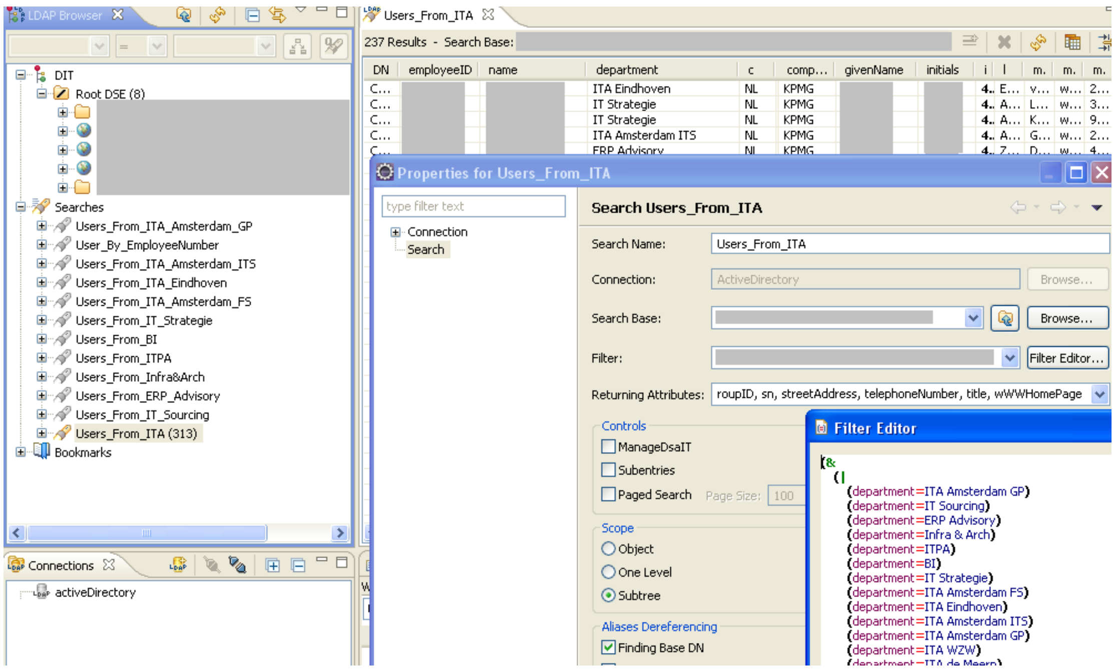
Apache UIMA (Unstructured Information Management Applications) [42] is an open-source software architecture that can be used for unstructured information analysis with the goal of discovering relevant information. It has a Java API which will be used in the proof of concept application to enable information extraction from employee CV documents. Because the application will only extract limited information from the CV documents (skill information), UIMA looks a bit too powerful for this purpose. Still the choice has been made to use this architecture because it can perfectly be used to extract much more relevant information from the CV files. As one of the project goals is to create an extendable and flexible solution, it seems wise to start using this architecture.

4.2 - Extraction from the data sources

This section describes what employee information will be extracted from the KPMG data sources and how the extraction will be performed.

4.2.1 – The Active Directory

Information extraction from the active directory can be accomplished in several different ways but because data extraction is not the focus of this research it seemed wise to go with the fastest and most easy approach. This seemed to be the open source package Apache Directory Studio. [43] This is a complete platform which runs inside Eclipse [44] (the software developing environment used in this research) which is able to connect to any LDAP-based directory server. Apache Directory Studio was able to connect to the Active Directory Server of KPMG and it turned out that it was fairly easy to retrieve data from the server and to export it to a comma separated value file (.csv).



237 Results - Search Base: [redacted]

DN	employeeID	name	department	c	comp...	givenName	initials	i	l	m.	m.	m.
C...	[redacted]	[redacted]	ITA Eindhoven	NL	KPMG	[redacted]	[redacted]	4..	E...	v...	w...	2...
C...	[redacted]	[redacted]	IT Strategie	NL	KPMG	[redacted]	[redacted]	4..	A...	L...	w...	3...
C...	[redacted]	[redacted]	IT Strategie	NL	KPMG	[redacted]	[redacted]	4..	A...	K...	w...	9...
C...	[redacted]	[redacted]	ITA Amsterdam ITS	NL	KPMG	[redacted]	[redacted]	4..	A...	G...	w...	2...
C...	[redacted]	[redacted]	ERP Advisory	NL	KPMG	[redacted]	[redacted]	4..	Z...	D...	w...	4...

Search Users_From_ITA

Search Name: Users_From_ITA

Connection: ActiveDirectory

Search Base: [redacted]

Filter: [redacted]

Returning Attributes: roupID, sn, streetAddress, telephoneNumber, title, wWWHomePage

Controls: [ ] ManageDsaIT [ ] Subentries [ ] Paged Search Page Size: 100

Scope: ( ) Object ( ) One Level (● Subtree)

Aliases Dereferencing: [✓ Finding Base DN]

```
(&
  (|
    (department=ITA Amsterdam GP)
    (department=IT Sourcing)
    (department=ERP Advisory)
    (department=Infra & Arch)
    (department=ITPA)
    (department=BI)
    (department=IT Strategie)
    (department=ITA Amsterdam FS)
    (department=ITA Eindhoven)
    (department=ITA Amsterdam ITS)
    (department=ITA Amsterdam GP)
    (department=ITA WZW)
    (department=ITA de Meern)
```

Figure 4.1 - Apache Directory Studio

The csv files containing the employee data from the active directory are stored locally and a java class has been constructed which is able to read out these csv files. For each employee in the csv files, a Java employee object is constructed (which in turn can be expressed as RDF employee instances). This process is described in section 4.4. For the parsing of the csv files the open-source package opencsv is used. [45]

4.2.2 – CV Documents

Apache UIMA (see section 4.1.4) is used to extract the skill information from the employee CV documents. While the UIMA framework is able to perform complex data extraction it turned out it was relatively easy to construct a working CV skill extraction module.

The CV documents on the ITA fileserver are all in the Adobe PDF (.pdf) or Microsoft Word (.doc) format. Before UIMA can start with concept extraction from the CV documents they have to be transformed from their current format into a text file (.txt). The java application that has been created to accomplish this uses the Apache Tika library [46].

The skills in the CV documents have to be linked to the employee instances which have been extracted from the active directory. The CV documents do not contain unique identifiers (like the employee number) so the skills from the CV document are coupled to the owner of the CV document (extracted from the Active Directory) based on the employee's name that appears in the CV. It is clear that this approach should not be used in a large organization like KPMG, but because this proof of concept application contains CV documents and employees from the IT Advisory department only, this approach is sufficient. When the application would be used on a larger scale, a simple solution to this problem would be to let employees link their CV manually.

Concepts that UIMA should extract from the CV documents are thus the skills and the name of the employee. The UIMA annotator is used to detect occurrences of these data concepts and whenever they are found they are stored, together with their position in the document, in a CAS object. When the analysis of the CV document is completed the CAS objects are returned (and can then be used by the java application).
Detecting occurrences of the data concepts (name and skills) is currently done with certain assumptions about the format of the CV files. Most of the CV files that are stored on the ITA fileserver are constructed with the Skills & Experience application. While the content of these documents is varying, most of them contain a name and a skills section which are indicated with terms like 'Name' or 'Full Name' and 'Skills' or 'Skills Acquired'. The annotator scans the CV document for these occurrences, and when found, it uses a regular expression to extract the content directly after it. The regular expression for extracting the name of the employee looks as follows:

```
( ( [\\p{Punct}]{0,2} ) ( [\\p{Blank}]{0,2} ) ( [a-zA-Z0-9]+ ) ( [\\p{Punct}]{0,2} )
( [\\p{Blank}]{0,2} ) )+
```

This means that the line containing the name of the employee may start with 0, 1, or 2 punctuation characters followed by 0, 1, or 2 blank characters, followed by 1 or more characters from the alphabet or numerical values, in turn followed by 0, 1, or 2 punctuation characters followed by 0, 1, or 2 blank characters, and this sequence should be repeated at least 1 time.
With this regular expression the annotator was able to extract each employee name instance from the CV documents. The regular expression which has been used to extract the skill instances assumes that each skill is placed on a new line. The annotator was able to extract all skills from the CV documents conform this layout.

The regular expression could be adapted to detect more complex cases. This is needed because sometimes employees place several skills on a single line or they add a description or their level of experience to the skill.

4.3 – The Vocabularies

The extracted employee data should be expressed in RDF using vocabularies. It is good practice to reuse as much concepts from existing vocabularies as possible.
So, first of all there should be investigated which already available vocabularies can be reused to express the employee information. In the early phase of the design stage the author researched several vocabularies like FOAF, DOAC, and ResumeRDF.

DOAC (Description Of A Career) is a vocabulary that can be used to describe the expertise and experience of persons. It reuses concepts from the FOAF vocabulary and allows for defining a person's experience and education. The vocabulary contains very limited concepts and therefore the choice has been made to search for an alternative. [47]

ResumeRDF is a vocabulary that can be used to describe information which is contained inside the CV (Resume) of a person. It contains much more concepts than the DOAC vocabulary and ResumeRDF seemed the right candidate for expressing the properties of the KPMG employees. Unfortunately this vocabulary could not be used because it links a CV to a person, and a CV to a skill. For the proof of concept application the data should be expressed as a person who has a certain skill. Due to these restrictions the concepts from ResumeRDF vocabulary could not be reused. [48]

4.3.1 - FOAF

FOAF (Friend Of A Friend) is a vocabulary that can be used to describe persons, their activities, and their relations to other persons. FOAF is one of the most famous vocabularies (the most famous for describing people) and is used in many examples about the Semantic Web (see section 2.3.2). FOAF will be used to describe the basic properties of the employees.

Because the FOAF vocabulary did not contain suitable concept to describe certain properties of an employee in an organization, and because no other suitable vocabularies could be found to describe these properties, a custom vocabulary containing these concepts was created (see section 4.3.3). This Employee vocabulary contains the Employee class which is a subclass of the FOAF Person class. The resulting Employee can reuse the properties of the FOAF Person class and on top of that all custom defined properties of Employee from the Employee vocabulary.

The classes used from the FOAF vocabulary are:

- Person: An Employee is a subclass of a FOAF Person.
- Document: The homepage of an employee is a FOAF Document
- Organization: The different establishments (physical buildings) of KPMG are FOAF Organizations.

The properties used from the FOAF vocabulary are:

- firstName: The first name of an Employee as a Literal.
- lastName: The last name of an Employee as a Literal
- homepage: The homepage of an Employee as a FOAF Document.
- mbox: The email address of an Employee as a RDF resource.
- phone: The phone number of an Employee as a RDF resource.
- title: The function (junior, manager, etc.) of an Employee as Literal.
- nick: The name employees use to log in to the KPMG network (distinguished name) as a Literal.

The usage of the last two properties, FOAF title and FOAF nick, may be not entirely justifiable as they can both be used to express other concepts. The FOAF title could be used to express 'Mr.' or 'Mrs.' and the FOAF nick could be used to express the nickname of a person. Another FOAF property that is reused and needs some explanation is the FOAF businessCard property. This property has the 'testing' status which means that it is not in the official FOAF vocabulary yet, however, [49] describes its usage. The businessCard property is used to describe the properties of the establishments of KPMG, which is described in more detail in the next section.

An employee expressed in RDF with concept from the FOAF vocabulary is shown below. The concepts from the Employee vocabulary are shown in section 4.3.3.

```
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:j.0="http://xmlns.com/foaf/0.1/"
xmlns:j.1="http://www.somewhere.nl/employeeVocabulary/01/ev.owl#"
xmlns:j.2="http://www.w3.org/2006/vcard/ns#">
<j.1:Employee rdf:about="http://www.somewhere.nl/kpmgInstanceStorage/01/
       108099">
    <Employee Vocabulary...>
    <j.0:firstName>Alex</j.0:firstName>
    <j.0:homepage>
       <j.0:Document
          rdf:about="https://personal.ema.kworld.kpmg.com/ajansen/"/>
    </j.0:homepage>
    <j.1:isSituatedAt>
      <j.0:Organization
         rdf:about="http://www.somewhere.nl/kpmgInstanceStorage01/KPMG-
             Netherlands_AmstelveenLangerhuize">
         <j.0:businessCard>
            <j.2:VCard>
                 <VCARD...>
            </j.2:VCard>
         </j.0:businessCard>
      </j.0:Organization>
    </j.1:isSituatedAt>
    <j.0:lastName>Jansen</j.0:lastName>
    <j.0:mbox rdf:resource="mailto:Jansen.Alexander@kpmg.nl"/>
    <j.0:nick>ajansen</j.0:nick>
    <j.0:phone rdf:resource="tel:+31206 562934"/>
    <j.0:title>Stagiair Universitair</j.0:title>
  </j.1:Employee>
</rdf:RDF>
```

4.3.2 – vCard-RDF

vCard-RDF is a vocabulary which can be used to express business card information. [50] In this project vCard-RDF is used to store information about the establishments where the KPMG employees are situated. This incorporates the name and address of the establishment. This could be extended with properties like a phone number, email-address, or the latitude and longitude coordinates. Below, the RDF of the Amstelveen Langerhuize establishment is shown. Note that the business card of the organization is linked to the FOAF organization class instance of KPMG-Netherlands-AmstelveenLangerhuize by the FOAF property businessCard.

The classes used from the vCard-RDF vocabulary are:
- VCard: The VCard resource contains all the classes and properties used to describe the information about the establishment.
- Address: A VCard resource can have the property adr which connects the VCard to the Address resource. The Address resource contains address information.
- Organization: A VCard resource can have the property org which connects the VCard to the Organization resource. The Organization resource contains information about the organization.
- Email: A VCard resource can have the property email which connects the VCard to the Email resource, containing the email address.
- Tel: A VCard resource can have the property tel which connects the VCard to the Tel resource, containing the telephone number.

The properties used from the vCard-RDF vocabulary are:
- countryName: is a property of the Address class and contains the name of the country as a Literal.
- locality: is a property of the Address class and contains the name of the city as a Literal.
- postalCode: is a property of the Address class and contains the postal code as a Literal.
- region: is a property of the Address class and contains the state (whenever used in addresses) as a Literal.
- streetAddress: is a property of the Address class and contains the street address as a Literal.
- organizationName: is a property of the Organization class and contains the name of the organization as a Literal.
- organizationUnit: is a property of the Organization class and contains the name of the establishment being described as a Literal.
- adr: is a property of the VCard class and links the VCard to the Address resource.
- org: is a property of the VCard class and links the VCard to the Organization.
- fn: is a property of the VCard class and contains the full name as a Literal of the object of the VCard.
- email: is a property of the VCard class and links the VCard to the Email resource.
- tel: is a property of the VCard class and links the VCard to the Tel resource.

```
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:j.0="http://xmlns.com/foaf/0.1/"
xmlns:j.1="http://www.somewhere.nl/employeeVocabulary/01/ev.owl#"
xmlns:j.2="http://www.w3.org/2006/vcard/ns#">
<j.0:Organization
    rdf:about="http://www.somewhere.nl/kpmgInstanceStorage/01/KPMG-
    Netherlands_AmstelveenLangerhuize">
    <j.0:businessCard>
       <j.2:VCard>
          <j.2:adr>
             <j.2:Address>
                <j.2:country-name>NETHERLANDS</j.2:country-name>
                <j.2:locality>Amstelveen</j.2:locality>
                <j.2:postalCode>1186DS</j.2:postalCode>
                <j.2:street-address>Laan van Langerhuize 1</j.2:street-
                    address>
             </j.2:Address>
          </j.2:adr>
          <j.2:fn>Amstelveen Langerhuize</j.2:fn>
          <j.2:org>
             <j.2:Organization>
                <j.2:organization-name>KPMG NETHERLANDS</j.2:organization-
                   name>
                <j.2:organization-unit>Amstelveen Langerhuize
                   </j.2:organization-unit>
             </j.2:Organization>
          </j.2:org>
       </j.2:VCard>
    </j.0:businessCard>
</j.0:Organization>
</rdf:RDF>
```

4.3.3 – Employee vocabulary

The Employee vocabulary contains all the concepts that are needed to express properties of the KPMG employees which could not be found in existing vocabularies (or were not suitable for reuse due to restrictions).

