FAKULTÄT FÜR !NFORMATIK

Faculty of Informatics

SemProM

Semantic based Project Management

DISSERTATION

zur Erlangung des akademischen Grades

Doktorin der Sozial- und Wirtschaftswissenschaften

eingereicht von

Birgit Dippelreiter
Matrikelnummer 9804243

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Univ.-Prof. Dipl.-Ing. Dr. Hannes Werthner

Diese Dissertation haben begutachtet:

(Univ.-Prof. Dipl.-Ing. Dr. Hannes Werthner)

(ao. Univ.-Prof. Dr. Jürgen Dorn)

Wien, 22.04.2012

(Birgit Dippelreiter)

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

SemProM

Semantic based Project Management

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

Doktorin der Sozial- und Wirtschaftswissenschaften

by

Birgit Dippelreiter
Registration Number 9804243

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Univ.-Prof. Dipl.-Ing. Dr. Hannes Werthner

The dissertation has been reviewed by:

(Univ.-Prof. Dipl.-Ing. Dr. Hannes Werthner)

(ao. Univ.-Prof. Dr. Jürgen Dorn)

Wien, 22.04.2012

(Birgit Dippelreiter)

Erklärung zur Verfassung der Arbeit

Birgit Dippelreiter
Oberortsstrasse 34, 2440 Gramatneusiedl

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum)	(Unterschrift Verfasserin)

Acknowledgements

This thesis would not have been possible without the support of my supervisor, my colleagues, friends and family.

I am very grateful to Prof. Werthner, who supported me and for his feedback. I am thankful to Prof. Dorn for his valueable feedback.

Thanks to my colleagues and friends for great discussions, their feedback and advices and especially, for a great time.

Last but not least I'd like to say thank you to my husband Alex, my parents Linde and Sieghard, to my sister Barbara, my brother Bernhard and my sister-in-law Elisabeth for their patience and support.

Abstract

Successful project management (PM) has to obey predefined constraints, such as time, budget and resources. But very often, projects fail or are cancelled, because they cannot meet those constraints.

In this thesis I deal with common problems of current project management processes and systems and how to tackle them. Common problems include, but are not limited to interfaces between PM relevant systems, no standardized data storage, no predefined processes, no templates or no early detection of emerged problems. Another problem is that current project management systems mainly support the ongoing project phase and do not explicitly consider the initiating and closing phase of a project management life cycle. A further identified deficit is that most companies do not reuse knowledge of finished projects. In fact, very often they do not archive information in a central and well-structured storage.

By using semantic technologies, more precisely a PM ontology, semantic annotations and ontological reasoning, within an existing open source project management system, I am able to review if these weaknesses can be tackled. This system incorporates and links historical project knowledge that contributes to a more effective setup of upcoming projects; thus, obtaining a project management knowledge base.

From this, the research question is “do semantic technologies improve current project management processes?”. To deal with this question, I handled the following steps: I conducted interviews with project managers mainly of IT companies regarding strengths and weaknesses of current project management systems. On the basis of the interview outcome, I defined three scenarios, which highlight the shortcomings of current PM solutions and are carried forward as requirements for the project management ontology as well as for the prototype and to evaluate both of them. In addition, I accomplished an evaluation of current open source project management systems to identify current PM functionalities and to select a system as basis for prototyping.

Based on the scenarios, I designed the project management ontology, containing concepts of projects, such as tasks, employees, competences, documents and lessons learned. Aim was to cover a wide range of project relevant information. After implementing the prototype, including the PM ontology, semantic annotations and ontological reasoning, I carried out the evaluation by conducting interviews with project managers of IT companies.

Kurzfassung

Erfolgreiches Projektmanagement (PM) setzt bestimmte Vorgaben, wie Zeit, Kosten und Ressourcen voraus. Dennoch werden auf Grund von unterschiedlichsten Problemen Projekte vorzeitig für beendet oder gescheitert erklärt.

In dieser Dissertation beschäftige ich mich mit Problemen von aktuellen Projektmanagement-Prozessen und -Systemen und wie diese gelöst werden können. Häufige Probleme sind unter anderem

- nicht vorhandene Schnittstellen zwischen Systemen, die für das Projektmanagement relevant sind,
- keine einheitliche und zentrale Speicherung/Ablage von projektrelevanten Daten,
- keine vorgegebenen Prozesse/Standards und Dokumenttemplates
- oder kein "Frühwarnsystem" für aufkommende Probleme.

Ein weiteres Defizit von aktuellen Projektmanagementsystemen ist, dass sie im Grunde nur die"ongoing" Projektphase und nicht die Vor- und Nachprojektphase des Projektmanagement Life Cycles abdecken. Des Weiteren wird vorhandenes Wissen von abgeschlossenen Projekten nicht wieder verwendet. Mitunter dadurch, da Dokumente und Informationen nicht zentral und gut strukturiert abgespeichert werden.

Mit Hilfe eines Prototypen, bestehend aus einem open source Projektmanagement-System und semantischen Technologien, kann ich meine Annahme überprüfen, ob Schwächen von aktuellen PM Lösungen beseitigt werden können. Semantische Technologien sind: eine Ontologie über Projektmanagement, semantische Annotation von Daten und "ontological reasoning".

Dieser Prototyp verknüpft und integriert bestehendes Projektwissen, um dieses für neue bevorstehende Projekte wieder zu verwenden und dadurch zu verbessern. Des Weiteren wird dadurch eine Projektmanagement-Wissensbasis erstellt. Die daraus resultierende "research question" ist, ob semantische Technologien Projektmanagement-Prozesse verbessern.

Um diese Frage zu beantworten, habe ich die folgenden Tätigkeiten durchgeführt: Als Erstes habe ich Projektmanager in IT Unternehmen interviewt. Ziel war es Stärken und Schwächen von

deren Projektmanagementsystemen und Lösungen in Erfahrung zu bringen. Basierend auf den Ergebnissen habe ich drei Anwendungsfälle definiert, die die Probleme von Projektmanagement widerspiegeln. Diese Anwendungsfälle werden für die Anforderungen der PM Ontologie und des Prototypen, sowie für die Evaluierung verwendet. Neben den Interviews habe ich eine Evaluierung von bestehenden open source Projektmanagementsystemen durchgeführt. Zum Einen, um aktuelle Funktionalitäten von PM Systemen zu identifizieren und zum Anderen, um ein System für die Implementierung des Prototypen zu bestimmen.

Basierend auf den Szenarien habe ich die PM Ontologie entwickelt. Diese beinhaltet Konzepte von Projekten, Tasks, Mitarbeitern, Kompetenzen, Dokumenten und Erfahrungen von abgeschlossenen Projekten. Ziel war es ein möglichst breites Spektrum von projektrelevanten Informationen abzubilden. Nach der Implementierung des Prototypen habe ich diesen mit Projektmanagern und Mitarbeitern in IT Unternehmen getestet und evaluiert.

Contents

CHAPTER 1

Introduction

1.1 Motivation

Successful project management (PM) manages projects within predefined constraints, such as time, budget and resources. But very often, projects fail or are cancelled due to certain shortcomings. Emam et al. [26] outline that 11,5%-15,5% of IT-projects fail. Reasons stated are

- senior management is not adequately involved,
- scope and requirements change repeatedly,
- projects are over budget and
- a lack of necessary management skills is shown.

In addition, the Chaos Report [22] argues that the most common factors of project failures are related to a lack of user input, incomplete and changing requirements as well as specifications and lack of executive support. It states that 31,1% of all projects are cancelled while the success rate is only 16,2%.

Cerpa et al. [11] analyze 70 failed projects to identify the most common failure factors. In total, they received questionnaires describing more than 300 projects, conducted in the U.S, Australia and Chile. Nearly 93% of all projects failed due to time constraints. Other factors are that the projects had a wrong estimation of project duration, long hours were not rewarded and that risks were not adequately controlled and managed.