The classes of the Employee Vocabulary:
- Employee: An Employee is a subclass of FOAF Person.
- Department: An employee works at a certain Department (like IT Advisory). Department is a subclass of the FOAF Group class.
- Service: An employee has experience with providing certain Services. An Employee from the Business Intelligence Department will have experience with providing the Business Intelligence service. An Employee from the IT Advisory GP (General Practice) Department might have experience with several services like the IT Attestation service and/or the IT Audit service. Service is also a subclass of the FOAF Group class.
- PersonalSkill: Each employee possesses certain skills. These personal skills are expressed with the PersonalSkill class.
- Skill: All PersonalSkills of the employees have a skill concept in the skills ontology. These concept skills are of the class Skill.

Readers who are confused by the existence of both Skill and PersonalSkill are directed to section 3.3 where the relation between those two classes is explained.

An employee can have the following properties:
- company: The name of the company of the employee as a Literal.
- employee number: The number of the employee as a Literal.
- initials: The initials of the employee as a Literal.
- office: The office location of the employee (usually room and floor number) as a Literal.
- isSituatedAt: Links an Employee to a FOAF Organization (section 4.3.1).

– memberOf: Links an Employee to a FOAF Group. This property can be used to link an Employee to a Department or a Service.
– hasSkill: Links employees to the PersonalSkill they possess.

The memberOf property has been added to the Employee vocabulary because the existing property FOAF member links a FOAF Group to its members. Because the inverse relation is need, namely the relation which links an Employee to a certain group (Department or Service), the inverse relation memberOf had to be created.

A Skill, from the skills ontology, contains the following properties:
– skillName: Contains the name of the skill as used within KPMG (My Site), as a Literal.
– skillDefinition: Contains a link to an external webpage (non-RDF) where a description of this skill is given as an anyURI (xsd).

A PersonalSkill, which is possessed by an employee, can contain the following properties:
– hasSkillConcept: Links the PersonalSkill to the concept Skill in the Skills ontology.
– skillLevel: The level that the employee who possesses this specific skill has as a Literal.
– skillYearsExperience: The years of experience that the employee who possesses this specific skill has as a Literal.

Below an example is given of an employee with properties from the Employee vocabulary.

```
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:j.0="http://xmlns.com/foaf/0.1/"
xmlns:j.1="http://www.somewhere.nl/employeeVocabulary/01/ev.owl#"
xmlns:j.2="http://www.w3.org/2006/vcard/ns#">
<j.1:Employee rdf:about="http://www.somewhere.nl/kpmgInstanceStorage/01/
       108099">
   <j.1:company>KPMG</j.1:company>
   <j.1:employeeNumber>108099</j.1:employeeNumber>
   <j.1:initials>EA</j.1:initials>
   <j.1:isSituatedAt
      rdf:resource="http://www.somewhere.nl/kpmgInstanceStorage/01/KPMG-
      Netherlands_AmstelveenLangerhuize"/>
   <j.1:memberOf
      rdf:resource="http://www.somewhere.nl/kpmgSkillsOntology/01/
      skills.owl#ITAAmsterdamGPDepartment"/>
   <j.1:office>B04.01</j.1:office>
   <FOAF...>
  </j.1:Employee></rdf:RDF>
```

4.4 - Skills ontology

The background knowledge that the system uses for the enrichment of the employee data is stored in the skills ontology. The skills ontology contains the skills, and relations between them, which are available inside the organization of KPMG.
A skill in the ontology contains a skill name, which is a copy of the name that KPMG uses on My Site, and a skill definition, which is a URL to an external (not on the

KPMG intranet) website with information wherefrom employees can derive the meaning of the skill. The skills are placed in a hierarchy and various relations are defined between the individual skills in order to relate them to each other.

4.4.1 – Approach

As described in section 3.2.4 the skills ontology should contain a subset of the skills which are currently available inside KPMG's My Site application, with focus on IT Advisory. Employees which use My Site to define their expertise have to select their skills from a dropdown menu. There are two of such menus, one for business skills and one for technical skills. The first step was to extract those skills from these menus and place them in a list (one for each category). From these lists a selection was made of the skills that employees from the IT Advisory department can possess.
The first selection was made by the author who selected all skills of which he thought that they would exist in the IT Advisory department. Then the skills that appeared in the CV documents of the employees from the IT Advisory department, which were not yet contained in the selection, where added to the selection. The selected skills were placed in a hierarchy using Protégé (section 4.1.3). Then, KPMG employees were asked to browse through the constructed hierarchy and to make remarks about missing skills or wrongly placed skills. The last stage was to add relations between the skills in the hierarchy. First the author defined relations and then KPMG employees were asked to verify the constructed relations and to make remarks about missing relations.

4.4.2 - OWL Species

Because there are various ways to construct the skills ontology choices had to be made about how to construct it. The choice was mainly about what variant of OWL to use because the chosen OWL species determines what can, and cannot, be done. The following discussion incorporates the various species of OWL which were discussed in section 2.2.6.

Three variants of the skills ontology have been considered wherein each skill in the skills ontology could have certain properties like a name, a definition, and relations to other skills (and departments and/or services).

Model A: OWL Full
OWL Full is the most powerful species of OWL. Each distinct skill concept could be defined as a separate skill class with a name, synonyms, a definition, and relations to other skill classes. The skill classes could be placed in a hierarchy based on their meaning by using the 'rdfs:subclassOf' property. The downside of OWL Full is that some well-known Semantic Web applications do not support it (for example Protégé, the application wherein the author has experience) [51, 52]. Besides, many reasoners do not support OWL Full [53].

Model B: OWL DL
OWL DL is less powerful than OWL Full. With OWL DL it is not possible to give a class property values, which means that it is not possible to state that the skill class 'ITAudit' has the name 'IT Audit' and is defined at 'http://www…'. Because skill class instances can be given property values, a single skill class could be defined and

then each skill concept could be an instance of this skill class. This approach does not allow the usage of the property 'rdfs:subclassOf' to create a hierarchy between the skills because all skill concepts are class instances. Another property could be used to construct the hierarchy (like skos:broader and skos:narrower, see section 5.2, where this is approach is used for constructing another small skills ontology). However, because we want to create a hierarchy based on the meaning of the skills the usage of the rdfs:subclassOf property seems the most logical. Its usage results in a hierarchy wherein all subclasses of a skill class together comprise the complete meaning of the skill class. All subclasses of the Data Management skill class taken together cover the whole data management (skill) domain. It should be said that currently the resulting hierarchy is only used in the visualization of the skills ontology, but nevertheless it still is the correct representation of reality and could furthermore be used in future extensions of the application and/or ontology. An approach which could be used to stay in OWL DL and at the same time use the rdfs:subclassOf property is to create the hierarchy of skill classes and then create for each skill class an instance which stores the additional information of the skill. [54] The downside of this approach is that the information about each skill concept is separated over the skill class, which is in the rdfs:subclassOf hierarchy, and the skill class instance, which contains the name, definition and other relations of the skill concept.

Model C: OWL 2 DL
While OWL 2 DL does still not support classes having property values it does support something else which is pretty useful, namely punning. With punning the same URI can be used for entities of different kinds. This means that a class and a class instance can be given the same URI. [55] The approach discussed in Model B, where each skill concept consists of a class which is in the class hierarchy and a class instance which contains additional information about the skill, looks suddenly attractive. If the same URI would be used for the skill class and the skill class instance the result would look like OWL Full while staying in OWL DL. It turned out that most Semantic Web tools that support OWL DL also support punning. Therefore the choice was made to create the skill ontology using OWL 2 DL with punning.

4.4.3 – The Data Model

The previous section explained that the choice was made to use OWL 2 DL to construct the skills ontology. To make use of the rdfs:subclassOf property and at the same time storing several properties of skills there was decided to make use of punning. This, and the concepts used in the data model, are explained using the figure shown below. Note that this figure was already displayed in section 3.3 to explain the basic idea of the skills ontology.

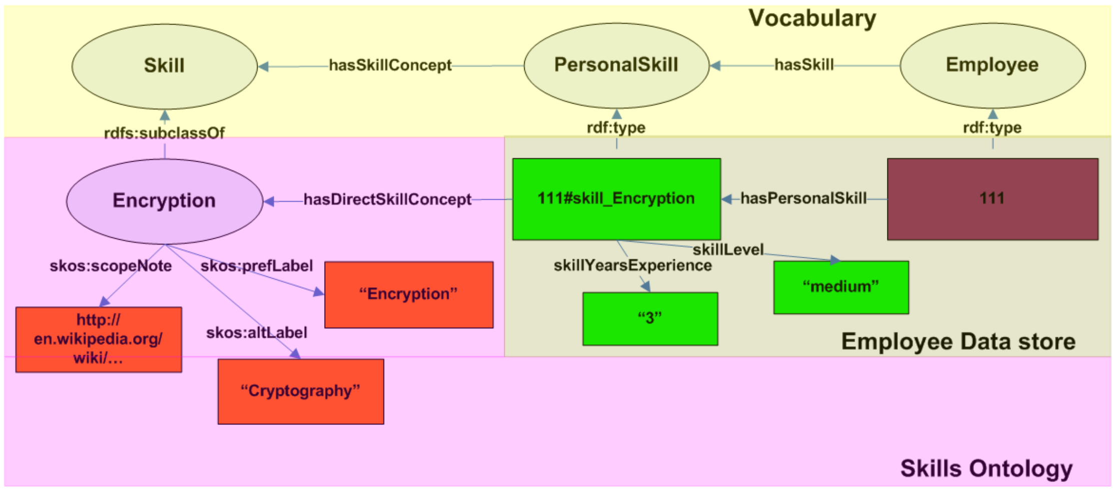


Figure 4.2 – Data Model

The Skills Ontology part of the figure (pink) shows the Encryption skill with three properties that describe the skill. The properties from the SKOS vocabulary (Simple Knowledge Organization System) [56] are used to indicate the preferred name, the alternative name(s) (synonyms), and the definition of the skill. The figure also shows that the encryption skill is in a rdfs:subclassOf hierarchy. This situation can only be realized in OWL Full, but as stated before, the skills ontology will be in OWL 2 DL. The figure gives a simplified view of the situation and the real situation is displayed below.

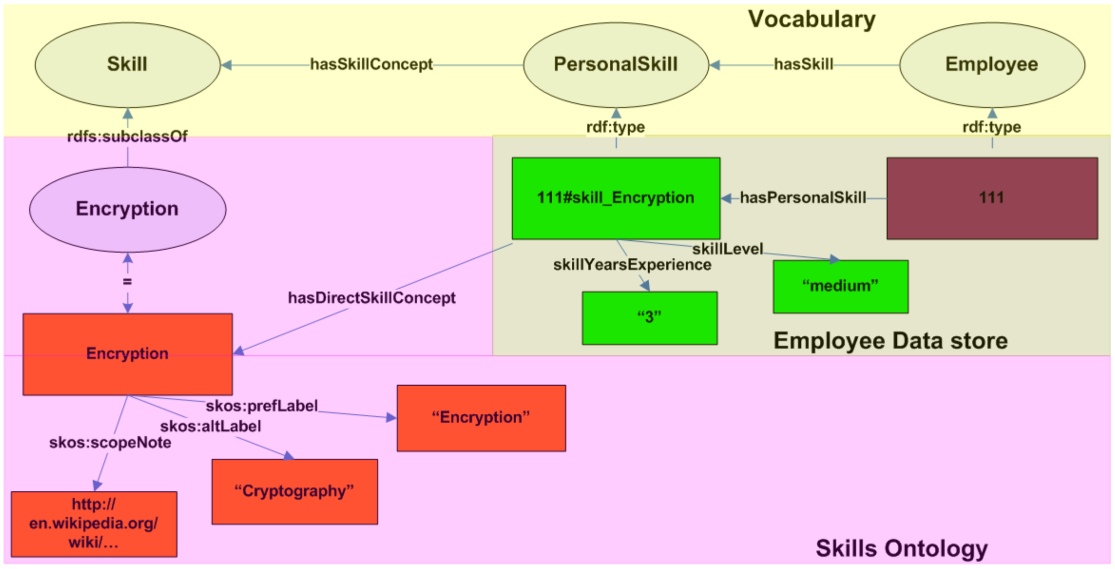


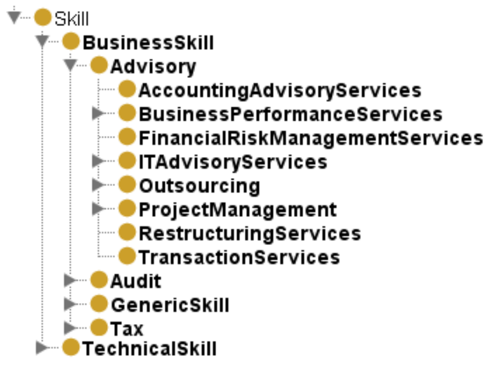
Figure 4.3 - Data Model OWL 2 DL - punning

This figure shows that the skill concept in the skills ontology consists of both the skill class and the skill class instance which both have the same URI (punning). The figures which will be displayed in the rest of this document will be displayed like the 'simplified' version.

Now that the choice for the OWL variant and the data model have been explained we can continue the discussion of the skills ontology, starting with the skills hierarchy.

4.4.4 – Skill hierarchy

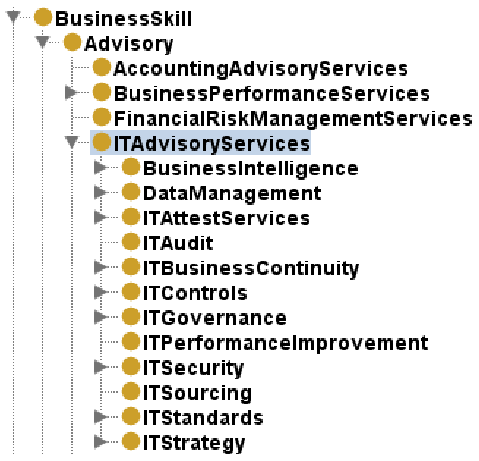
The skills in the ontology are placed in a hierarchy based on their meaning. The highest level of this hierarchy divides the skills into the categories technical skills and business skills. The business skills can be divided into skills which belong to one of the three (high level) services that KPMG is providing, namely Audit, Tax, and Advisory. Skills that do not belong in one of these three categories are placed inside the GenericSkill class. Just like the GenericSkill class some other skill classes have been added which can not be found on My Site. They were needed in order to construct the ontology as consistent and logical as possible.



```
Skill
  BusinessSkill
    Advisory
      AccountingAdvisoryServices
      BusinessPerformanceServices
      FinancialRiskManagementServices
      ITAdvisoryServices
      Outsourcing
      ProjectManagement
      RestructuringServices
      TransactionServices
    Audit
    GenericSkill
    Tax
  TechnicalSkill
```

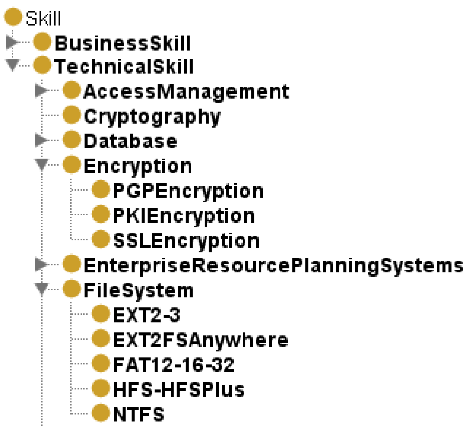
Figure 4.4 – Business Skills

The advisory skills have been ordered according to the services wherein they are used. The skills beneath the IT Advisory Service class are shown in figure 4.3. Part of the hierarchy constructed from the technical skills is shown in figure 4.4.



```
BusinessSkill
  Advisory
    AccountingAdvisoryServices
    BusinessPerformanceServices
    FinancialRiskManagementServices
    ITAdvisoryServices
      BusinessIntelligence
      DataManagement
      ITAttestServices
      ITAudit
      ITBusinessContinuity
      ITControls
      ITGovernance
      ITPerformanceImprovement
      ITSecurity
      ITSourcing
      ITStandards
      ITStrategy
```

Figure 4.5 – IT Advisory

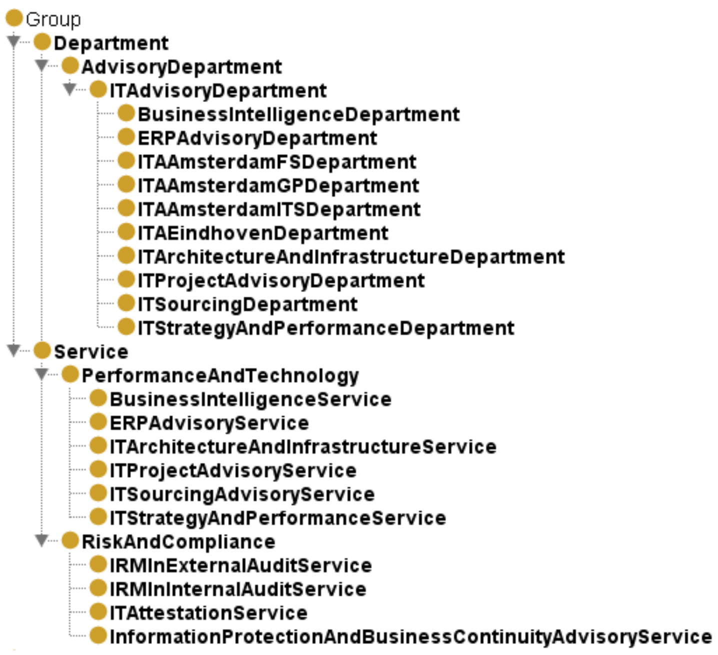


```
Skill
  BusinessSkill
  TechnicalSkill
    AccessManagement
    Cryptography
    Database
    Encryption
      PGPEncryption
      PKIEncryption
      SSLEncryption
    EnterpriseResourcePlanningSystems
    FileSystem
      EXT2-3
      EXT2FSAnywhere
      FAT12-16-32
      HFS-HFSPlus
      NTFS
```

Figure 4.6 – Technical Skills

Departments are included in the skills ontology and relations between departments and skills can be defined. This is of great added value because certain skills will certainly be available in a specific department which means that an employee from that department will most probably possess those skills. Employees from more specialized departments like the Business Intelligence department mainly perform a specialized service, the business intelligence service. Employees from the more general departments like the ITA Amsterdam GP department perform various services. To incorporate this difference, the specialized departments will be linked to a larger skill set than their corresponding service. The resulting behavior will be that an employee from the Business Intelligence department will posses more skills related to business intelligence than an employee from the ITA GP department who sometimes performs Business Intelligence services. Within KPMG a separation is made between Performance & Technology (P&T) services and Risk & Compliance (R&C) services. Employees from the more general departments (FS, GP, ITS) commonly perform R&C services and ERP services, but also to lesser extent the other services.

Note that there is a difference between the services which are subclasses of the Skill class (figure 4.4) and the services which are subclasses of the Service (figure 4.7) class. The first services are used to group all skills together which belong to this specific service. Furthermore, these skills (with ‘services’ in their name) come from My Site and are thus currently used within CV documents of employees. The latter services are used to link employees that have experience with the service to a specific skill set. This skill set contains only a few (the most common) skills from the same service skill group.



- Group
- Department
  - AdvisoryDepartment
    - ITAdvisoryDepartment
      - BusinessIntelligenceDepartment
      - ERPAdvisoryDepartment
      - ITAAmsterdamFSDepartment
      - ITAAmsterdamGPDepartment
      - ITAAmsterdamITSDepartment
      - ITAEindhovenDepartment
      - ITArchitectureAndInfrastructureDepartment
      - ITProjectAdvisoryDepartment
      - ITSourcingDepartment
      - ITStrategyAndPerformanceDepartment
- Service
  - PerformanceAndTechnology
    - BusinessIntelligenceService
    - ERPAdvisoryService
    - ITArchitectureAndInfrastructureService
    - ITProjectAdvisoryService
    - ITSourcingAdvisoryService
    - ITStrategyAndPerformanceService
  - RiskAndCompliance
    - IRMInExternalAuditService
    - IRMInInternalAuditService
    - ITAttestationService
    - InformationProtectionAndBusinessContinuityAdvisoryService

Figure 4.7 - Departments & Services

4.4.5 – Relations

The introduction chapter explained that it should be possible to state that skills have a closely related, or related, meaning. As section 3.3 explained such relations could be used to infer expertise of employees enabling improved search performance. During implementation another relation was added to relate the skills in the ontology, namely the subSkill / superSkill property. The table below lists the three properties used to relate the skills.

Relation	**Meaning**	**Employee with skill A**
SKOS:closeMatch	defines that skill A and skill B have a closely related meaning	will probably possess skill B
SKOS:related	defines that the meaning of skill A and the meaning of skill B are in some way related	might possess skill B
subSkill / superSkill	defines that skill A is a superSkill of skill B	might possess skill B

While the SKOS closeMatch and the SKOS related property are used between skills throughout the whole ontology, the subSkill (and its inverse superSkill property) are only used between skill classes that are direct subclasses of each other. Because the variety of different skills placed in the layers of the hierarchy it is not possible to state that an employee which has a certain skill also has all the subclass skills of that skill. However, in some cases this does hold and therefore the subSkill property is used. Section 3.3 already explained that these relations are used to infer that employees have certain skills. Before explaining the details of this process there is first a short note about combining SKOS and OWL followed by an explanation about relating departments and services to skills.

There is a difference between OWL and SKOS as OWL is intended for knowledge representation and SKOS is intended for knowledge organization (to create thesauri) [57]. There are some guidelines described in [58] about how to use SKOS in an OWL ontology. The skills ontology uses SKOS as described in the 'Formal / Semi-Formal Hybrids' pattern. The Skill class is a subclass of SKOS:concept and skill instances are related to each other by using the SKOS semantic relations SKOS closeMatch and SKOS related. The skills ontology is more an ontology than a thesaurus because it also contains departments and services which can be related to skills and rules to infer expertise of employees.

The previous section discussed that employees are member of a department and that departments are linked to skills. Employees can also indicate that they have experience in certain services. There was also explained that some departments are mainly concerned with certain specific services. To capture this knowledge in the skills ontology two additional properties are needed. Departments and services are related to skills with the custom defined property containsSkill. The containsService relation can be used to indicate that a certain department is mainly concerned with a certain service.

Relation	**Meaning**
containsSkill	A Department or Service contains a skill
containsService	A Department contains a Service

The next section explains how these relations are exploited to enable improved expertise finding.