Emam et al. [26], Ringo Doe [22] and Cerpa et al. [11] refer to a statistical overview of failures mentioned above. In [54], Pinto and Mantel describe a study, including 97 projects with a view to identify controllable factors of project failures. They identified ten factors, which are relevant for a successful project:

- project mission,
- top management support,
- project schedule/plan,
- client consultation,
- personnel,
- technical tasks,
- client acceptance,
- monitoring and feedback,
- communication and
- trouble-shooting [54].

Regarding problems another identified deficit is that most companies do not reuse existing knowledge of finished projects. In fact they do not archive information in a central and well-structured storage [19]. In addition, current project management systems mainly support the ongoing phase and do not consider the initiating and closing phase of a project management life cycle. Honda et al. [42] define the most problems within IT projects during the initiating phase. Thus, project managment systems are not flexible enough to react, e.g., on lessons learned, which would increase ongoing projects within the PM life cycle.

Before explaining the project management life cycle, I give a definition of the term "Project Management" and "Project Management Systems.

The Project Management Institute [1] defines the term "Project Management" as follows: *"Project management is the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements. Project management is accomplished through the appropriate application and integration of the 42 logically grouped project management processes comprising the 5 Process Groups. These 5 Process Groups are:*

- *Initiating,*
- *Planning,*
- *Executing,*
- *Monitoring and Controlling, and*
- *Closing [1]."*

Ahlemann [4] defines "project management systems" in five categories:

- Single-Project Management Systems (S-PMS)
  "This software, ..., was designed to support project managers in their project planning and controlling tasks. It is mainly implemented as a desktop solution and can only be used to plan and control single, isolated projects [4]."

- Multi-Project Management Systems (M-PMS)
  "Multi project management systems offer functionality to coordinate a set of projects, at least in terms of resources. It is a software with a central database management system [4]."

- Enterprise Project Management Systems (E-PMS)
  "As soon as all projects within an organization need to be managed in a standardized and streamlined way according to a specific methodology, a new type of project management solution is required. The so-called enterprise project management systems are - at least partly - workflow-enabled and are highly flexible. Due to an advanced degree of configurability, they can be adjusted to specific methodologies without the need of programming. These systems support the entire project lifecycle, especially including portfolio planning and controlling [4]."

- Performance-Oriented Project Management Systems (P-PMS)
  "Although most enterprise project management solutions offer reporting capabilities, they are frequently not well prepared for performance measurement. They especially lack a reasonable set of standard key performance indicators (KPI), the possibility to define custom KPI and merge them into project and project management scorecards. Furthermore, the so-called performance-oriented project management solutions provide the user with advanced business intelligence and reporting features that make it easy to analyze performance data and set up individual performance reporting [4]."

- Knowledge-oriented Project Management Systems (K-PMS)
  "The last type of project management software systems additionally offers tools and features that facilitate work with project management knowledge, experiences and lessons learnt. For instance, an advanced skills management, a knowledge portal, a highly sophisticated document management, intelligent search funcionality or a project management knowledge base are typical characteristics of such applications [4]."

The project management life cycle consists of an initiating (starting), an ongoing (planning, executing and monitoring & controlling) and a closing (ending) phase [1] (depicted in Figure 1.1). The initiating phase of a project management life cycle covers the initialisation of a project, such as the project idea, the input of the project data into the PM system or the composition of the project team. The ongoing phase or intermediate phase handles the process of the project; for instance, in case of an IT project, the process of developing software. The closing phase includes the finalization of the software as well as storing information concerning lessons learned and final reports [1].

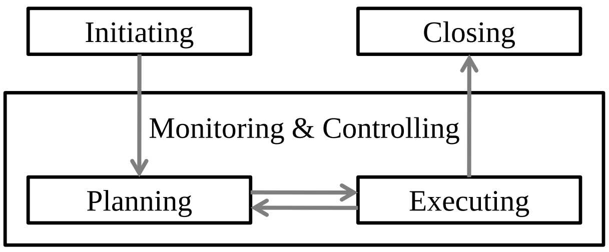


Figure 1.1: Project management life cycle [1]

In more detail, dividing the project management life cycle into four phases, I obtain the parts: project initiating, project planning, project execution and project closure. The activities contained in the initiating process are as follows [74]:

- Develop a business case
- Undertake a feasability study
- Establish the terms of reference
- Appoint the project team
- Set up a project office
- Perform phase review

The planning phase of a project includes the activities [74]:

- Create a project plan
- Create a resource plan
- Create a financial plan
- Create a quality plan
- Create a risk plan
- Perform phase review
- Contract the suppliers
- Create a procurement plan
- Create a communication plan
- Create an acceptance plan

The execution phase is usually the phase including the longest duration. Its activities are [74]:

- Build the deliveables
- Monitor and control, including time management, cost management, quality management, change management, risk management, issue management, procurement management, acceptance management and communications management.
- Perform a phase review

The last part of the PM life cycle, the closing phase, contains the activities [74]:

- Perform project closure
- Review project completion

1.2 Problem Statement

In this thesis I deal with common problems of current project management processes and systems and how to tackle them. Such problems include, but are not limited to little interfaces between PM relevant systems (archive, charging or calendar), no standardized storage, no predefined processes, no templates or no early detection of emerging problems. I argue that with the use of semantic technologies current project management systems can be improved and problems tackled. More precisely, one prediction is that the initiating and closing phase of the PM life cycle will be optimized. Figure 1.2 outlines the potential advantages of the use of semantic technologies within the PM life cycle. Knowledge of finished projects can be reused at any kind of the project process; at the initiating or ongoing phase. For instance, a new project is applied within the project management system, the system displays suggestions regarding personnel for the project team or the most suitable employees for a task. In addition, relevant information of lessons learned and documents of finished projects are outlined.

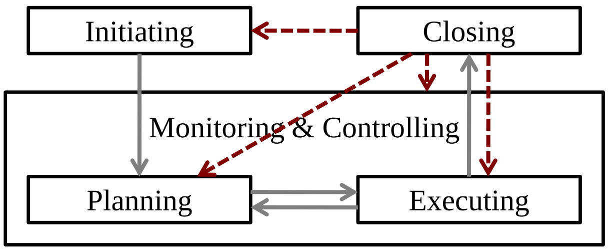


Figure 1.2: Semantic project management life cycle

Another conclusion is that ontologies and ontological reasoning can improve the relationships of employees to tasks and projects concerning the required competences. Until now, in the best case such information is stored in several tables within a database, which requires keys and reference keys. Thus, it complicates the search and, in addition, the search result is not as good/plausible as it could be. Another possibility is that the connection between this information (employees, competences, tasks/projects) even does not exist, because of different or no storage. With the use of an ontology, information of employees, their competences, tasks and projects, their required competences as well as further information is depicted and related with each other. Thus, search gets simplified, the outcome optimized and the allocation of employees to tasks improved.

To give a short introduction to the meaning of Semantic Web, it is *"an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation. It is the idea of having data on the Web defined and linked in a way that it can be used for more effective discovery, automation, integration, and reuse across various applications [...] where data can be shared and processed by automated tools as well as people [7, 13]."*

"The semantic web technologies are used to carry out knowledge acquisition, modeling, representation, publishing, storage, and reuse [13]."

This thesis is the outcome of the project "Semantic based Project Management", short "SemProM"; funded by a three year fellowship from the Austrian funding agency FFG (the Austrian Research Promotion Agency).

Research Question and Contributions of this Thesis

Within this thesis I argue that the incorporation of semantic technologies within current PM systems is able to tackle the shortcomings of current project management. As introduced, semantic technologies will enable a better search functionality and information retrieval as well as the reuse of already stored information for the purpose of reducing time and costs. Knowledge of finished projects can be reused for ongoing and new projects and to observe process and risks of project management life cycles, which will increase the quality of project management. In addition, ontologies improve the integration of information; for instance, between different systems.

Ahlemann et al. write in [4]: *"The more an organization collects performance-related data of projects and their methodology, the more the data can be used to improve the project management. ...Project management software can assist in this process by helping staff to carry out project management based on know-how from past projects. Examples are the estimation of effort based on experiences made in completed projects, the search for staff members with very specific skills or the sharing of knowledge stored in the form of documents [4]."*

They stated that the implementation of this system requires intelligent and flexible functionalities. They propose a possible solution in the following statement: ... *in order to find relevant documents for a certain problem a simple search algorithm based on catchwords will hardly* *lead to reasonable results. Instead, an intelligent search engine based on fuzzy results or using ontologies is more promising [4].”*

As a consequence, the main research question is:

“Do semantic technologies improve current project management processes?”

To enable a better process to evaluate this question, I subdivided it into four sub-problems related to contributions.

Problem 1:

Detailed information of relationships between projects/tasks, their required competences, time-management, staff members and their competences is often missing.

Most companies do not use only one system for project management. They have several ones for, e.g., time, cost, project and information management. Quite often these systems are not related; thus, it is nearly impossible to connect all project relevant information with each other.

Contribution 1:

Ontologies and ontological reasoning enable a better management of assignment of personnel to tasks as well as of competences, personnel and tasks.

The ontology will cover all necessary relations of different information items, such as project, employess, competences, documents and lessons learned. In addition, querying through the ontology enables an easier handling of the information.

Problem 2:

Mostly, current project management systems provide a kind of cockpit of up-to-date information of ongoing projects. Even so, with this information they are not able to fix upcoming problems.

Most PM systems enable a listing of ongoing projects, including information about, e.g., their end dates or process status. In fact, the majority of companies do not provide to store knowledge of finished projects centrally or to retrieve information easily. Thus, it is getting difficult to access out the stored knowledge to improve ongoing projects by, for instance, giving advice of lessonslearned.

Contribution 2:

With the use of semantic technologies, possible upcoming problems can be fixed by using knowledge of finished projects in advance. In addition, it reduces time and cost in fact of reusing knowledge. Thus, if up-to-date information of ongoing projects is shown in a kind of cockpit, project managers and members are able to deal in time with upcoming problems.

With the use of a project management ontology, information of projects, employees, competences, documents and lessons learned is related. This ensures that any information regarding projects can be linked and accessed. In addition, with the use of ontological reasoning knowledge of finished projects is used as proposal of new ones. For instance, lessons learned, documents and software code can be used to avoid a programming problem of a new project, which was already solved within a previous project.

Problem 3:

Because of non-uniform data storage, no annotation of information and no relations between project related and relevant information, search through archived information is difficult.

Up to now, information of projects is often not stored in a central archive or simply not annotated according to a predefined standardized guideline. Thus, recovering and reusing knowledge of finished projects is basically impossible.

Contribution 3:

Ontological description of project related and relevant information (of completed, archived and current projects) improves the search functionality in order to recover and reuse knowledge of finished projects, especially to identify related and similar information. With the use of an ontology and ontological description a project management system is becoming a knowledge base in the scope of project management.

By storing knowledge as instances within an ontology, including predefined annotations of the information, complex querying through the information is enabled and more easing applicable than using a database. If a database is used each table requires keys and reference keys. Thus, it becomes complicated. Using an ontology enables a simple tree of concepts, related to each other quite easily. In addition, relations to documents are realized and thus, a knowledge base in the field of project management arises.

Problem 4:

The project management life cycle consists of the phases: initiation, ongoing and closure; while the ongoing part is more or less covered by existing PM systems, the starting and ending phases are still badly covered by current PM solutions.

Honda et al. [42] claim that the most problems within IT projects are during the initiating phase. The ending phase of a project is effectively the "stepchild" of the PM life cycle, as nearly no one wants to document experiences, mistakes or wrong team compositions (lessons learned). In addition, writing documents like the final report are also unpopular. The last step of closing a project is to archive all project relevant information. Central data storage or uniform rules how to store the information mostly do not exist. As a consequence, the starting phase of a project cannot be supported and covered as it should be. If information of stored projects is retrieved and reused, new projects are supported by the knowledge of mistakes or experiences of already finished ones.

Contribution 4:

Ontologies and ontological reasoning improve the start and end phases of a project management life cycle.

With the use of an ontology, knowledge of different information items concerning project management is stored and related. Thus, access to any kind of information as well as related ones is guaranteed. In addition, querying through the stored knowledge is simplified. Thus, the ontology and consequently, ontological reasoning ensure a uniform data storage of project knowledge, like documents or lessons learned. This enables an access of knowledge of finished projects to improve starting ones, by, for instance, proposals of similar projects or team suggestions.

Each of these contributions are covered by the scenarios described in chapter 4 "Scenarios", which outline on the one hand common problems of current PM and on the other hand the requirements of the project management ontology, the prototype and the evaluation process. The evaluation contains the ontology, the prototype and the research question as well as contributions.

1.3 Methodological Approach

Working Steps

To answer the main research question and evaluate the contributions, my approach consists of the following main steps:

1. **Literature review** about similar work concerning project management enhanced with semantic technologies, existing ontologies and methods, which can be used for my approach.
   The purpose was to differentiate and define my approach compared to existing semantic project management approaches, and to identify which methodological approach I can use for the treatise of this thesis or what ontological engineering approach most fits to my needs. Furthermore, it was necessary to get an overview of existing ontologies regarding

project management, competences, documents or time and date. With this information the decision to use existing ones and merge them or to implement a new ontology was finalized.

2. **Interviews** to get an overview of current PM solutions as well as their weaknesses, strengths and standards.

3. An **evaluation of open source project management systems** to analyze the capabilities of current solutions and to select a system for prototyping.
   More interesting would be to carry out a commercial PM system for prototyping, because systems like MSProject are quite often used by companies. Even so, I carried out an open source PM system due to the fact that the source code of commercial ones were not available.

4. Based on the interviews, a **specification of the scenarios**.
   The scenarios highlight common shortcomings of current PM. In addition, they outline the requirements of the ontology as well as of the prototype. They were also carried out to evaluate the prototype and ontology and in further work to evaluate the research question and contributions.
   I decided to carry out scenarios instead of a detailed specification of the requirements, because the requirements were becoming more descriptive.

5. Design of an **ontology** containing, for example, project management, competences of employees, lessons learned and document information.
   The main purpose of this work was to define the meaning of the concepts and properties precisely and to enable a flexible structure of the relations. The implementation of the ontology followed an iterative process.

6. **Implementation of the prototype**, containing

   a) the project management ontology, including concepts, properties and relations as well as first instances to test the ontology,

   b) choosing the APIs (Application Programming Interface) to insert and update instances within the ontology and to request instances,

   c) adapt and enhance the PM system with the APIs to connect the system and the ontology and to enable predefined queries.

7. **Evaluation of my approach** using an evaluation guideline by evaluating the prototype and the ontology based on the scenarios.

Approach

The methodological approach of this work follows Hevner et al.' [39] guidelines for design science in information systems research. Hevner et al. illustrate a conceptual framework for executing, understanding as well as evaluating information system research, highlighted in figure

1.3. Information system research is a combination of behavioral and design-science. Thus, it is a process of developing theories and artifacts and evaluating them.