4.4.6 – SWRL: rules for inference

The defined relations have to be exploited in a certain way to enable the improved expertise finding. The first approach taken was to incorporate the existence of these relations in SPARQL queries. This resulted in complex (large) queries. A query that can be used to search for an employee with the Database Design Skill, or a skill with a closely related meaning, is shown below.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX so: <http://www.somewhere.nl/kpmgSkillsOntology/01/skills.owl#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?employee ?skillA ?skillB WHERE {
{ ?employee so:hasSkill ?skillA  .
?skillA so:hasSkillConcept so:DatabaseDesign }
UNION
{ ?employee so:hasSkill ?skillA  .
?skillA so:hasSkillConcept ?skillB .
so:DatabaseDesign skos:closeMatch ?skillB} }
```

This query only incorporates the relation closeMatch while there can be more relations. After experimenting a little with these queries another approach was adopted. The expressive power of OWL was extended by adding SWRL Rules (section 2.2.7). SWRL rules can be used to exploit the relations discussed in the previous section.

4.4.6.1 – Relate Skills

The first relation discussed was the SKOS closeMatch property which is used to indicate that two skills have a closely related meaning. This is illustrated in figure 4.8. Employee instance '111' has got a personal skill with a hasDirectSkillConcept relation to the DatabaseDesign skill in the skills ontology. The skills ontology defines that the DatabaseDesign skill has a closeMatch relation with the DatabaseDevelopment skill.

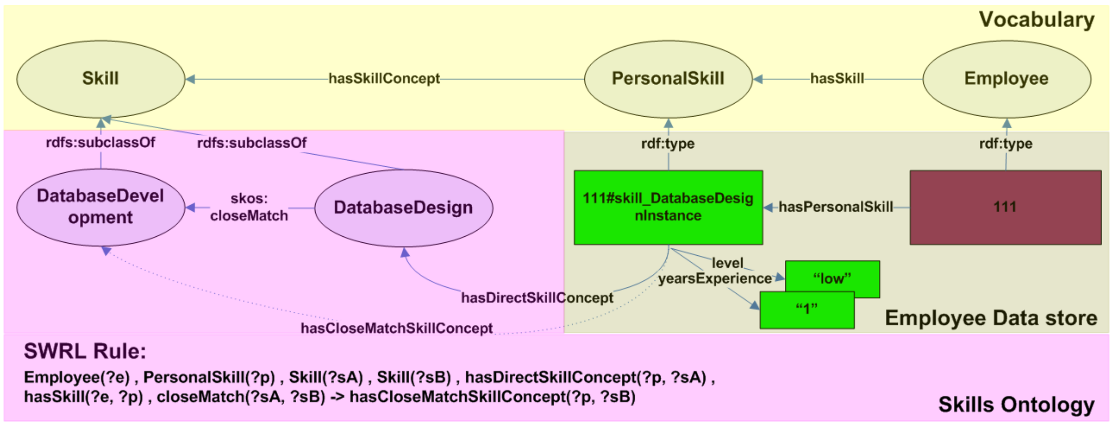
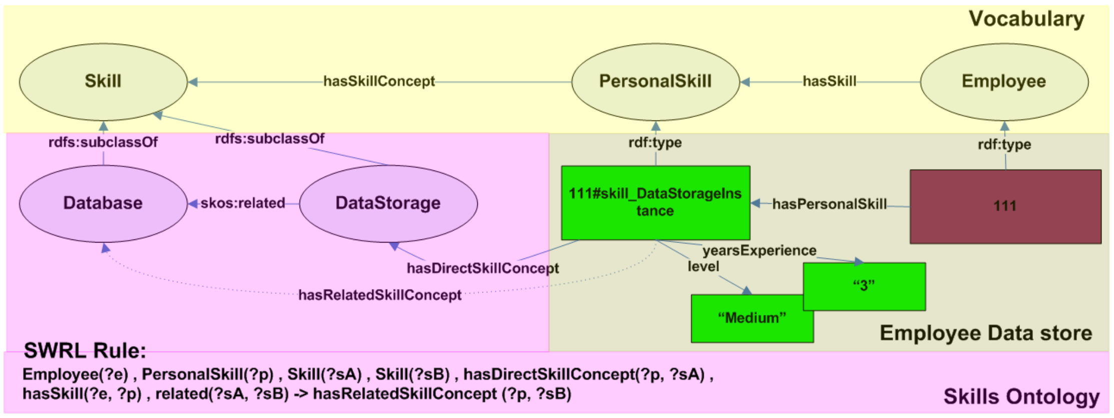


Figure 4.8 - hasCloseMatchSkillConcept

The main question now is how can we exploit the closeMatch relation between the two skills in the ontology in such a way that we can use it to easily find employee '111' when a search is performed for employees with the 'Database Development' skill? It cannot be the case that employee '111' would possess the 'Database Development' skill, because it is not the same skill but a skill with a closely related meaning. There was chosen to create an additional relation from the personal skill of the employee to the skill concept in the ontology which has a closeMatch relation with the skill concept which represents the personal skill of the employee. This relation is called 'hasCloseMatchSkillConcept' and is, just like the 'hasDirectSkillConcept' property, a sub property of the 'hasSkillConcept' property. This additional relation allows for easy (small) queries and fast data retrieval when a search is performed. The following SWRL rule is used to infer this relation (SWRL rules are easy to read and write and pretty much self descriptive).

Employee(?e) , PersonalSkill(?p) , Skill(?sA) , Skill(?sB) , hasDirectSkillConcept(?p, ?sA) , hasSkill(?e, ?p) , closeMatch(?sA, ?sB) -> hasCloseMatchSkillConcept(?p, ?sB)

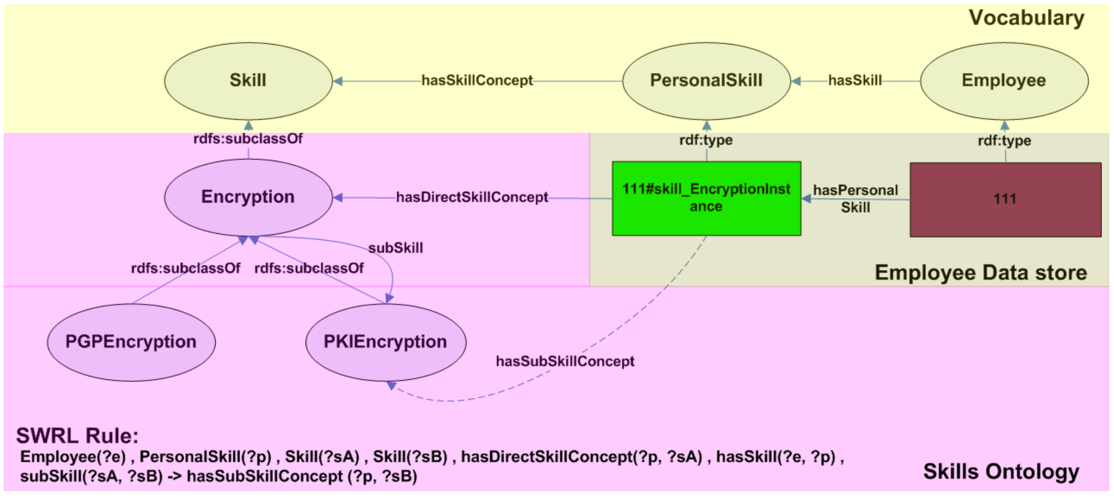
The approach used with the SKOS related property is equal to the approach used with the SKOS closeMatch property discussed above. The only difference is that the SKOS related property is used between skills to indicate that their meaning is in some way related. This is thus a weaker relationship than the SKOS closeMatch property. The inferred property is called 'hasRelatedSkillConcept'.



SWRL Rule:
Employee(?e) , PersonalSkill(?p) , Skill(?sA) , Skill(?sB) , hasDirectSkillConcept(?p, ?sA) , hasSkill(?e, ?p) , related(?sA, ?sB) -> hasRelatedSkillConcept (?p, ?sB)

Figure 4.9 - hasRelatedSkillConcept

The previous section introduced the custom defined hierarchical relation property called subSkill. This relation is only allowed between direct subclasses and the figure below illustrates its usage.



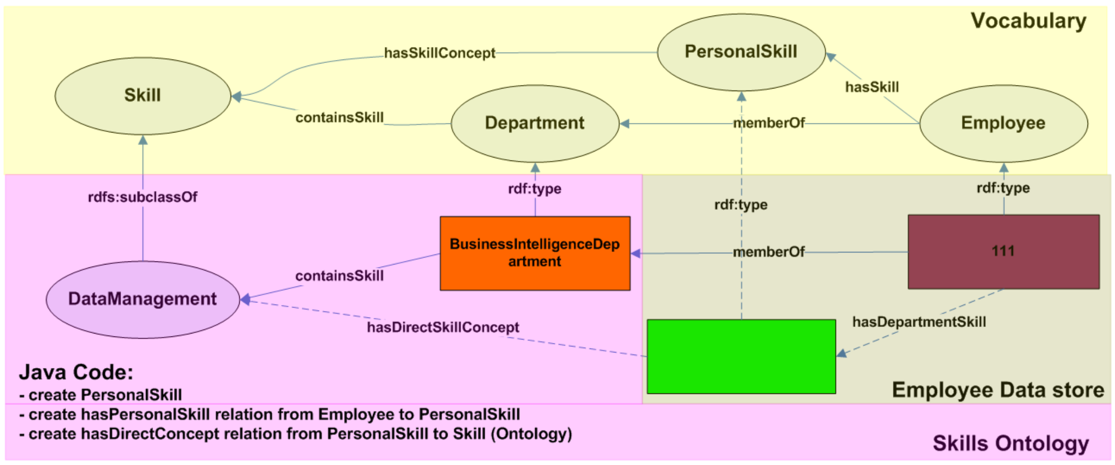
SWRL Rule:
Employee(?e) , PersonalSkill(?p) , Skill(?sA) , Skill(?sB) , hasDirectSkillConcept(?p, ?sA) , hasSkill(?e, ?p) , subSkill(?sA, ?sB) -> hasSubSkillConcept (?p, ?sB)

Figure 4.10 - hasSubSkillConcept

The employee has a personal skill with the Encryption skill concept. In this figure, the skills ontology defines two subclasses of Encryption, namely PGPEncryption and PKIEncryption. The PKIEncryption skill is defined to be a subSkill of the Encryption skill. This means that employees who have the Encryption skill could also have the PKIEncryption skill. The figure also displays the SWRL rule used to generate this inference and the resulting relation between the personal skill and the skill in the ontology is called 'hasSubSkillConcept'.
It should be obvious that the rdfs:subclassOf property cannot be used to infer this relation because then this inference would be made for each skill having a subclass.

4.4.6.2 – Relate Departments, Services, and Skills

To indicate that employees from a certain department will have expertise related to the activities of that department, departments are linked to skills with the containsSkill relation. This is shown in the figure below.

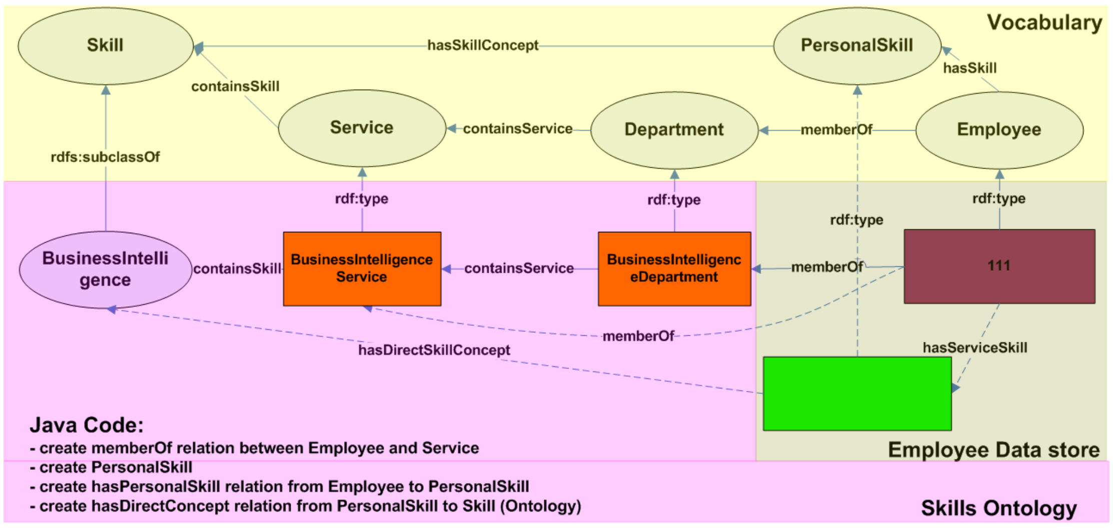


Java Code:
- create PersonalSkill
- create hasPersonalSkill relation from Employee to PersonalSkill
- create hasDirectConcept relation from PersonalSkill to Skill (Ontology)

Figure 4.11 - hasDepartmentSkill

Employee '111' is memberOf the Business Intelligence Department. The skills ontology defines that this department contains the Data Management skill. This means that employee '111' should be related to the Data Management skill in the ontology. The problem now is that there is no personal skill instance to use to indicate the relation between the employee and the ontology skill. SWRL does not allow for instance creation [bron] (only relations can be created). Therefore a piece of Java code was written which adds the personal skill instance, and the corresponding relations, to the model. Note that the relation between the employee and the personal skill object is called 'hasDepartmentSkill', which is a sub property of the 'hasSkill' relation.

To incorporate the fact that employees from the Business Intelligence department have more skills related to business intelligence than employees who sometimes perform the business intelligence service (discusses in section 4.4.4) the Business Intelligence department is linked to the Business Intelligence Service with the containsService relation as illustrated below.



Java Code:
- create memberOf relation between Employee and Service
- create PersonalSkill
- create hasPersonalSkill relation from Employee to PersonalSkill
- create hasDirectConcept relation from PersonalSkill to Skill (Ontology)

Figure 4.12 - hasServiceSkill

This works equally as the situation described above. The difference is that a memberOf relation is created between the employee who is memberOf the department and the service which is contained by the department (department - containsService -

service). The relation between the employee and the personal skill is now called 'hasServiceSkill'.

The SWRL rules discussed in section 4.4.6.1 do also apply for the personal skills which are created because an employee is member of a department (or service) which contains skills.

4.5 – Java Code

The Application is build in Java with use of the Jena framework. This section describes the most important Java classes and their functionality.

- VocaManager: The Vocabulary Manager reads out the various vocabularies that are needed to express all concepts in RDF. It is able to read out vocabularies stored on the Internet but the default setting is to read the vocabularies from files stored on the hard disk. The VocaManager maintains a representation of the skills ontology as a Jena Model.
- DBConnector: The Database Connector creates a connection with the database that stores the employee instance data. All instance data is stored in a Jena Model. The Database Connector provides all kind of functionality to retrieve instances from the Model. This class also contains the Inference Model which contains the enriched data. The Inference Model is constructed from combining the skills ontology, the instance data, and the reasoner (Pellet).
- QueryStorage: The DBConnector class contains the Query Storage which stores various queries used to retrieve instance data from the Jena Models.
- EmployeeManager / SkillManager / EstablishmentManager: These Manager classes are used as a layer between the Java objects and the RDF instances.

4.6 - The Gui

The Graphical User Interface (GUI) of the application has been developed to run in an arbitrary web browser. This is accomplished with Java Servlets and Java Server Pages [59], and Apache Tomcat [60] is used as web server to host the application. Because of time restriction on the project, both the design and the functionality of the GUI are really basic.

A distinction is made between a regular employee and an administrator. The administrator GUI (blue) enables an administrator to view and change properties of employees and the employee GUI (silver) enables employees to define their skills and search for colleagues.

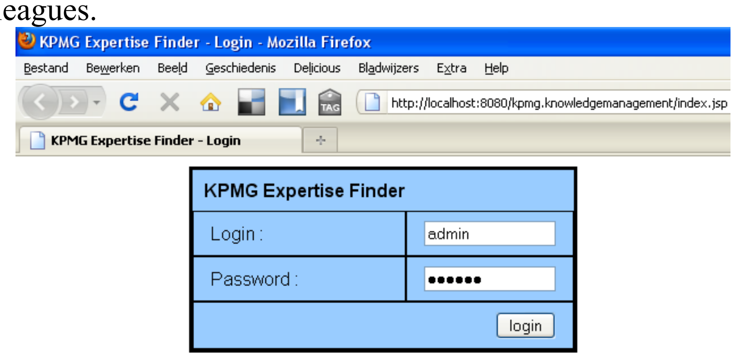