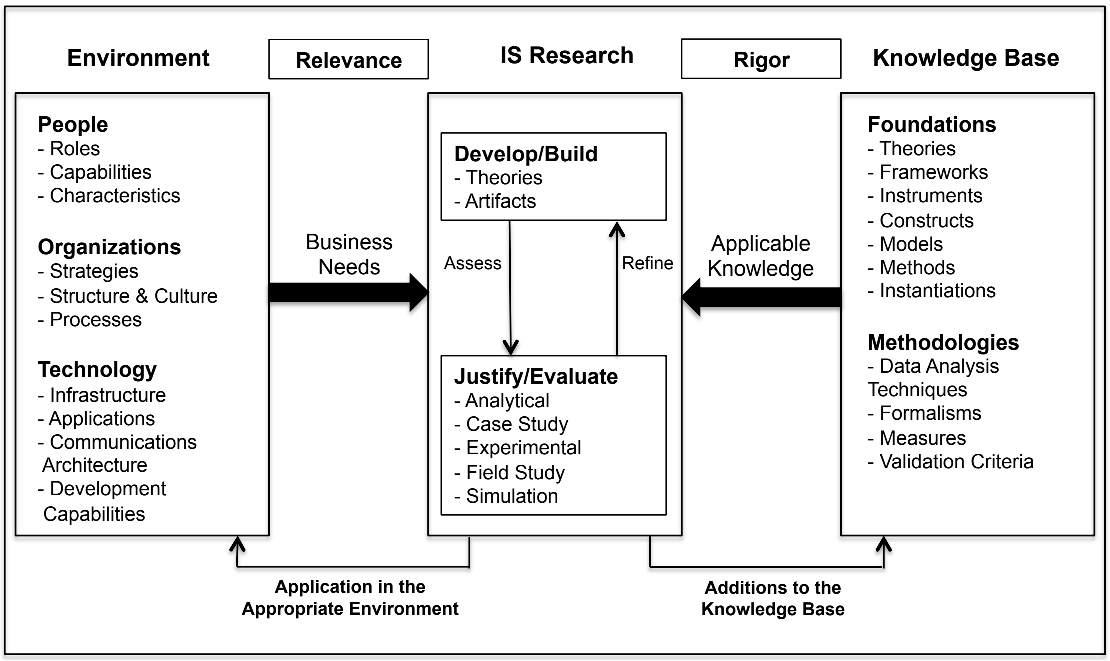


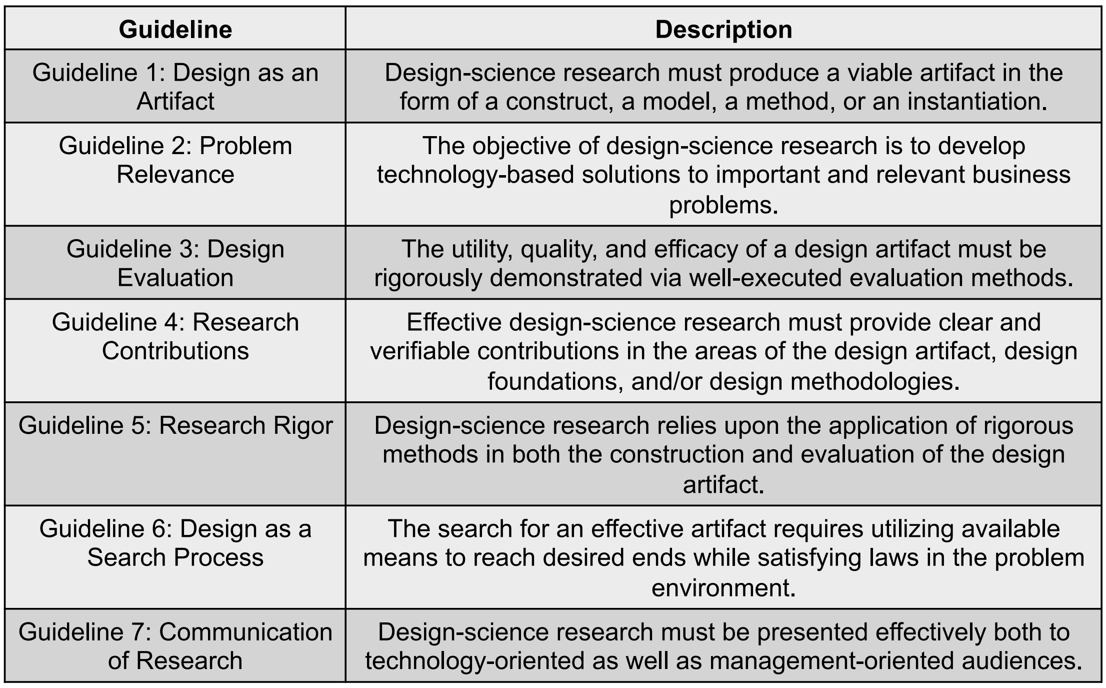
Figure 1.3: Information System Research Framework [39]

Hevner et al. introduces seven guidelines for design science in information systems research. A short description of the seven guidelines is listed in table 1.4.
However, I only outline the guidelines, which I assume to be essential for my work:

Problem Relevance:

I used 11 face-to-face interviews and a desktop study to identify the problem relevance. Most of the interviewed companies are located in the domain of IT and their working range contains software development and services [19]. The contact persons were mainly project managers. The interviews contained questions as regarding data storage, reusing knowledge and strengths and weaknesses of current PM solutions in the domain of IT. To clarify the main results, the recordings of the interviews were transcribed and the main content was outlined. As a result, the key findings can be summarized as follows [19]:

- Companies miss a central, well structured and standardized data storage.
- Most of the companies do not reuse knowledge of already finished projects to improve ongoing and further projects.
- Project manager as well as project members miss an up-to-date cockpit of ongoing projects.



Guideline	Description
Guideline 1: Design as an Artifact	Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation.
Guideline 2: Problem Relevance	The objective of design-science research is to develop technology-based solutions to important and relevant business problems.
Guideline 3: Design Evaluation	The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods.
Guideline 4: Research Contributions	Effective design-science research must provide clear and verifiable contributions in the areas of the design artifact, design foundations, and/or design methodologies.
Guideline 5: Research Rigor	Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artifact.
Guideline 6: Design as a Search Process	The search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment.
Guideline 7: Communication of Research	Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences.

Figure 1.4: Design-Science Research Guidelines [39]

- Companies often have more than one system for project management; hence they would need interfaces between those systems to prevent redundant work.

Comparing problems in project management stated and listed in the literature at the beginning of this chapter and the outcome of the interviews, they are quite coherent.

Design as an Artefact:

For my work I identified three artifacts; scenarios, a prototype and an ontology.

Based on the interview results I identified scenarios, which cover the requirements of the ontology as well as prototype. In further consequence, the scenarios serve as a basis for the evaluation process of the two other artifacts.

The aim of the software prototype was to evaluate the research question whether semantic technologies improve current project management processes. Therefore, the existing open source PM system dotProject [24] was enhanced with semantic technologies; more precisely, with a PM ontology, semantic annotations of the information and ontological reasoning. To evaluate the system, the scenarios, described in chapter 4 “Scenarios”, were applied.

The prototype will improve processes, such as the information retrieval or the setup of a new project team.

The project management ontology represents the relations and connections between the different information items regarding project management. For instance, tasks, employees and their competences. Basis of the ontology were the outcome of the interviews, the scenarios and some ideas of existing ontologies, such as the project management ontology of the Semantic MediWiki [37].

Design Evaluation:

Based on the scenarios the prototype will be evaluated. Thereby, the ontology artifact is part of the prototype; they can be evaluated at once.

Research Rigor:

Concerning my project, I used several research methods. As already mentioned above, the project SemProM, as a whole, is covered by Hevners guidelines for design science in information systems research [39].

For the conducted interviews I carried out a qualitative approach. Qualitative research is characterized by more flexibility and is in its essence explorative [2]. The main focus is to get lots of information of a small group. To translate it to my work, I did face-to-face interviews with eleven companies to get usable information about current project management.

For the implementation of the ontology I relied on the methodologies of Uschold and King [70], [71] and Grüninger and Fox [34]. A detailed description of the ontological engineering approach is given in chapter 5 “Ontologies”.

Based on Hevner et al. guidelines [39], I used a descriptive as well as observational method to evaluate the ontology as well as prototype. For the descriptive method I carried out the predefined scenarios as evaluation process. Based on the scenarios I identified real use cases. In addition, the observational method enabled a case study with project managers and members within a business environment using the evaluation guideline.

1.4 Structure of this Thesis

The structure of the thesis is as follows:

Section 2 “Related Work” highligths the state of the art literature of semantic project management systems; in addition, information of required input, such as available ontologies or ontological engineering approaches.

Section 3 “Background Work” outlines the interviews, the questionnaire and their results as well as the evaluation of open source project management and Semantic Desktop systems. This chapter highlights the base of the project.

The scenarios, introduced in section 4 “Scenarios”, cover the requirements concerning the semantic project management prototype and the project management ontology. In addition, they serve as a basis for the evaluation of the prototype and thus, of the contributions. The ontology, the core of the prototype, its concepts, properties and relations, are displayed in section 5 “Ontology”.

In section 6 “Prototype”, I discuss the open source project management system dotProject, the installation of the required software and the structure of the prototype; the modified and added scripts, the interfaces and architecture. The evaluation of the ontology and the prototype as well as the evaluation of the research question and the contributions are handled in section 7 “Evaluation”. Conclusion and further work are outlined in the last section “Conclusion & Outlook”.

CHAPTER 2

Related Work

In this section I discuss the state of the art and related work within the range of my work. In addition, I seperate them into different domains. The first part illustrates similar work in the field of semantic project management; followed by Semantic Desktop Systems. The last part contains the domain of ontologies; existing ontologies and ontological engineering approaches.

2.1 Semantic Project Management

In the field of semantic project management, Mohammadi et al. [48] developed a semantic project management system with the main focus on semantic web services. They also used the ontology PROMONT as a base ontology. The difference to my approach is that Mohammadi et al. implemented a new project management system with the use of semantic web services, while I am enhancing the existing open source project management system dotProject. In addition, my focus lies on reusing already stored knowledge.

The Semantic MediaWiki [37], which is distributed by Ontoprise [51], is also related to the field of semantic project management. They implemented a project management ontology and provides collaborative knowledge management via a semantic wiki. This work, especially the PM ontology, influenced some relations and properties within my project management ontology, like the concept information or the property date.

Another Wiki, the semantic wiki QMS, implemented by the GRIHO group is used to improve the quality of the technology transfer projects. QMS is a quality management system and provides outstanding capabilities for document management [31]. The implemented QMS ontology is equivalent to the ISO9001 standard. In general, the approach of using an existing system and enhance it with semantic technologies is similar to mine; the difference is that both [37] and [31] apply a wiki, a content management system, to implement a project management system, whereas I have chosen a project management system.

Another related work is the project “CONTO” - Ontology-based Competencies Managament in Information Technology” [49]. This project is funded by the Romanien Ministry of Education and Research and deals with a generic framework of an intelligent information system for competence management. Their vision of a competence management system (CMS) includes the functionalities to provide the knowledge of the competences of employees and to support the acquisition of knowledge of competences. Therefore, they implemented a CMS ontology containing information of company jobs, competences, domains, groups and persons. The domain concept is divided into sub-concepts, such as technical domain. The concepts company jobs and persons have the same sub-concepts, like Programmer or System Engineer. The main difference of their approach to mine is that their focus is lying on competence management, while my focus is on the “whole” project management, just including competences. Furthermore, the sub-concepts of the concept competences of my PM ontology is classified into different competence types (methodological, professional, social and personal). In contrast, their classification is divided into behaves for, knows how and knows what.

Sun et al. [68] introduce an ontology-based knowledge management system (KMS) for product management to provide knowledge, like artificial sources, multiple experts or domain terms. In addition, this system should be integrated into existing tools, such as data stores and should contain work processes to reconstruct the history of an information item. For implementing the system, they created an information resource, product semantic, organization competence and management activity ontology. Sun et al. focus is to improve product development between several organizations. My focus is to improve project management within an organization, but also between companies.

Another interesting work is carried out by Xu et al. and explained in [75]. They present CaBMA, Case-Based Project Management Assistant, a prototype for a knowledge based system. This tool implements case-based reasoning on top of project management software, like MS Project. Aim of this prototype is to assist project planners in creating new work breakdown structures. CaBMA extends current project management systems by adding a knowledge layer [75]. The difference to my approach is that they just reuse project plans, while I recover and reuse any kind of knowledge of projects to improve and simplify new projects.

In [6] Anumba et al. present a collaborative project information management system in a semantic web environment. Thus, they implemented an ontology-based approach to project information management in the domain of construction projects. Similar to Mohammadi et al. [48], they also used semantic web services. Anumba et al. outline that the main component of their system is the ontology mapping module, which coordinates the requests and responses. In addition, their system contains an ontology editor to generate new ontologies. Their focus lies on the relationship of two ontologies and how to map them. Thus, they receive a quite flexible data structure, but must also be careful to map the information correctly. In contrast, my prototype contains one project management ontology, which is the base of the PM knowledge base. Thus, every user has to work with the same information structure.

XProM (Expert Program Manager), a collaborative knowledge-based project management tool is introduced in [40] by Hewett et al. The system includes a collaborative project organizer module and a rule-based project control advisor. This advisor is used to identify critical tasks and evaluate the progress of tasks. Similar to Xu et al. [75] also this approach just includes a small range of the domain of project management, whereas I try to cover the whole process of project management.

A semantic web-based approach to knowledge management for grid applications, such as e-Business or e-Learning, is covered by Chen et al. in [13]. Therefore, they use ontologies for knowledge acquisition and modeling. To represent the knowledge they use the Web ontology language and to support decision-making semantic-based reasoning is carried out. Therefore they use a DL-based reasoner. While their approach covers knowledge management in more general, I deal with knowledge management in the domain of project management.

Another work, carried out by Colucci et al. [15], deals with a Description Logic approach to the semantic-based composition of teams in an organization. For their work, they are using the relationship one task and many employees within the project team. Likewise to my approach, they also compare and match the competences of the employees with the project specification. While competences of employees are just one part of my project management ontology, in the work of Colucci et al. competences are an important issue, the central module to compose teams.

Although there are lots of similar approaches and ideas, the results are quite different. Most of the approaches mentioned above only deal with parts of the domain project management. Whereas I try to cover the whole process of project management including projects, tasks, employees and their competences, information (documents) and lessons learned. In addition, some use semantic web services to implement their idea and nearly all implement their PM system or knowledge base from scratch. There are several approaches which would be conform with my approach, but most of them are implemented quite different or they carried out a different approach how to solve the problem. Furthermore, either they are too complex or too specific. For instance, the Semantic MediaWiki [37] of Ontoprise has similar approaches regarding the ontology, but instead of a project management system they use a wiki. Mohammadi et al. [48] have a quite similar idea to mine, but their approach is totally different to mine. They use the ontology PROMONT, which ist too general for my purpose, and implement a new PM system using semantic web services. In contrast, my work is based on an existing open source PM system.

2.2 Semantic Desktop

Another semantic system which influenced my work and was originally part of this work is the Semantic Desktop, which provides a personal information management. Within such a system, the stored information is represented by triples in RDF (Resource Description Framework). In addition, the information is related to each other and the metadata regarding information is stored in an ontology.

There are already some existing implementations, such as IRIS [14] or Gnowsis [61, 62]. While IRIS belongs to the CALO research project at SRI International [14], Gnowsis was part of the NEPOMUK project [62]. DeepaMehta [60], an open source semantic desktop, is based on the Topic Maps standard. For the implementation process, they carried out psychological requirements; cognitive psychology to facilitate a cognitive adequate working environment. For the user interface, a graph visualization is used [60]. A further semantic desktop is Haystack [43]. Similar to the other semantic desktop systems is Haystack a personal information management system to provide managing documents, email, tasks, dates and related information. Main focus of this alternative is the Resource Description Framework (RDF).

Current work in the field of Semantic Desktop is to enable collaboration between such systems [17,59]. For my work interesting aspects of the Semantic Desktop are parts of their ontologies, such as date and time or documents. In addition, the idea of a Semantic Desktop is quite similar to a semantic project management system. The difference is that originally the main focus of a Semantic Desktop is to improve personal information management, while semantic project management improves the collaboration and work of several persons. For my work the different ontologies of the Semantic Desktop, e.g., documents or date, were quite helpful and served as a model to implement the proejct management ontology.