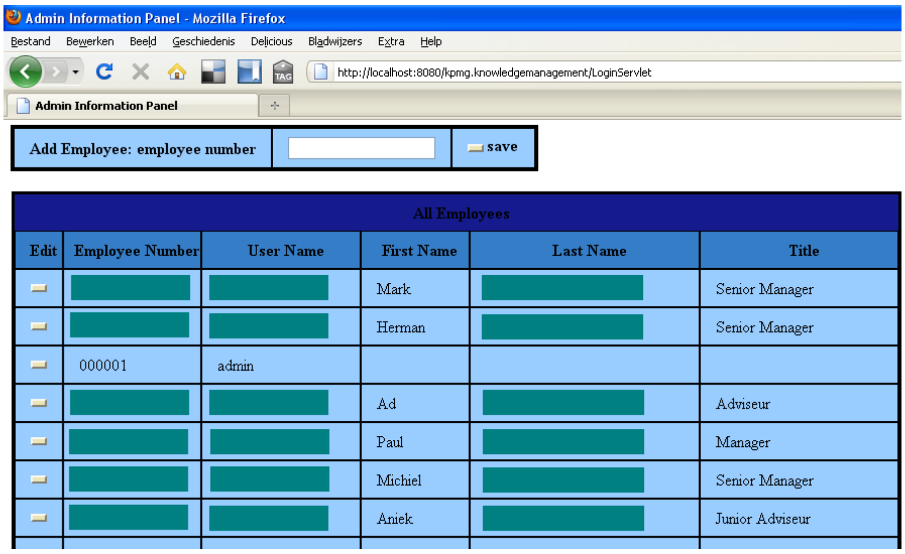

KPMG Expertise Finder - Login - Mozilla Firefox
Bestand Bewerken Beeld Geschiedenis Delicious Bladwijzers Extra Help
http://localhost:8080/kpmg.knowledgemanagement/index.jsp
KPMG Expertise Finder - Login
KPMG Expertise Finder
Login :
admin
Password :
••••••
login

Figure 4.13 – Login Screen

4.6.1 - The administrator GUI

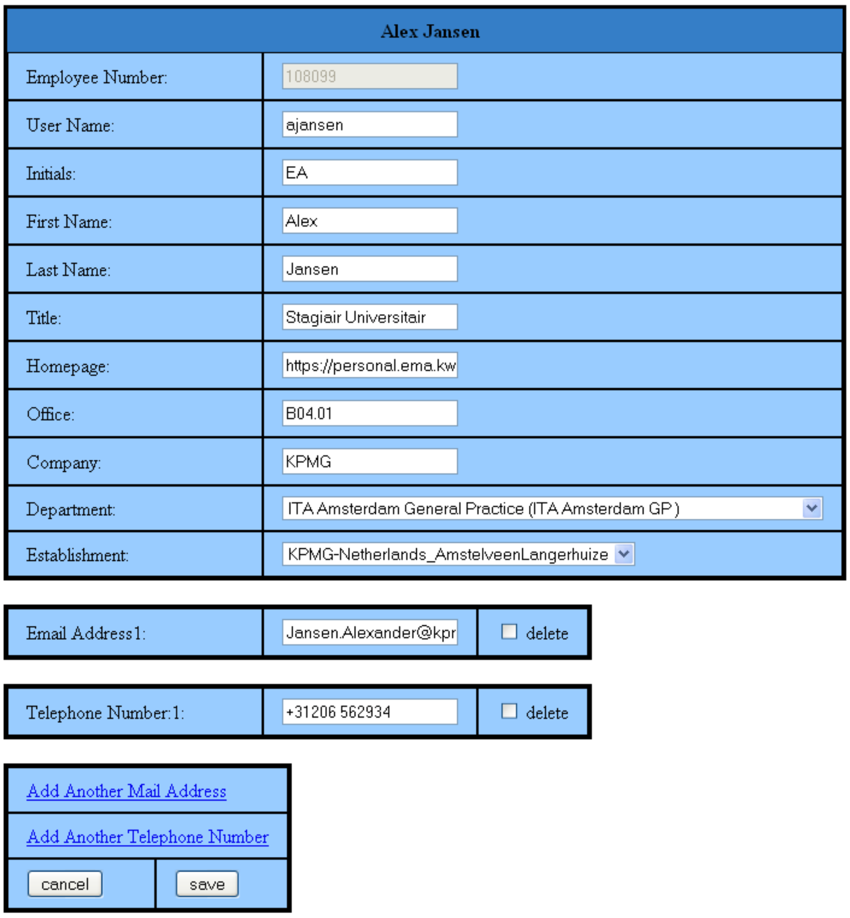
When administrators logon to the system they are presented with an overview of the employees in the system. The properties of employees can be changed and new employees can be created.


Admin Information Panel - Mozilla Firefox
Bestand Bewerken Beeld Geschiedenis Delicious Bladwijzers Extra Help
http://localhost:8080/kpmg.knowledgemanagement/LoginServlet
Admin Information Panel
save

Add Employee: employee number		save

All Employees					
Edit	Employee Number	User Name	First Name	Last Name	Title
▬	[redacted]	[redacted]	Mark	[redacted]	Senior Manager
▬	[redacted]	[redacted]	Herman	[redacted]	Senior Manager
▬	000001	admin			
▬	[redacted]	[redacted]	Ad	[redacted]	Adviseur
▬	[redacted]	[redacted]	Paul	[redacted]	Manager
▬	[redacted]	[redacted]	Michiel	[redacted]	Senior Manager
▬	[redacted]	[redacted]	Aniek	[redacted]	Junior Adviseur

Figure 4.14 - Admin GUI: Overview



Alex Jansen	
Employee Number:	108099
User Name:	ajansen
Initials:	EA
First Name:	Alex
Last Name:	Jansen
Title:	Stagiair Universitair
Homepage:	https://personal.ema.kw
Office:	B04.01
Company:	KPMG
Department:	ITA Amsterdam General Practice (ITA Amsterdam GP)
Establishment:	KPMG-Netherlands_AmstelveenLangerhuize

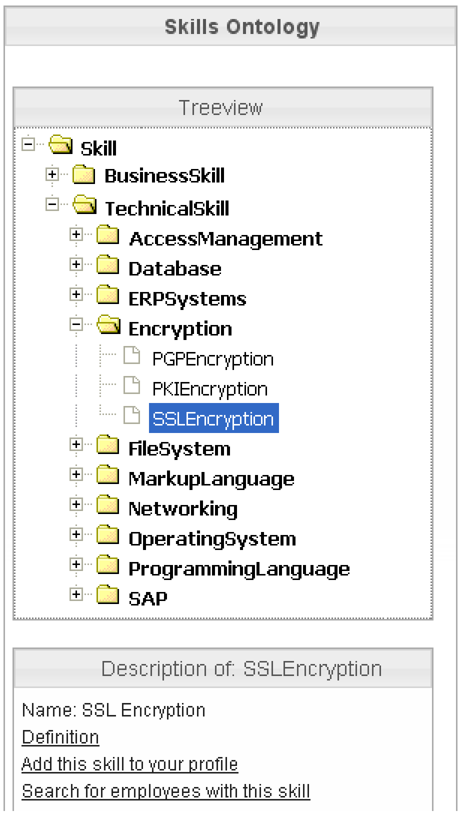
Email Address1:	Jansen.Alexander@kpr	☐ delete

Telephone Number:1:	+31206 562934	☐ delete

Figure 4.15 - Admin GUI: Edit Employee

4.6.2 - The Employee GUI

The employee GUI enables employees to review their personal information and to define their skills. A search panel can be used to find colleagues with certain skills. Image 4.12 shows an overview of the employee GUI.



Skills Ontology

Treeview

- Skill
  - BusinessSkill
  - TechnicalSkill
    - AccessManagement
    - Database
    - ERPSystems
    - Encryption
      - PGPEncryption
      - PKIEncryption
      - SSLEncryption
    - FileSystem
    - MarkupLanguage
    - Networking
    - OperatingSystem
    - ProgrammingLanguage
    - SAP

Description of: SSLEncryption

Name: SSL Encryption
Definition
Add this skill to your profile
Search for employees with this skill

Figure 4.16 – Treeview of skills ontology

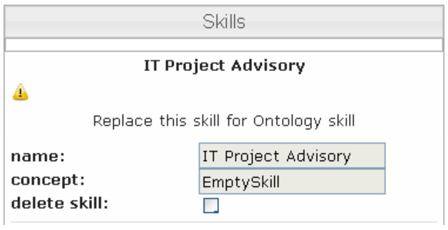
At first the skills ontology was visualized with a javascript package called jOWL. [61] It turned out that the package did not support the OWL 2 DL punning feature and this resulted in strange behavior. Too bad, because the jOWL package looks very good and works easy. Because a tree view allows to browse the ontology in a convenient way, a custom solution was build to visualize the skills ontology with the DynaTree javascript package [62] (it must be said that the way in which the tree is now created will certainly not win the beauty contest, but again, this is just a proof of concept).

The treeview visualizes all skills of the ontology and whenever a skill is selected all information about the skill is shown in a description box below the treeview as shown in image 4.10.

The skill name (as used within KPMG) and the skill definition (when available) are shown. Employees can add the selected skill to their profile or search for employees with this skill. When they add the skill to their profile they can add their skill level and the total years of experience they have with the skill. Currently, no predefined values are available and employees are free to fill these fields as they like. The service experience box enables employees to indicate their experience with the various services.

When employees indicate that they want to search for employees who have a specific skill from the skills ontology, the skill is added to the search panel in the upper part of the GUI. There is also the possibility to search for colleagues with skills based on a free text search. The search, and the search results, are discussed in more detail in chapter 5.

Skills that were extracted from CV documents and did not match any skill instance from the skills ontology are represented by the 'EmptySkill'. The employee is notified that this skill should be linked to a skill from the skills ontology.



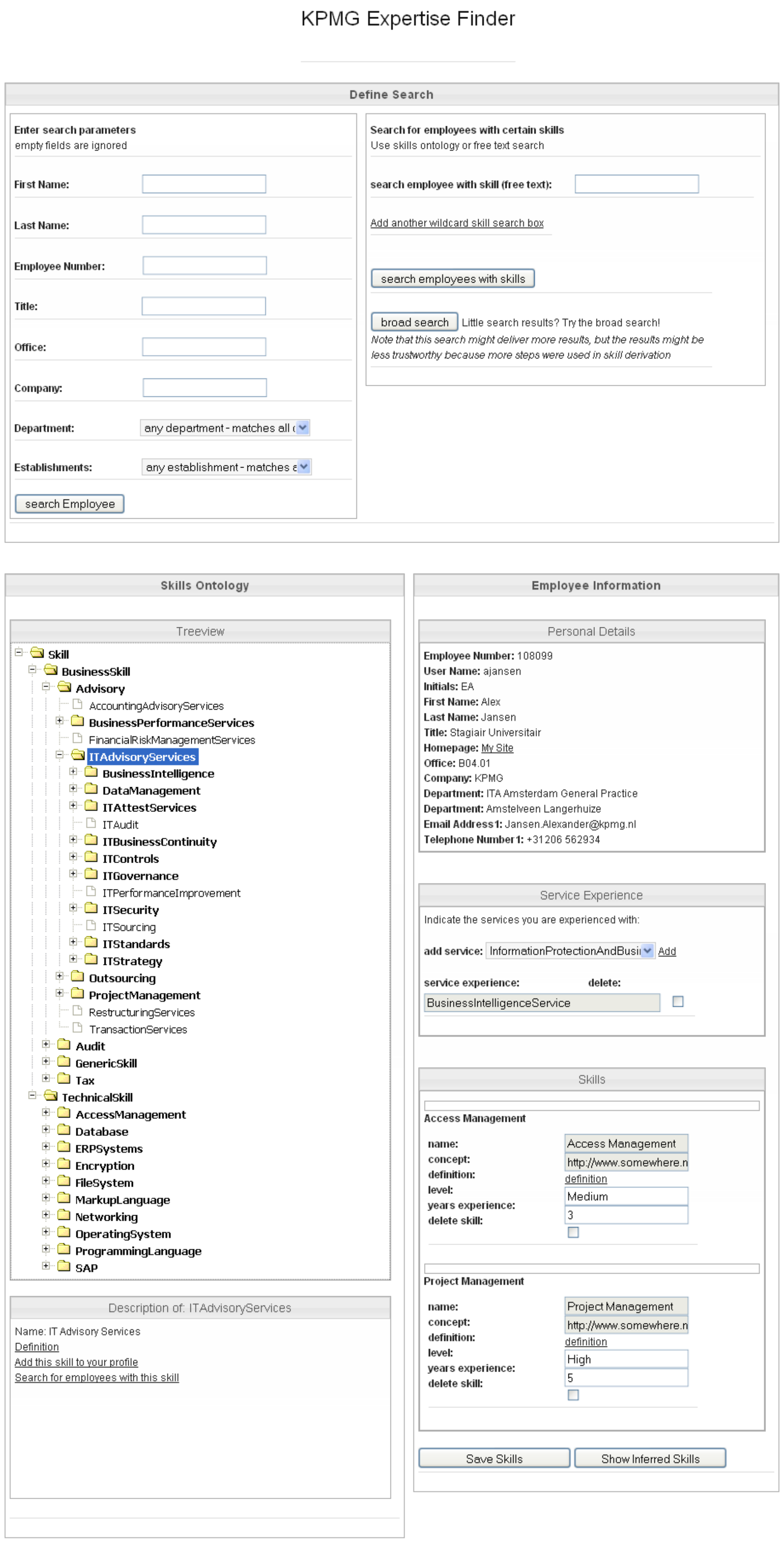
Skills

IT Project Advisory

Replace this skill for Ontology skill

name:	IT Project Advisory
concept:	EmptySkill
delete skill:	☐

Figure 4.17 - unknown Skill


KPMG Expertise Finder

Define Search

Enter search parameters
empty fields are ignored

First Name:	
Last Name:	
Employee Number:	
Title:	
Office:	
Company:	
Department:	any department - matches all (
Establishments:	any establishment - matches a

search Employee

Search for employees with certain skills
Use skills ontology or free text search

search employee with skill (free text):

Add another wildcard skill search box

search employees with skills

broad search Little search results? Try the broad search!
Note that this search might deliver more results, but the results might be less trustworthy because more steps were used in skill derivation

Skills Ontology

Treeview

- **Skill**
  - **BusinessSkill**
    - **Advisory**
      - AccountingAdvisoryServices
      - **BusinessPerformanceServices**
      - FinancialRiskManagementServices
      - **ITAdvisoryServices**
        - **BusinessIntelligence**
        - **DataManagement**
        - **ITAttestServices**
        - ITAudit
        - **ITBusinessContinuity**
        - **ITControls**
        - **ITGovernance**
        - ITPerformanceImprovement
        - **ITSecurity**
        - ITSourcing
        - **ITStandards**
        - **ITStrategy**
      - **Outsourcing**
      - **ProjectManagement**
      - RestructuringServices
      - TransactionServices
    - **Audit**
    - **GenericSkill**
    - **Tax**
  - **TechnicalSkill**
    - **AccessManagement**
    - **Database**
    - **ERPSystems**
    - **Encryption**
    - **FileSystem**
    - **MarkupLanguage**
    - **Networking**
    - **OperatingSystem**
    - **ProgrammingLanguage**
    - **SAP**

Description of: ITAdvisoryServices

Name: IT Advisory Services
Definition
Add this skill to your profile
Search for employees with this skill

Employee Information

Personal Details

Employee Number: 108099
User Name: ajansen
Initials: EA
First Name: Alex
Last Name: Jansen
Title: Stagiair Universitair
Homepage: My Site
Office: B04.01
Company: KPMG
Department: ITA Amsterdam General Practice
Department: Amstelveen Langerhuize
Email Address 1: Jansen.Alexander@kpmg.nl
Telephone Number 1: +31206 562934

Service Experience

Indicate the services you are experienced with:

add service: InformationProtectionAndBusi Add

service experience:	**delete:**
BusinessIntelligenceService	☐

Skills

Access Management

name:	Access Management
concept:	http://www.somewhere.n
definition:	definition
level:	Medium
years experience:	3
delete skill:	☐

Project Management

name:	Project Management
concept:	http://www.somewhere.n
definition:	definition
level:	High
years experience:	5
delete skill:	☐

Save Skills Show Inferred Skills

Figure 4.18 - Employee GUI: Overview

Chapter 5 – Example usage of the application

The previous chapters discussed the construction of the proof of concept application. A skills ontology has been constructed, data of KPMG employees has been expressed as RDF and various relations and rules have been defined to relate employees and skills. This chapter discusses example usage of the expertise finding application. The first section explains the process of expertise finding with the constructed expertise finding application. Example searches are discussed and results are explained. The second section discusses the extendability of the application by demonstrating the process of linking another knowledge base to the system.

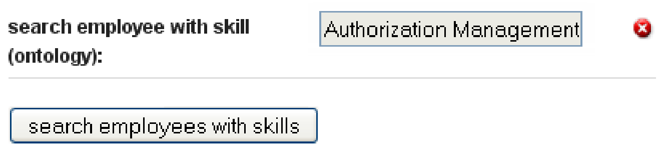
5.1 – Expertise Finding

The skills ontology with the SWRL rules, the instance data, and the Pellet reasoner are combined in the inference model. This is an enriched Ontology Model (Jena) which is used in the process of expertise finding. Section 4.4 discussed the skills ontology with the SWRL rules. There was explained that the SWRL rules are used to infer relations between personal skills of employees and skills in the skills ontology, based on relations between skills in the skills ontology. The Pellet reasoner has support for SWRL rules and it can therefore be used to infer the additional relations. The personal skills of employees which are derived based on the membership of an employee of a certain department or service (section 4.4.6.2) are also created and added to the model. Whenever employees search for colleagues with certain skills, i.e. expertise finding, a query is constructed and run on the inference model. A query for a search for all employees with the skill ‘Authorization Management’ is displayed below.

```
SELECT ?employee WHERE
{
   ?employee rdf:type ev:Employee .
   {
      ?employee ev:hasSkill ?ps .
      ?ps ev:hasSkillConcept ?s .
      {
         {
            ?s skos:prefLabel ?name
         }
         UNION
         {
            ?s skos:altLabel ?name
         }
         FILTER (str(?name) = "Authorization Management" )
      }
   }
}
```

Because the super property hasSkill is used in the query all sub properties of hasSkill are also incorporated. The result is that this query also retrieves the employees who have a skill based on the fact that they are member of a certain department or service. The super property hasSkillConcept incorporates all possible relations between a personal skill and an ontology skill. The showed query wherein these both super properties are used is thus able to find all employees that (could) have expertise related to ‘Authorization Management’. Then, for each employee in the search results, there can be inspected why the employee has this expertise.

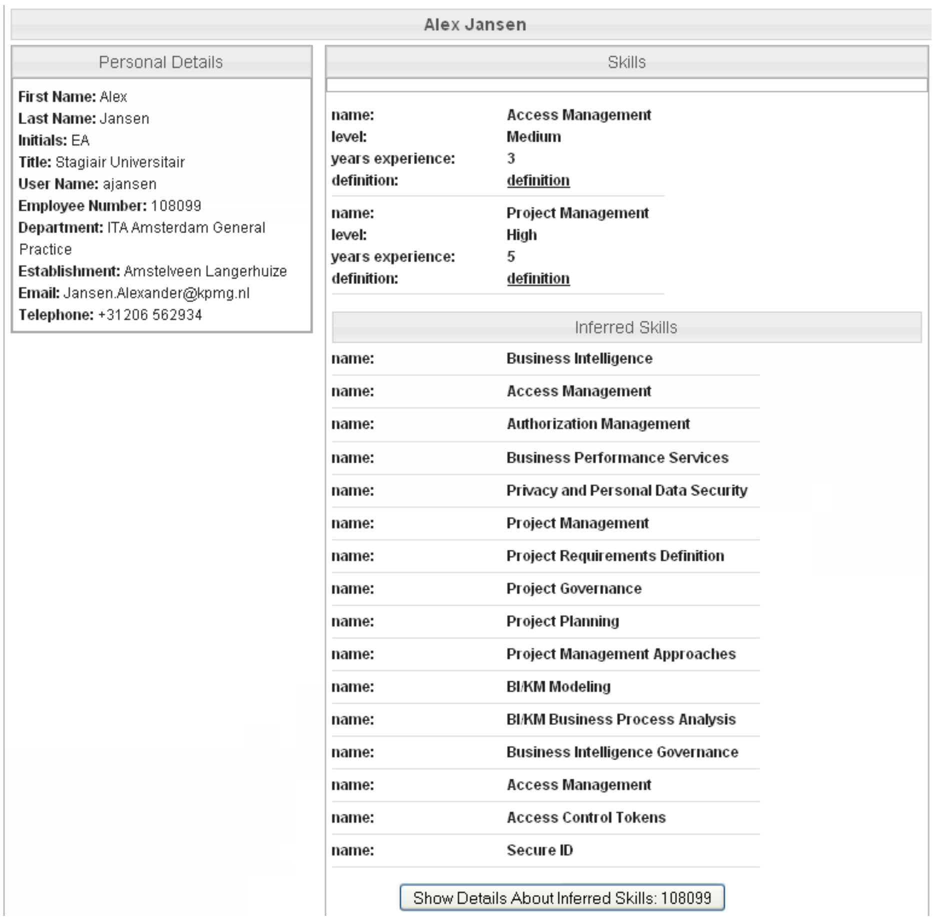
It is important to present the relation between the employee and the skill in the search results as there is a difference between an employee who explicitly stated that he has a certain skill and an employee of which the system thinks (inferred) that he possesses the skill. This search example is illustrated below.



search employee with skill (ontology): Authorization Management

Figure 5.1 - Search for employee with skill

Imagine an employee who is looking for a colleague with experience in Authorization Management. The employee could use the treeview to traverse the skills ontology and then add the skill to the search section. After pressing the search button the query shown above will be run against the inference model. The search results show all employees that have the skill Authorization Management and the employee who initiated the search can easily see what the relation between each employee and the searched skill is. For example, employee ‘Alex Jansen’ who appeared in the search results has the Authorization Management skill as an inferred skill. This means that Alex Jansen did not tell the system explicitly that he possesses this skill. The employee performing the search can inspect where the relation between Alex Jansen and the Authorization Management skill came from by clicking the show details button.



Alex Jansen

Personal Details

First Name: Alex
Last Name: Jansen
Initials: EA
Title: Stagiair Universitair
User Name: ajansen
Employee Number: 108099
Department: ITA Amsterdam General Practice
Establishment: Amstelveen Langerhuize
Email: Jansen.Alexander@kpmg.nl
Telephone: +31206 562934

Skills

name:	Access Management
level:	Medium
years experience:	3
definition:	definition
name:	Project Management
level:	High
years experience:	5
definition:	definition

Inferred Skills

name:	Business Intelligence
name:	Access Management
name:	Authorization Management
name:	Business Performance Services
name:	Privacy and Personal Data Security
name:	Project Management
name:	Project Requirements Definition
name:	Project Governance
name:	Project Planning
name:	Project Management Approaches
name:	BI/KM Modeling
name:	BI/KM Business Process Analysis
name:	Business Intelligence Governance
name:	Access Management
name:	Access Control Tokens
name:	Secure ID

Figure 5.2 - Search results (only showing one employee)

This reveals (see figure 5.5) that the Authorization Management skill was inferred from the fact that Alex Jansen possesses the Access Management skill, and that

Authorization Management has a closely related meaning with Access Management. Furthermore, Alex Jansen indicated that he has a medium expertise level and 3 year experience with the Access Management skill. Note that without the knowledge captured inside the skills ontology it would not have been possible to relate Alex Jansen to the skill Authorization Management. The employee performing the search could look at the other search results to see if there are employees who explicitly stated that they posses the Authorization Management skill, or if there are employees who have more experience with the skill.

If a search results in little results (little employees that have the skill) the employee performing the search can try the 'broad search'. The regular search explained above returns all employees that have a personal skill with a relation to a skill concept that is being searched. This is illustrated in the figure below.

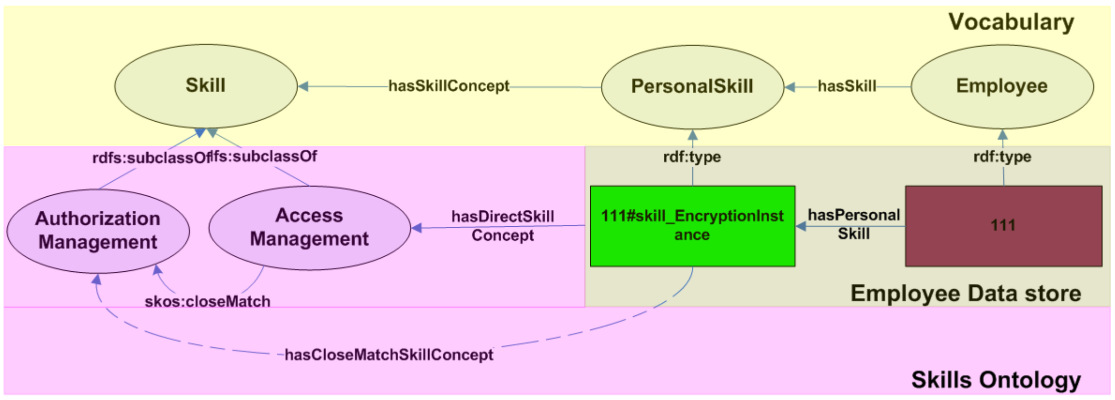


Figure 5.3 - Search

The broad search incorporates the existence of one additional relation between the skill that is being inferred and the skill that is searched for. This is illustrated in the figure below.

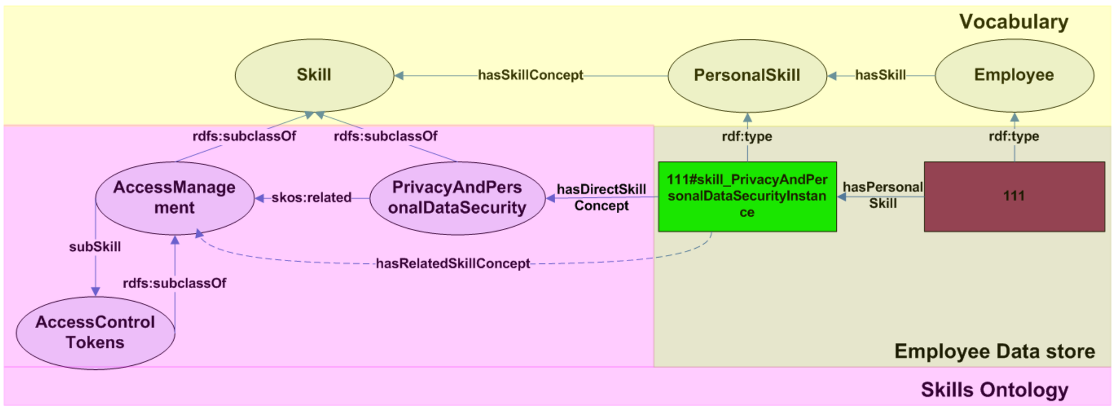
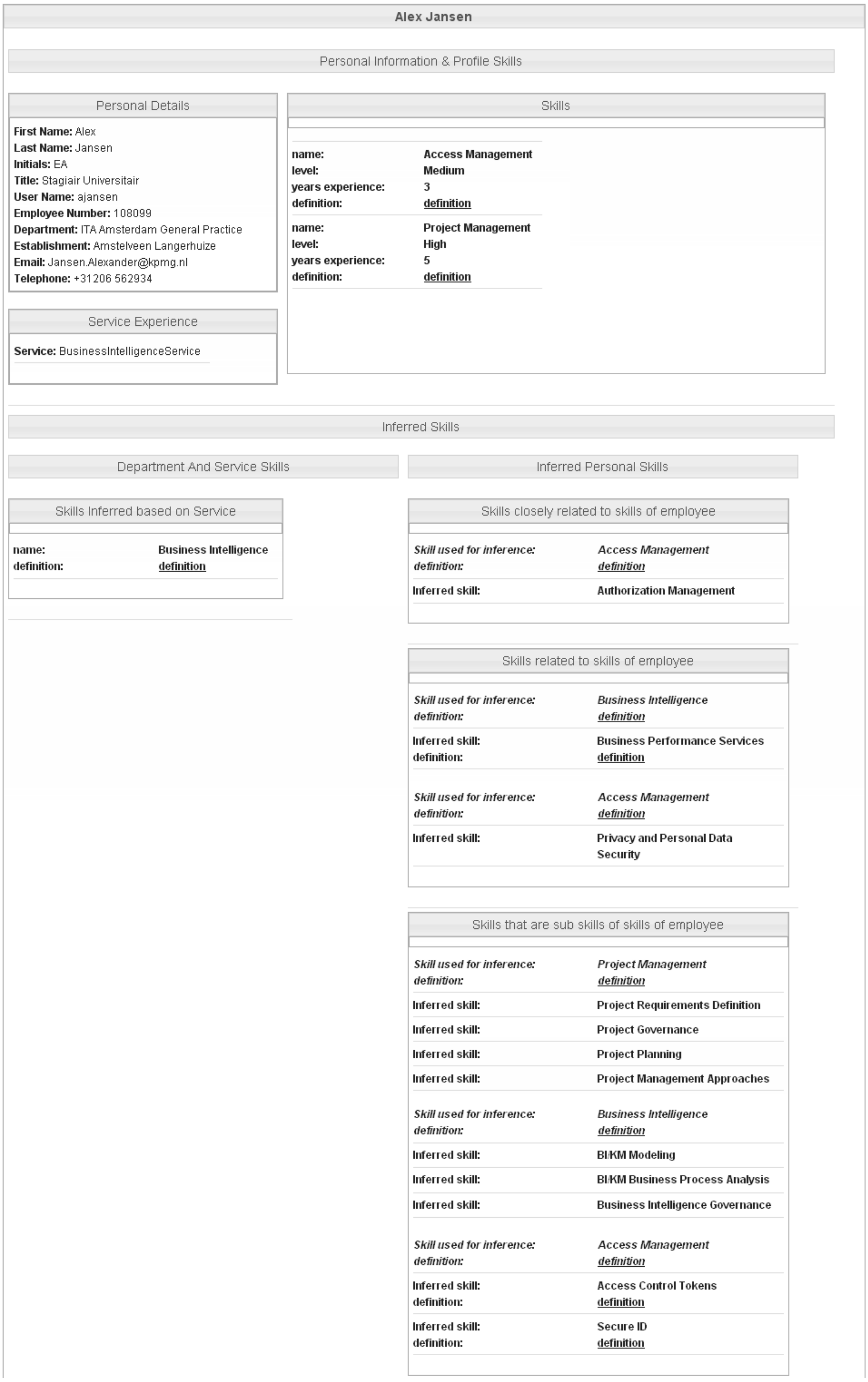


Figure 5.4 - Broad Search

In figure 5.4 the employee who is performing the search is looking for employees who have experience with Access Control Tokens. After performing the regular search no results came up. The system offers the possibility to perform a broad search as shown in figure 5.6. The employee is warned that the results of this search might be less trustworthy because an additional step was used in the inference.


Alex Jansen

Personal Information & Profile Skills

Personal Details

First Name: Alex
Last Name: Jansen
Initials: EA
Title: Stagiair Universitair
User Name: ajansen
Employee Number: 108099
Department: ITA Amsterdam General Practice
Establishment: Amstelveen Langerhuize
Email: Jansen.Alexander@kpmg.nl
Telephone: +31206 562934

Service Experience

Service: BusinessIntelligenceService

Skills

name:	Access Management
level:	Medium
years experience:	3
definition:	definition
name:	Project Management
level:	High
years experience:	5
definition:	definition

Inferred Skills

Department And Service Skills

Skills Inferred based on Service

name:	Business Intelligence
definition:	definition

Inferred Personal Skills

Skills closely related to skills of employee

Skill used for inference:	*Access Management*
definition:	*definition*
Inferred skill:	Authorization Management

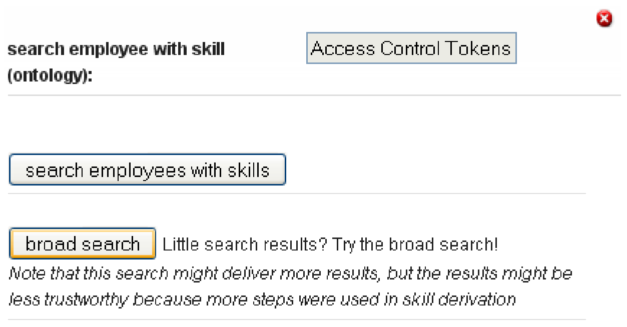
Skills related to skills of employee

Skill used for inference:	*Business Intelligence*
definition:	*definition*
Inferred skill:	Business Performance Services
definition:	definition
Skill used for inference:	*Access Management*
definition:	*definition*
Inferred skill:	Privacy and Personal Data Security

Skills that are sub skills of skills of employee

Skill used for inference:	*Project Management*
definition:	*definition*
Inferred skill:	Project Requirements Definition
Inferred skill:	Project Governance
Inferred skill:	Project Planning
Inferred skill:	Project Management Approaches
Skill used for inference:	*Business Intelligence*
definition:	*definition*
Inferred skill:	BI/KM Modeling
Inferred skill:	BI/KM Business Process Analysis
Inferred skill:	Business Intelligence Governance
Skill used for inference:	*Access Management*
definition:	*definition*
Inferred skill:	Access Control Tokens
definition:	definition
Inferred skill:	Secure ID
definition:	definition

Figure 5.5 - Detailed information about inferred skills (employee Alex Jansen)


search employee with skill (ontology): Access Control Tokens

search employees with skills

broad search Little search results? Try the broad search!
Note that this search might deliver more results, but the results might be less trustworthy because more steps were used in skill derivation

Figure 5.6 - broad search warning

This additional step is showed in figure 5.4. Here, the employee possesses the Privacy And Personal Data Security skill. The ontology defines that this skill is related to the Access Management skill. The ontology also defines that the Access Control Tokens skill is a subSkill of the Access Management skill. The broad search is thus able to retrieve all employees who are one additional relation further 'away' from the skill being searched. The query used in this particularly broad search is displayed below. It shows that a broad search looks for all employees that have a personal skill (ps) with a skill concept (sA) in the ontology which is related to some other skill concept (sB) in the skills ontology, where sA is related to sB with with either the SKOS related, the SKOS closeMatch or the skillRelation (subSkill / superSkill) property and furthermore has the a skill name that matches 'Access Controls Tokens'.