2.3 Ontologies

While developing the PM ontology a literature review regarding already existing ontologies was done. For the purpose the main focus lies on project management and date and time ontologies. Thus, e.g., the PROMONT ontology [3] formalizes the typical elements for project structuring. The further development of this ontology, divided into a project and product ontology, was done as PR^2ONTO in the project PERMETER [35]. Due to the fact that PROMONT covers different project management specifications, especially the DIN 69901 [3] it is too general for my purpose, identified by the scenarios. Regardless, I looked closely to the PROMONT ontology and used, e.g., the properties time and responsible person.

In addition, the Semantic MediaWiki [37] has to be mentioned as well. This Wiki contains a project management ontology and provides the collaborative knowledge management. Contrary to PROMONT it fits to certain requirements concerning the scenarios. For instance, properties regarding planned and actual start dates or the concept structure of Resource and Person (Person is a subclass of Resource).

Other ontologies, which influenced my building approach, are included in the Semantic Desktop Gnowsis [62], especially date and time. During the implementation process of the PM ontology I realized that a specific date and time ontology is not needed. For my purpose just date and time properties related to tasks are sufficient.

Due to the fact I categorized the competence part of the ontology only in a first structure, I identified the categorization of [41]. For further development competence ontologies, like the ontology of Biesalski [8], can be used.

Another ontology related to project management is described by Kwon et al. [46]. At the Korea Institute of Science and Technology, an ontology for R&D project meetings was designed. Therefore, they analyzed and classified meetings and related data. I will not use this ontology, because the project management ontology will not cover the aspect of meetings.

Dong et al. describe in [23] a multi-site project track and trace ontology. It contains a hierarchy of project organization, an employee, project and criterion ontology. While the project organization hierarchy describes and classifies different roles of employees, the employee ontology identifies the employee's name and his responsibilities within a project. The project ontology contains all relevant information regarding a project, such as name, date or status and is related to the criterion ontology, which can be compared with tasks of a project [23]. Despite the fact this ontology covers some aspects of the PM ontology, it does not contain my required terms, relations and properties. Thus, I do not use this ontology.

While I was going through different ontologies of various domains, such as time, date, project management, documents or competences, I realized and decided that none of these ontologies fits to the predefined requirements. In addition, the idea to merge a date and time ontology with the project management ontology was dumped due to the fact that date and time modelled as properties, specifically related to concepts, would accomplish my requirements. On the one hand, other ontologies follow the same approach, like the ontology of the Semantic MediaWiki, and on the other hand, the requirements do not claim date and time as separate concepts.

As a result, I implemented the ontology by myself, including some ideas and solutions of existing ontologies. For instance, some properties and relations of the MediaWiki [37] or PROMONT [3] as well as competence ontology.

Ontological Engineering Methodologies

When starting with the work of the ontology, I had to detect which ontology development method and methodology I will carry out beside the decision using existing ontologies or building a new one. For this purpose, Gomez-Perez et al. [33] give an overview of classical methods for building ontologies. Within their publication they mention the CYC method [47], the Grüninger and Fox methodology [34], the Uschold and King method [70], the KACTUS approach [45, 64], the On-To-Knowledge methodology [65, 67] and the METHONTOLOGY [29, 30].

In the 1980s CYC, a knowledge base (KB) about the world, was started by the Microelectronics and Computer Technology Corporation and it is still under development. Building the CYC ontology, the following steps were carried out [33, 47]:

- Process 1: Manual coding of articles and pieces of knowledge.

- Process 2: Knowledge coding aided by tools using the knowledge already stored in the Cyc KB.
- Process 3: Knowledge codification mainly performed by tools using knowledge already stored in the Cyc KB.

This method has to be mentioned, because CYC is the largest ontology, which is in use, for the most part. Although, the methodology is not carried out for building other ontologies.

Uschold and King's methodology contains four processes for developing an ontology [33, 70]:

- Process 1: Identify the purpose and the scope
- Process 2: Build the ontology
- Process 3: Evaluate
- Process 4: Document

A detailed description of this methodology is given in chapter 5 "Ontologies".

Grüninger and Fox's methodology is similar to Uschold and King's. Even so, they provide six steps do build an ontology [33, 34]:

- Process 1: Identify motivating scenarios
- Process 2: Elaborate informal competency questions
- Process 3: Specify the terminology using first order logic
- Process 4: Write competency questions in a formal way using formal terminology
- Process 5: Specify axioms using first-order logic
- Process 6: Specify completeness theorems

Likewise to Uschold and King, a detailed description of this guideline is depicted in chapter 5 "Ontologies".

The methodologies of Uschold and King and Grüninger and Fox are quite similar. The methodology of Grüninger and Fox is more formal, while Uschold and King's approach is less detailed.

The KACTUS methodology is to develop ontologies by carring out the requirements of the application development. Thus, the ontology is dependent of the application. The process is as follows [33, 45, 64]:

- Process 1: Specification of the application

- Process 2: Preliminary design based on relevant top-level ontological categories
- Process 3: Ontology refinement and structuring

I did not use this approach to implement my project management ontology, because it is too general and superficial for my purpose.
METHONTOLOGY has its origin in knowledge engineering approaches and software development processes. The methodology contains *"the identification of the ontology development process, a life cycle based on evolving prototypes and techniques to carry out each activity in the management, development-oriented, and support activities [33]."*

The tasks of the conceptualization activity are [29, 30, 33]:

- Task 1: Build the glossary of terms
- Task 2: Build concept taxonomies
- Task 3: Build ad hoc binary relation diagrams
- Task 4: Build the concept dictionary
- Task 5: Define ad hoc binary relations in detail
- Task 6: Define instance attributes in detail
- Task 7: Define class atrributes in detail
- Task 8: Define constants in detail
- Task 9: Define formal axioms
- Task 10: Define rules
- Task 11: Define instances

Despite this methodology has its root in software development processes and provides the most precisely descriptions of activities, it does not fit to my requirements for implementing the ontology. For instance, with the use of Scenarios, like in Grüninger and Fox's approach, I am able to define requirements for the prototype as well as ontology in a visual form.

The On-To-Knowledge methodology contains the following processes [33, 65, 67]:

- Process 1: Feasability study
- Process 2: Kickoff
- Process 3: Refinement
- Process 4: Evaluation

- Process 5: Maintenance

The feasability study contains the problem identification, while the ontology kickoff includes the requirements specification, the analyzation of the input source and the development of the baseline taxonomy. Conceptualization and formalization as well as adding relations and axioms belong to process 3 refinement. In the evaluatin process, process 1 is reviewed. The last process deals with managing organizational maintenance process [67].

To build the project management ontology I use a mixture of Uschold and King and Grüninger and Fox. The reason is that they cover best the needs concerning the ontology engineering approach. While the engineering approach of Uschold and King [70] is more related to modeling and building up an ontology within a special domain, Grüninger and Fox [34] use a more formal and technical approach which is query centered. By merging both approaches, I obtain a detailed documentation of the ontology, its concepts, properties and relations, its term definitions and an illustration how to extend the ontology.

CHAPTER 3

Background Work

As a starting point for the work, based on the literature, I formulated the following statements outlining hypothetical reasons why projects tend to miss their goals.

- Most companies do not use a standardized archiving (system) to store information of finished projects.
- Work is done twice or more often, because
  - previously stored information cannot be found,
  - information is not stored correctly (e.g., no keywords or wrong identification) or
  - lack of knowledge of previous and similar work on other projects.
- Deadlines are missed and budgets are overrun because resource management, especially regarding responsibility assignments, is not used optimally.

To verify the statements mentioned above and to get a summary of current project management solutions, interviews with companies, working in different domains, were conducted. In addition, I performed an evaluation of current open source project management systems to identify common project management functionalities and to decide which system will be used for prototyping. In addition, an evaluation of Semantic Desktop systems was carried out.