```
SELECT ?employee WHERE
{
   ?employee rdf:type ev:Employee .
   {
      ?employee ev:hasSkill ?ps .
      {
         ?ps ev:hasSkillConcept ?sA .
         {
            {
               ?sA skos:related ?sB
            }
            UNION
            {
               ?sA skos:closeMatch ?sB
            }
            UNION
            {
               ?sA ev:skillRelation ?sB
            }
         }
         .
         {
            {
               ?sB skos:prefLabel ?name
            }
            UNION
            {
               ?sB skos:altLabel ?name
            }
            FILTER (str(?name) = "Access Control Tokens" )
         }
      }
   }
}
```

Whenever a search delivers too many results (employees) the search can be specified further by providing additional search parameters. Adding an additional skill(s) to the search section results in a search for employees that possess all the mentioned skills.

A search on a free text field like the name fields results in two separate queries. One query tries to direct match the search term while the other query incorporates substrings. A search for employees with first name 'Alex' will return as direct match all employees with first name 'Alex' and the substring match query will return employees with first name 'Alexander' (or 'Alexia', etc.).

Performance
When the GUI is started the inference model is calculated. Calculating the inference model with 300 employees of which 50 contain skills takes about 5-10 seconds on a 2000mhz computer. When the model is calculated it is stored in RAM and answers to the described queries return almost instantly. It would be interesting to see how the inference model performs with a larger skills ontology and 100.000 employees, all having several skills. Probably the calculation of the model might take some time, but whenever the model would be stored in the RAM of a mainframe it will probably have a reasonable performance.

5.2 – Extendability of the application

One of the goals of this project was to create the proof of concept application in a way that would it be flexible and extendable. Semantic Web techniques were chosen because they are intended for integrating various datasets and the expectation is thus that the constructed application is indeed flexible and extendable. This section discusses the linking of an additional skills ontology to illustrate the easiness of extending the application while maintaining full functionality.

An additional skills ontology was constructed which contains 20 skills from the cryptography domain. The data model used to construct the cryptography ontology is different from the data model used to construct the KPMG skills ontology. Section 4.4.2 discussed three different model approaches which were considered to construct the KPMG skills ontology. The second approach, with one main skill class and then for each skill concept one individual of that same main skill class, was used for constructing the cryptography ontology. The property SKOS narrower, and its inverse SKOS broader, have been used to construct the hierarchy and the name of the skills is stored in the skillName property. Figure 5.7 shows a part of the cryptography ontology.

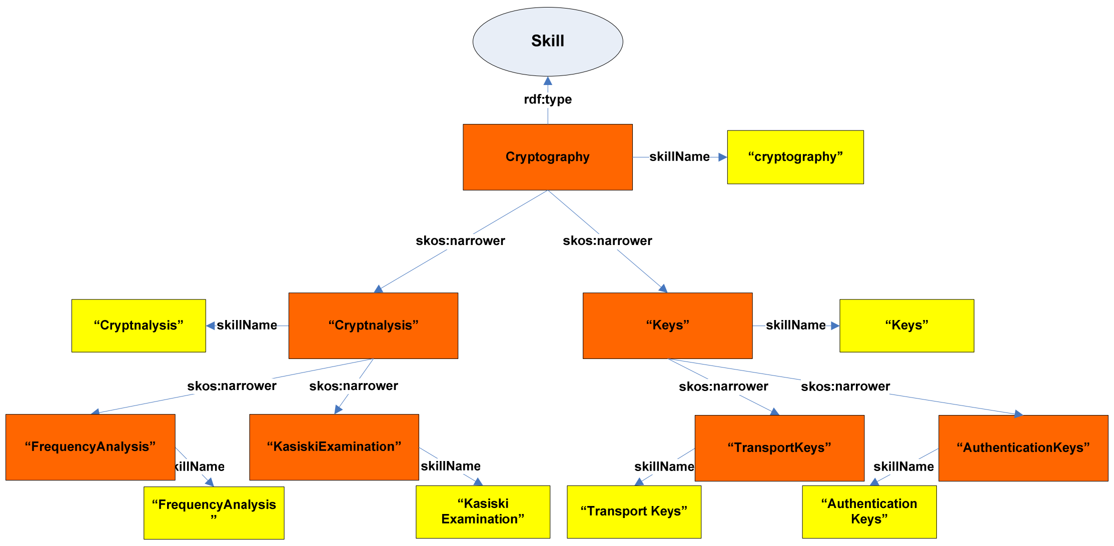
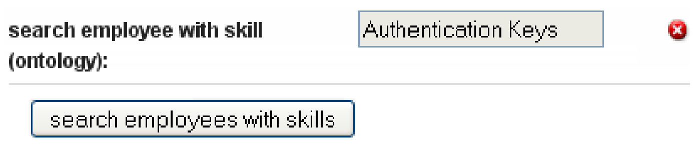


Figure 5.7 - A part of the cryptography ontology

The KPMG skills ontology does not contain many skills related to cryptography. It is therefore interesting to see if it is possible to use the cryptography ontology in combination with the KPMG skills ontology. Thanks to the Semantic Web techniques this can be accomplished by adding just two additional steps. First the application should load the cryptography ontology whenever the regular KPMG skills ontology is loaded and secondly a mapping between the ontologies should be defined. The mapping consists of statements which are added to the resulting ontology model. These statements declare concepts from the one ontology equal to concepts in the other ontology. The skill class from the KPMG skills ontology is declared equal to the skill class from the cryptography ontology with the owl:equivalentClass property. The prefLabel property, used to store the name of the skills in the KPMG skills ontology, is declared equal to the skillName property from the cryptography ontology with the property owl:equivalentProperty. These two statements were all that was needed to combine both ontologies! Employees can now add skills from the cryptography skills ontology to their profile and tell the system their personal skill level and the amount of year experience they have with this skill. Nothing in the application needs to be changed, not even the queries for expertise finding.

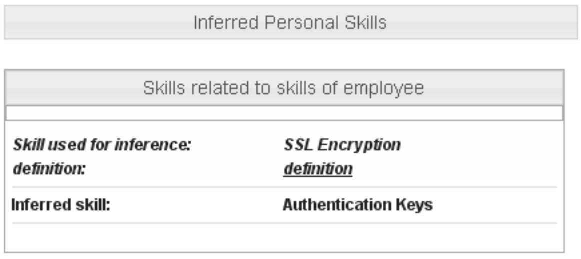
The cryptography ontology does not contain any additional relations (next to the hierarchical relations) about how the skills relate to each other. Without these relations the system cannot use the background knowledge to support the expertise finding process in any ‘smart’ way. However, the properties used in the regular skills ontology can be used to relate the skills from the cryptography ontology. It is also possible to relate the skills from the different ontologies to each other, which is illustrated in the next example.



search employee with skill (ontology):
Authentication Keys
search employees with skills

Figure 5.8 – Search for expertise from additional ontology

The SSL Encryption skill from the KPMG Skills ontology is declared SKOS related to the Authentication Keys skill from the cryptography ontology. This can be achieved by adding just a single statement to the model. The result is that the skill from the added cryptography ontology can now be used in the process of expertise inference. Imagine an employee who told the system that he possesses the SSL Encryption skill. Whenever the cryptography ontology is loaded and the described statement is added to the model, the knowledge of the system about the expertise of the employee is extended! The system is now able to use the fact that the SSL Encryption skill and the Authorization Keys skill have in some way a related meaning to increase the search performance. The search results for a search for employees with expertise related to Authentication keys will display the employees that possess the SSL Encryption skill as likely candidates. The application does not require any changes for this to work and even the same query can be used, as displayed below.


Inferred Personal Skills

Skills related to skills of employee

Skill used for inference:	*SSL Encryption*
definition:	*definition*
Inferred skill:	Authentication Keys

Figure 5.9 – Inferred expertise, based on relations between both ontologies

```
SELECT ?employee WHERE
{
   ?employee rdf:type ev:Employee .
   {
      ?employee ev:hasSkill ?ps .
      ?ps ev:hasSkillConcept ?s .
      {
         {
            ?s skos:prefLabel ?name
         }
         UNION
         {
            ?s skos:altLabel ?name
         }
         FILTER (str(?name) = "Authentication Keys" )
      }
   }
}
```

It is also possible to relate departments and services to skills from the cryptography ontology and infer expertise of employees based on their membership from these departments and services. Whenever the cryptography ontology would have had its own properties to relate skills to each other there should have been investigated to what extent the meaning of these properties corresponds to the meaning of the properties used in the KPMG skills ontology to relate skills. Corresponding properties could than be declared equivalent properties and the system would than also be able to infer skills of employees based on the relations used in the cryptography ontology. When the cryptography ontology would contain properties with meanings that do not correspond with meanings of the properties in the KPMG skills ontology, additional SWRL rules could be defined to make use those properties for inferring expertise.

Chapter 6 – Conclusion & Recommendations

This chapter starts with presenting the conclusions of this research and follows with some recommendations and suggestions for future work.

6.1 - Conclusion

Chapter one described the analysis of the problem situation and the main research question that was formulated was '*How can Semantic Web techniques improve the accessibility and the findability of KPMG employee information, in the area of expertise finding?*'. This question was divided into two sub questions which will now be discussed.

6.1.1 – Improved expertise finding

This sections covers the first sub question, '*How can background knowledge about the skills possessed by the KPMG employees be exploited to improve expertise finding within KPMG.*'.

To improve the expertise finding background knowledge about the expertise of employees was defined in an ontology. The constructed ontology defines skills and the relations between them. This skills ontology can be used in an expertise finding application to enable it to understand how the skills possessed by employees relate to each other. A proof of concept expertise finding application, which uses the skills ontology, was constructed to show how the improved expertise finding can be accomplished.

6.1.1.1 – Personal skill

The skills ontology contains skills which employees can possess. If this would be modeled as an employee who possesses a skill it would not be possible to add additional information about the relation between the employee and the skill (like the fact that an employee has a certain skill level or a certain amount of years experience with the skill). To allow employees to define their experience with a certain skill from the skills ontology, the personal skill object was introduced. Employees possess a personal skill which links to a skill concept in the ontology that represents the concept of the personal skill. A skill in the ontology is thus a skill concept which is possessed by everybody who has that particular skill, and the personal skill object is the personal object that can be used to store the experience that the employee has with the skill concept in the skills ontology.
Because it was not possible to express additional information about the relation between an employee and a skill in the ontology an additional class was created (the personal skill class). This solution, wherein an intermediaty entity is created, is known officially as an n-ary relation.

The following two sections discuss the process of expertise derivation via the constructed semantic relations between the skills in the ontology in more detail.

6.1.1.2 – Expertise derivation

Expertise about an employee can be inferred based on the personal skills that they possess. Personal skills are represented by skill concepts in the skills ontology which are related to other skill concepts. Several different relations have been used to relate the skills in the ontology.
To relate skills in the ontology the properties closeMatch and related from the SKOS vocabulary have been used. These properties are respectively used to indicate that skills have a closely related and a related meaning.
To exploit the relations between the skills SWRL rules have been defined which create an additional relation from the personal skill object to the related skill concept in the skills ontology. This enables easy querying and fast expertise finding. The name of the additional relation reveals wherefrom the inference was made (hasCloseMatchSkillConcept and hasRelatedSkillConcept for respectively closely related meaning and related meaning of skills).

The skills have also been placed in a hierarchy based on their meaning with the use of the rdfs subclassOf property. Because all skills are placed in the rdfs subclassOf hierarchy based on their meaning, another property was needed to account for the situation wherein the hierarchical relation can be used to infer expertise. In some cases, whenever an employee has a skill, it is likely that he or she will also have a skill which is a direct subclass of that skill. This particularly holds for the lowest levels of the hierarchy (an employee with the general skill 'IT Advisory' will probably not have expertise related to all the subclasses of this skill, while an employee with the more specific skill 'data storage' will most probably have some expertise related to one of the subclasses such as 'database development'). The custom defined property subSkill has been created to indicate that the hierarchical relation between the two skills can be used to relate the employee (his personal skill) to the hierarchically related skill. Again the name of the used property reveals wherefrom the relation was inferred (the property hasSubSkillConcept).

While some might argue that the need for such an additional hierarchal relation indicates that the data model is not entirely correct, because the subclassOf relation is ignored in the process of expertise derivation, I believe that it is. This discussion is related to the fact that the skills ontology is a combination between an ontology and a knowledgebase. The subclassOf relation has been used to structure the skills based on their meaning, and assuming that this has been done correctly the result is a skill hierarchy which is entirely logical for the employees. They can easily find the skill they are looking for. Also, The meaning of a skill class from the resulting hierarchy is represented by the meaning of all its subclasses. That this fact is currently not used in the process of expertise derivation does not mean that this representation will not be used in future expansions of the application.

6.1.1.3 – Expertise inference based on departments and services

Departments and services have been related to skills in the skills ontology to allow expertise derivation based on the membership of employees to the departments and services. Employees who are member of a department or indicated that they have experience with a certain service will be related to the associated skill concepts by a personal skill object. The SWRL rules will also infer additional experience of the employee based on these skills.

6.1.1.4 – Reuse of existing vocabularies

The decision was made to reuse existing vocabularies where possible. In some cases it turned out that the chosen vocabularies could not be reused because they were not able to represent the concepts as needed. This was the case with the ResumeRDF vocabulary which defines that a CV document contains both the person and its skill, while I needed to model a person who contains (possesses) a skill. The vocabularies that have been reused are all commonly used which enables a high interoperability which could be valuable in future extensions.

6.1.1.5 – Expertise finding

The background knowledge about the skills enables the application to provide search results to search commands that in the past would have returned empty. Searches can be formulated to retrieve employees that possess combinations of skills and even these kinds of search commands will likely produce results. The relevance of the search results can be inspected by the employee who is performing the search because the system presents the results in a way that it is clear wherefrom the skills were inferred. Another improvement is that it is now possible for employees to store the expertise level and the amount of years experience with the skills. This allows the employee who is performing the search to immediately see the experience that the employees in the search results have with the skill. Another improvement is that many employees that did not tell the system anything about their personal expertise can still popup as search result, because they are linked to skills in the skills ontology based on their membership of a certain department.
One of the problems with the current expertise finding applications is that the expertise indicators (like skills) are undefined and open to multiple different interpretations. This problem is solved in the constructed application by showing the hierarchy of the skills ontology in a treeview. Selecting a skill from the treeview will show additional information about the skill like its definition. The hierarchy and additional information about each skill enables the employees to understand the different skill concepts in the ontology which allows for more precise search formulation, which in turn results in more relevant search results.
Another advantage of the treeview is that it can be used by employees to define their expertise. Instead of showing all skills in an alphabetically ordered list the skills are now sorted on meaning and stored with additional information. Employees can thus easily and unambiguously define their expertise. While this wasn't the goal of the project it is still an important finding because more precisely defined expertise will result in more useful results.

6.1.2 - Extendability

Because the discussed proof of concept application addresses only one of the problems KPMG experiences with its expertise finding applications, the constructed application should be extendable and flexible so that it can easily be extended in the future. The second sub question was therefore, *'Is the chosen approach flexible and extendable?'*.

The choice for Semantic Web techniques really helped here because these techniques are intended for integrating various datasets and it turned out that the resulting application was indeed flexible and extendable. An additional skills ontology with a different data model could easily be linked to the expertise finding application. Only two steps were needed to let the constructed expertise finding application make use of the additional ontology. The application had to be informed about the existence of the additional skill ontology (it should load the additional ontology to the ontology model) and a mapping between the two ontologies had to be defined. This mapping consists of two statements which declare both skill classes from the different ontologies, and the properties used for the name of the skill, equal. The OWL equivalentClass and equivalentProperty properties were used to accomplish this.

Employees can now use the expertise finding application to also define their expertise with skills from the additional skills ontology, including skill level and years of experience with the skill. They can also search for colleagues with expertise related to the skill concepts in the additional skills ontology. No changes were needed to the application or the queries.

The properties which are used in the original skills ontology were used to relate the skills from the additional skills ontology to each other and to skill from the original ontology (and to departments and services). Without any additional changes the application was able to use these relations to infer additional expertise of employees.

6.1.3 – Concluding remarks

This section contains some concluding remarks about the expertise finding application and the skills ontology, it’s applicability outside KPMG, and the various tools and techniques used and their documentation.

The skills ontology is the key

The skills ontology is the most important component in the expertise finding application. The relations that are defined between the different skills are used to infer additional expertise of employees and this results in the improved accessibility and findability. Incorrect relations or relations that are only true in some cases might result in wrong expertise derivations. Because the search results show the inferences made the employees will probably recognize the wrong inferences themselves. These incorrect relations could then be reported and adjusted in the skills ontology, but the relations that only hold in some cases are more difficult to fix. This would require additional research.

Applicability outside KPMG

I believe the constructed expertise finding application can be applied in various different settings. The skills, departments, and services should be reviewed and probably adjusted for the changed setting but the model and inferences can be reused. Take for example the EEMCS faculty of the Delft University of Technology. The faculty contains various departments, all with their own research areas. These research areas could be modeled as services. Each employee of the faculty could then use the system to define its expertise. Expertise could be derived based on the fact that an employee is member of a certain department and/or has experience with research areas outside the department.

Tools, techniques, packages, documentation
This project used various different tools and techniques and most of them were in some way related to the Semantic Web. In most cases it was not straightforward in what ways the tools should be used together, could be used together, or could not be used together, while the ‘could not be used’ could be related to incompatible versions, bugs, or just to the fact that the tools were not meant to be used together. There are lots of information sources available which can help to solve these problems but it is hard to find the information that you’re looking for. I think that the information sources about the ontology languages could also use some improvements. While the new W3C site is a welcome improvement, I think it still is not sufficient. In the design phase I thought of ways to the store personal properties that employees have with skills from the ontology. I came up with the solution of the personal skill class, which can be used to express the additional properties of the relation between the employee and the skill in the ontology. I discovered the literature around this subject, called the n-ary relations problem, when I was finished with implementing my solution.

6.2 – Recommendations & Future Work

This sections gives some recommendations and gives some ideas for extending the functionality of the expertise finding application.

6.2.1 – Recommendations

While the constructed proof of concept application demonstrates the ideas that motivated this research and shows that these ideas work, the proof of concept application remains a first step. The skills ontology contains over 300 skill concepts but it is still an ontology with a focus on IT Advisory. Many other skills, and relations between these skills, should be defined and added to the ontology. Before presenting some recommendations and suggestions for future work a short note from literature is repeated because I think it is a really valuable recommendation. The Swiss Life case (see section 2.3) stated that an expertise finding application can only become a success whenever the system addresses the technical, the content, and the cultural dimension. This means that the system should provide a solid technical base providing all required functionality. Furthermore the system should address the content dimension which includes the instance data and skills which should be correct and up-to-date. The cultural dimension is concerned with creating a climate of trust and openness wherein employees are encouraged to share their expertise information. When one of these dimensions lacks the required amount of attention the system can easily become a failure. For KPMG it is important that the cultural dimensions gets the attention needed in order to convince the employees of the importance of devoting some time to define their expertise in the system. When all employees would give five to ten minutes of their time the system would be filled with lots of useful expertise information about each employee! Other recommendations were given throughout the different chapters, but the most important ones are repeated below.

- Employees should be able to easily define their expertise.
- Expertise indicators, used to define expertise, should be unambiguously defined.
- The usage of Semantic Web techniques should really be considered because they enable easy linkage of different datasets and allow for easy enrichment of data.
- Vocabularies which are used to express the employee data should be reused where possible to create an interoperable situation, which could be important for future extensions.

6.2.2 – Future Work

While the constructed proof of concept application contains more functionality than planned beforehand, there are still lots of useful extensions possible. Some of these are shown below. They are grouped together based on amount of effort needed to realize the functionality. One extension requires the involvement of an employee.

Requires implementation
To realize the extensions shown below only little research and implementation effort are required. Some of these extensions could be realized with very little effort.

- The personal expertise level should be standardized to prevent confusion about used terms.
- Support for other languages could be added to the ontology. Currently the language used within the ontology is only English but this can easily be extended. All annotation properties currently contain the language tag '@en' to indicate that the language is English. Adding support for another language only requires the addition of the same property with another language tag ('@de' for German for example).
- A link to the agenda information of the employees could be constructed. This would enable to search for employees with certain expertise which also have time to perform the tasks they are needed for.
- An easy to use notification system could be added to the application which enables employees to quickly provide comments and/or suggestions about the skills ontology (missing skills, wrongly placed skills, wrong inferences etc.).
- An automatic CV reader could be constructed to enable extraction of more employee information from the CV documents, like project experience or university degrees.

Requires research
The extensions shown here require more research before they can be realized.

- Investigate till what extent it would be possible to automatically collect skills of employees from several different sources like:
    - Planning sheets
    - Intranet pages and documents

  This is a complex problem but the skills ontology can be used to help detecting occurrences of skills. Automatic skill detection could save employees time in maintaining their personal skill expertise.
- A solution is needed for the problem that relations between skills in the ontology could be true in some cases, but not all.

Requires involvement of employee
This extension requires the involvement of an employee.

- It was already explained that it should be possible for employees to report incorrect inferences. There should be someone who processes all this feedback.

References

[1] Maryam Fazel-Zarandi and Eric Yu. Ontology-Based Expertise Finding. In *PAKM '08: Proceedings of the 7th International Conference on Practical Aspects of Knowledge Management*: 232-243, Berlin Springer-Verlag, 2008.

[2] Ted Lewis. Where the Smart Money Is? *Computer*, 32, no. 11: 134-136, 1999.

[3] M.T. Maybury. Expert Finding Systems. Technical Report MTR06B000040. Retrieved from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.125.3280&rep=rep1&type=pdf , 2006.

[4] Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. Formal models for expert finding in enterprise corpora. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*: 43-50, 2006.

[5] Dawit Yimam-Seid and Alfred Kobsa. Expert Finding Systems for Organizations: Problem and Domain Analysis and the DEMOIR Approach. *Journal of Organizational Computing and Electronic Commerce*, 13, no. 1: 1-24, 2003.

[6] D.W. McDonald and M.S. Ackerman. Just talk to me: a field study of expertise location. In *CSCW '98: Proceedings of the 1998 ACM conference on Computer supported cooperative work*: 315-324, New York, ACM, 1998.

[7] Geert-Jan Houben, Ad Aerts, Lora Aroyo, Kees van der Sluijs, Bas Rutten, and Paul De Bra. State of the art semantic interoperability for distributed user profiles. Telematica Instituut, 2005.

[8] W3C. W3C Semantic Web activity. Retrieved 2009-11-28 from: http://www.w3.org/2001/sw/, 2001.

[9] W3C. RDF Frequently Asked Questions. Retrieved 2009-11-28 from: http://www.w3.org/RDF/FAQ, 2009.

[10] Tim Berners-Lee. Semantic Web - XML2000. Retrieved 2009-11-10 from: http://www.w3.org/2000/Talks/1206-xml2k-tbl/, 2000.

[11] Steve Bratt. Semantic Web, and Other Technologies to Watch. Retrieved 2009-12-16 from: http://www.w3.org/2007/Talks/0130-sb-W3CTechSemWeb/#(25), 2007.

[12] Tim Berners-Lee. Semantic Web on XML. Retrieved 2009-12-20 from: http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html, 2000.

[13] W3C. Representing Knowledge in the Semantic Web. Retrieved 2009-12-20 from: http://www.w3c.it/talks/2005/openCulture/slide7-0.html, 2005.

[14] Donovan Artz and Yolanda Gil. A survey of trust in computer science and the Semantic Web. *Web Semantics*, 5, no. 2: 58-71, 2007.

[15] Unicode Consortium. Unicode Consortium FAQ. Retrieved 2010-01-03 from: http://unicode.org/, 2007.

[16] Tim Berners-Lee, Wendy Hall, James A. Hendler , Kieron O'Hara, Nigel Shadbolt, and Daniel J. Weitzner. A Framework for Web Science. *Foundations and Trends in Web Science*, 1, no. 1: 1-130, 2006.

[17] W3C. Cool URIs for the Semantic Web. Retrieved 2010-01-10 from: http://www.w3.org/TR/cooluris/, 2008.

[18] W3C. Extensible Markup Language (XML). Retrieved 13-10-2009 from: http://www.w3.org/XML/, 2009.

[19] SMARTMUSEUM Consortium. SMARTMUSEUM Report of User Profile

Formal Representation and Metadata Keyword Extension. Retrieved from: http://www.smartmuseum.eu/del/D2.1_v1.1_SM.pdf, 2008.

[20] W3C. Resource Description Framework (RDF). Retrieved 13-10-2009 from: http://www.w3.org/RDF/, 2004

[21] David Embley. Toward Semantic Understanding - An Approach Based on Information Extraction Ontologies. In *ADC '04: Proceedings of the 15th Australasian database conference*: 3-12, Australian Computer Society, 2004.

[22] W3C. RDF Vocabulary Description Language 1.0: RDF Schema. Retrieved 13-10-2009 from: http://www.w3.org/TR/rdf-schema/, 2004

[23] W3C. OWL Web Ontology Language. Retrieved 2009-12-09 from: http://www.w3.org/TR/owl-features/, 2004.

[24] Ilaria Torre. Adaptive systems in the era of the semantic and social web, a survey. *User Modeling and User-Adapted Interaction*, 19, no. 5: 433-486, 2009.

[25] W3C. OWL Web Ontology Language Overview. Retrieved 14-10-2009 from: http://www.w3.org/TR/owl-features/, 2004

[26] The Rule Markup Initiative. RuleML Homepage. Retrieved 2009-12-15 from: http://ruleml.org/, 2009.

[27] Pascal Hitzler, Jurgen Angele, Boris Motik, and Rudi Studer. Bridging the Paradigm Gap with Rules for OWL. Retrieved 2009-12-30 from: http://www.w3.org/2004/12/rules-ws/paper/9/, 2004.

[28] W3C. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Retrieved 2009-12-15 from: http://www.w3.org/Submission/SWRL/, 2004.

[29] W3C. RIF FAQ. Retrieved 2009-12-10 from: http://www.w3.org/2005/rules/wiki/RIF_FAQ, 2005.

[30] W3C. RIF Working Group. Retrieved 2009-12-15 from: http://www.w3.org/2005/rules/wiki/RIF_Working_Group, 2005.

[31] Tom Gardiner, Dmitry Tsarkov, and Ian Horrocks. Framework For an Automated Comparison of Description Logic Reasoners. In *Proceedings of the 2006 International Semantic Web Conference (ISWC 2006)*: 654-66, Springer, 2006.

[32] W3C. SPARQL Query Language for RDF. Retrieved 2009-12-06 from: http://www.w3.org/TR/rdf-sparql-query/ 2008.

[33] Jacqueline R. Reich, Peter Brockhausen, Thorsten Lau, Ulrich Reimer 'Ontology-Based Skills Management: Goals, Opportunities and Challenges' Journal of Universal Computer Science, vol. 8, no. 5, 506-515, 2002.

[34] B. Aleman-Meza, U. Bojārs, H. Boley, J. G. Breslin, M. Mochol, L. J. Nixon, A. Polleres, and A.V. Zhdanova. Combining RDF Vocabularies for Expert Finding. In *ESWC '07: Proceedings of the 4th European conference on The Semantic Web*: 235-250, Berlin Springer-Verlag, 2007.

[35] FOAF project. About FOAF. Retrieved 2010-01-08 from: http://www.foafproject.org/about, 2010.

[36] SIOC-project. Semantically-Interlinked Online Communities (SIOC). Retrieved from: http://sioc-project.org/faq, 2010.

[37] W3C. Introduction to SKOS. Retrieved from: http://www.w3.org/2004/02/skos/intro, 2009.

[38] Oracle. Java homepage. Retrieved 2010-05-10 from: http://www.java.com/en/

[39] Jena. Jena – A Semantic Web Framework for Java. Retrieved 2010-05-15 from: http://openjena.org/ .

[40] Protege. The Protege Ontology Editor and Knowledge Acquisition System. Retrieve 2010-05-15 from: http://protege.stanford.edu/ .
[41] FactPlusPlus. FaCT++ homepage. Retrieved 2010-05-15 from: http://owl.man.ac.uk/factplusplus/ .
[42] The Apache Software Foundation. Apache UIMA. Retrieved 2010-06-12 from: http://uima.apache.org/ .
[43] The Apache Software Foundation. Apache Directory Studio homepage. Retrieved 2010-06-12 from: http://directory.apache.org/studio/ .
[44] The Eclipse Foundation. Eclipse homepage. Retrieved 2010-06-12 from: http://www.eclipse.org/ .
[45] OpenCSV. OpenCSV homepage. Retrieved 2010-06-12 from: http://opencsv.sourceforge.net/#what-is-opencsv .
[46] The Apache Software Foundation. Apache Tika. Retrieved 2010-06-12 from: http://tika.apache.org/ .
[47] Ramon Antonio Parada. DOAC: Description Of A Career. Retrieved 2010-06-12 from: http://ramonantonio.net/doac/0.1/ .
[48] Uldis Bojars DERI. ResumeRDF. Retrieved 2010-06-12 from: http://rdfs.org/resume-rdf/ .
[49] FOAF. FOAF businessCard property. Retrieved 2010-06-20 from: http://wiki.foaf-project.org/w/term_businessCard .
[50] W3C. Representing vCard Objects in RDF. Retrieved 2010-06-20 from: http://www.w3.org/TR/vcard-rdf/ , 2010.
[51] Protégé. Frequently Asked Questions – OWL Full – Retrieved from 2010-02-08: http://protege.stanford.edu/doc/owl-faq.html .
[52] Protégé. Mailing list discussion about Owl Full – Retrieved 2010-08-04 from: https://mailman.stanford.edu/pipermail/protege-owl/2010-January/013228.html .
[53] Pellet. Discussion about support for OWL Full – Retrieved 2010-08-04 from: http://clarkparsia.com/pellet/faq/owl-full/ .
[54] W3C. Representing Classes As Property Values on the Semantic Web. Retrieved 2010-06-18 from: http://www.w3.org/TR/swbp-classes-as-values/ , 2005
[55] W3C. "OWL 2 Web Ontology Language: Model-Theoretic Semantics. Retrieved 2010-05-05 from: "http://www.w3.org/TR/2008/WD-owl2-semantics-20080411/ .
[56] W3C. SKOS Simple Knowledge Organization System - Home Page. Retrieved 2010-06-21 from: http://www.w3.org/2004/02/skos/ .
[57] S. Jupp, S. Bechhofer, and R. Stevens. School of Computer Science, University of Manchester. SKOS with OWL: Don't be Full-ish! Retrieved 2010-06-24 from: http://owl1-1.googlecode.com/svn-history/r654/trunk/www.webont.org/owled/2008/papers/owled2008eu_submission_22.pdf .
[58] Semantic Web Deployment WG. SKOS Concept Semantics: Patterns for Working With SKOS and OWL. Retrieved 2010-06-28 from: http://isegserv.itd.rl.ac.uk/public/skos/2007/10/f2f/skos-owl-patterns.html ,2007.

[59] Sun Microsystems. Servlets and JSP Pages Best Practices. Retrieved 2010-06-28 from: http://java.sun.com/developer/technicalArticles/javaserverpages/servlets_jsp/

[60] The Apache Software Foundation. Apache Tomcat. Retrieved 2010-06-28 from: http://tomcat.apache.org/ .
[61] David Decraene. jOWL Semantic Javascript Library. Retrieved 2010-07-08 from: http://jowl.ontologyonline.org/ .
[62] DynaTree. Dynatree javascript package. Retrieve 2010-08-20 from: http://wwwendt.de/tech/dynatree/index.html .