3.1 Interviews

When starting with my work one of the first steps was to get practical information of current project management solutions; strengths as well as weaknesses. Possibilities to obtain this knowledge were

- to just base on literature and studies regarding failures in project management,

- to carry out a questionnaire accomplished through the internet or email or
- to do face-to-face interviews with project managers and members.

Possibility one, only literature and studies, did not make sense. I would have based my contributions, hypotheses and outcomes just on the work of others. A questionnaire through the internet or email was a consideration. However, the answers would have been short and maybe not meaningful. As a result, to get usable information about shortcomings, strengths and weaknesses as well as information regarding processes of current project management solutions, I decided to follow a qualitative approach [10] doing eleven face-to-face interviews with companies. In addition, the interviews were explorative as well as affirmative. Explorative, because I wanted to consolidate already known shortcomings of current project management. In addition, I tried to obtain "new" information (strengths, weaknesses and shortcomings) of PM as a base for further work. Affirmative due to the fact that the interviews were evaluative and to confirm my assumptions concerning current project management problems. When preparing the questionnaire I took some assumptions concerning PM problems and shortcomings based on the literature to identify important questions. During the interviews I was able to ask interposed questions to improve the quality of the results. While carrying out the interviews I tried to identify

- what software the companies use for project management,
- what functionalities such a software has,
- if the companies have standardized processes for doing their projects,
- if they archive information,
- how they store/archive information,
- if they reuse knowledge of finished projects and
- what the strengths and weaknesses of their current solutions are.

The questionnaire contains 27 questions (the questionnaire is shown in Appendix A). Relating to the expected results (given above) the most relevant questions are:

- What are the strengths of your current solution?
- What are the weaknesses of your current solution?
- Do you have any kind of archive for information relating to old/finished projects?
- How do you archive the information?
- Do you reuse archived information?

Since the research focused on IT projects, the majority of the interviewees were related to this domain. However, to get a broader view, refining the evaluation results I also invited interviewees of other domains. In addition, the size (number of employees) of the companies varies to consider different PM solutions within diverse enterprise sizes.

The basic groundwork of the questionnaire was a discussion within the working group of my university as well as meetings with a PM company, which sells a project management system focusing on PM handbooks. Based on this information, a first questionnaire was designed and tested with a business manager. Afterwards, the questionnaire was finalized.

Table 3.1 gives an overview of the companies interviewed including their sectors and sizes. As the interview results cover sensitive information, the company names have been anonymized.

Company	Number of employees	Sector
1	1000	Bank sector
2	35 (Austria)	Health sector
3	470	Public / IT sector
4	3000	IT sector
5	270	Bank / IT sector
6	800-900	Public / IT sector
7	50000	IT & services
8	200-300	IT & insurances
9	160 (Austria)	IT & services
10	3000-5000	Oil industry
11	125	IT / architecture

Table 3.1: Overview of the interviewed companies

Interview Results:

While I was carrying out the interviews, the contact persons were very honest and open in their answers; especially with respect to failed projects and the weaknesses in their current solutions. Certainly, they did not tell me the exact number or the perecentage of failed projects; nevertheless, they also did not whitewash their failures or failed projects. Three out of eleven companies told me that they have no statistics of their accomplished projects, whereas eight analyze their project results. For instance, concerning content, costs or controlling.

Table 3.2 gives an overview of the interview results and shows aggregated answers. Table 3.3 displays the answers of the strengths and weaknesses of the companies' current PM solutions. In contrast, table 3.4 highlights information about common archiving solutions and recovering information.

strengths	established standards and processes	simple and consistent approach	structured and clearly defined guidelines for the project structure
weaknesses	no interfaces between different systems	no “cockpit” for an overview of all projects	no central data storage
archiving	no central and consistent storage	electronic file systems, DVDs or hardcopies of documents	Wiki used for archiving

Table 3.2: Aggregated interview results

To summarize all the interview results, depicted in the tables 3.2, 3.3 and 3.4, the outcomes are as follows:

- **Strengths**
  The (established) standards and predefined processes were the most common statements, independent of the software used. The interviewees also mentioned their simple and user-friendly approach to the project management processes. One company stated that they only use Excel files for their project proposals, project execution and the closing reports. Another company has clearly defined standard processes for their projects, an application for the process control and predefined documents for different modules of the project process.

- **Weaknesses**
  The answers concerning the weaknesses of their used PM tool(s) were much more diverse. While some companies complained that they do not have an application programming interface (API) between their systems, others mentioned that they have no central data storage and/or bad project control. Others would require a complete solution or more defined project process standards. In addition, some would like to have a “cockpit” in the form of an overview of all current projects, their ongoing tasks and milestones. In this case, they would have the possibility of detecting problems related to time schedules, project costs or needs for additional staff for a specific project. In addition, nearly half of the interviewed companies do not have any standardized structured storage system for data about closed projects.

- **Archive**
  Half of the interviewed companies do not have a centrally accessible and structured storage system and therefore no search possibility. In addition, quite often information of finished projects is needed, e.g., customers need a documentation of finished projects. If companies store information, this knowledge is assigned to lessons learned or templates of documents.

Company	Strengths	Weaknesses
Company 1	• established standards and processes	• no integration between the systems • no „cockpit“ for the whole overview of projects • separate reporting
Company 2	• simple approach • easy to comprehend • needs nearly no disc space	• files are local • documents have to be sent by mail; different versioning
Company 3	• consistent approach	• costs and dates are not followed • shortage of resources
Company 4	• structured • clearly defined guidelines for the project structure • clearly defined guidelines for the documents • roles are defined • techniques are specified	• consistent project system would be highly desirable
Company 5	• central area • certified manager	• no interface between the systems of the company • no defined central file storage
Company 6	• cheap solution • integration Office products • includes relatively many functionalities	• no overall resource management possible
Company 7	• guaranteed consistency of processes & high internal publicity of the system • management tool is accessible to any employee	• resource management • tracking work orders
Company 8	• relatively high degree of freedom (if you can deal with it) • if there is a modification, the system is still flexible	• detailed information is missing • project leader would like to have access to the database to do their own reports (currently it is not possible)
Company 9	• consistent and documented project processes • project management tool box: document templates, tools, international background • open project management culture • meetings about project results	• in fact of international controlling additional work • no complete solution
Company 10	• clearly documented • all employees understand and accept the rules and systems	• quality of the documents • quality management (project plan should be the same as the end result) • priority of the different projects and tasks • resource management could be better
Company 11	• Each employee has the information he needs • Email reminder for important dates and documents	• Authorization structure and documents are growing • Database is too small • Print control

Table 3.3: Strengths and weaknesses of current project management solutions

Company	How do they archive	Archive information
Company 1	• documents are stored • electronic file system • documents on CD/DVD	• often too late awareness about already stored information • have no structured information
Company 2	• hardcopy of documents & electronic documents • no central storage • no consistent storage	• budget calculation • document structure for their events
Company 3	• central file storage SystemA	• products, which are developed, are used more often • “lessons learned” in meetings
Company 4	• no central file storage • no rules for saving project documents and information	• offers • model contract • templates for projects • “project jour fix” ➔ “lessons learned”
Company 5	• hardcopy, electronic file storage and database • no “real” archive system	• “lessons learned”
Company 6	• SystemB (long running storage)	• no structured information • no “lessons learned” • just the memory of the employees of previous projects • if it is needed stored information in the SystemB can be used
Company 7	• all information in SystemC ➔ experience knowledge is available, not only documents • project documents must be stored (hardcopy and electronic) • no system for archive storage	• workshops about project experiences • Wiki: experience insights (anonymous) • Forum with project management information
Company 8	• no rules for saving project documents • file server • MS SharePoint: documents, newsletter, …	• templates of documents • “Where is the required information stored?”
Company 9	• documents are stored centrally • if a project is finished ➔ information is archived	• strategic process model • team rooms, experience exchange • “lessons learned” • updates of document templates and processes • Wiki: information about project management and processes which are used in the company
Company 10	• archived in tools • no project can be deleted • collaboration tool e-room ➔ exchange of documents	• forum • international communication • e.g. project plans are reused
Company 11	• Database • USB flash drives	• Cost data & sample details • All documents related to a project (for at least 3 years)

Table 3.4: Archiving and reuse of information

In addition, I also asked the companies if they have statistics of successful and failed projects. Three said no, while the others replied with yes. They did not tell me how many of their projects exactly failed, but nearly all of them told me why. That is why I was able to identify the weaknesses of current project mangement solutions.

Another outcome of the interviews is that the most common software used is MS Project and MS Excel for project processing and SAP for the accounting. In general, the companies, which offer IT services and software utilize more complex PM systems than others.

The results of these interviews are quite important and interesting. While my assumptions were correct, the results highlight the importance of up-to-date and reliable information concerning current ongoing projects (e.g. time schedule, cost plans, open tasks) for the project manager and members.

In short, the key findings of the interviews are

- companies miss a central, well structured and accessible storage and archive,
- project leaders as well as project members need a listing of current ongoing projects with milestones and costs; in short, an up-to-date information cockpit with all relevant project items, and
- where the project management systems consist of several applications, interfaces between different applications are required to guarantee frictionless processes.

Almost 25 years ago Pinto and Slevin [55] identified ten factors, which have to be fulfilled for successful projects. These ten factors are:

1. *Project Mission - Initital clearly defined goals and general directions.*
2. *Top Management Support - Willingness of top management to provide the necessary resources and authority/power for project success.*
3. *Project Schedule/Plan - A detailed specification of the individual action steps of project implementation.*
4. *Client Consultation - Communication, consultation, and active listening to all impacted parties.*
5. *Personnel - Recruitment, selection, and training of the necessary personnel for the project team.*
6. *Technical Tasks - Availability of the required technology and expertise to accomplish the specific technical action steps.*
7. *Client Acceptance - The act of "selling" the final project to its ultimate intended users.*

8. *Monitoring and Feedback - Timely provision of comprehensive control information at each stage in the implementation process.*
9. *Communication - The provision of an appropriate network and necessary data to all key actors in the project implementation.*
10. *Trouble-shooting - Ability to handle unexpected crises and deviations from plan [54].*

Based on the interview results regarding the weaknesses of current project management I agree on these factors. Comparing the work of Pinto and Slevin [55] with the outcome of my work, the requirements regarding project management did not change during the last 25 years.

3.2 Semantic Desktop

When the idea emerged to improve current project management based on semantic technologies, I was thinking of interrelating a project management system and a Semantic Desktop. Thus, the system would be consisting of a client-server architecture and the client would be a fat-client, embedded into the Semantic Desktop system.

The **Semantic Desktop** can be described as follows: *"The use of ontologies, metadata annotations, and semantic web protocols on desktop computers will allow the integration of desktop applications and the web, enabling a much more focused and integrated personal information management as well as focused information distribution and collaboration on the Web beyond sending emails [66]".*

To choose the right one for the prototype I had a deeper look at the following Semantic Desktop systems:

- Gnowsis/Nepomuk [32]
- OpenIRIS [44]
- Haystack [38]
- DeepaMetha [18]
- DBin [16]
- chandler [12]

Gnowsis/Nepomuk:
In 2003 Leo Sauermann implemented Gnowsis, a Semantic Desktop, for his diploma thesis. From January 2006 to December 2008 Gnowsis was part of the NEPOMUK project (NEPOMUK = Networked Environment for Personal Ontology-based Management of Unified Knowledge). In 2009, Gnowsis changed from an open source system to a commercial product of Gnowsis.com. This is the reason, why the system will not be used for the prototype.

openIRIS:
In 2006 IRIS [44] started as open source software. One year later the first and latest version, version 0.4, was published. IRIS was originally implemented to serve as a user interface and knowledge base for the CALO project (Cognitive Assistant that Learns and Organizes) of SRI International. In August 2009 I sent an email to the contact email address of IRIS to get any information about the project, but I did not get any response. For this reason, and since the latest version is from 2006, the system will not be used for the prototype.

Haystack:
Haystack [38] is a semantic web browser. I tried to get information using the homepage, but some sites didn't work anymore. In addition, I also sent an email to contributors and innovators of Haystack and they sent me the latest version of Haystack. They also told me that Haystack is not under development anymore. As a consequence, the system will not be applied for prototyping.

DeepaMetha:
In 2000 Matthias Staps and Joerg Richter started with DeepaMehta [18]; an open source software. The latest version of DeepaMehta is 4.0 and was released in July 2011. At its homepage DeepaMetha is defined as "software platform for collaboration and knowledge management". Thus, it is not a Semantic Desktop in the way I am looking for. In addition, the system will not be used for the prototype.

DBin:
In 2005 a first draft of DBin was shown at the WWW2005 Developer Day. Since 2008 Version 2 of DBin [16] has been available. When I was trying to install and configure DBin I encountered errors and difficulties. To fix the problems I tried to use the documentation site of DBin, which did not work. As a consequence, I was not able to install and configure the system. In addition, DBin will not be applied for implementing the prototype.

Chandler:
The latest release of the system Chandler [12] is the version 1.0.3 from 2009. Chandler was designed for personal and small-group task management and as Calender. This system is not applicable for the prototype in fact it is already a quite complex system by itself. Chandler consists of a Chandler Desktop and a Chandler Server to enable collaborative work.

As a result of the evaluation of Semantic Desktop systems, I identified that most of the Semantic Desktop systems will not be developed further on or did not correspond to the needs of this thesis. The only system, Gnowsis, which is still under development and most likely fits to the requirements, is now implemented by the company Gnowsis.com. In addition, most of the implemented functionalities are now used as commercial adaptations. When I started with this work it was a predefined requirement that all software, used for this project, has to be open source.

Other systems, like Chandler or DeepaMetha, which are still under successful development are going straight forward to reach their aims. Even so, these goals touch lightly my proposal, they do not fit.

Beyond that the interview results highlight that a central knowledge base with direct access from any computer is preferred. Thus, I decided not to implement the idea of a fat-client-server solution and implement a server based project management system with access via web-browser. I will not use a Semantic Desktop for the prototype.

Although I decided not to use the Semantic Desktop for my work, it was important for me to give a short introduction to the domain of Semantic Desktop systems. Not least because these systems are quite similar to my idea of a PM knowledge base; they are just a personal information management system.

3.3 Project Management Systems

For the evaluation of the project management systems I defined a framework, containing functional and non-functional requirements as well as processes and simple use cases. It also included, e.g., several project related requirements such as the latest forum entries concerning the community behind the open source system. The evaluation framework contains different approaches based on the literature, such as [9] and [27], and requirements already mentioned.

The work of [9] describes a software system evaluation framework and its level of categorization as well as examples of project classifications. The work of [27] deals with a study to test a combination of methodologies for system evaluation.

Within the framework I

- compared the PM systems and
- decided which PM system will be used for further work.

For the evaluation I only considered open source PM tools. The reason for this lies in the strategy to improve current project management by extending an existing project management system with semantic technologies. As a result, the source code is needed and commercial ones does not offer the code. Nevertheless, commercial PM systems are quite similar to open source ones and provide nearly the same functionalities, like MS Project.

In addition, since I was using an open source project management system, other persons are able to improve and extend the work.

The approach to evaluate open source PM systems was as follows:

1. I performed an online validation to obtain a list of the most common and available open source project management systems.
2. Following this I chose the four systems
   - dotProject [24],
   - WebCollab [73],
   - Projectpier [56] and
   - PHProjekt [53].
3. Based on the interview results and existing PM evaluation frameworks, different requirements as well as simple use cases were defined.

I obtained these requirements by, firstly, working through the literature and comparing different evaluation processes [36, 63]; secondly, by analyzing the interview results to detect the basic requirements of a common used PM system, and thirdly, by defining my own requirements concerning the implementation of the prototype. Thus, for instance, the use cases reflect basic functionalities, which should be implemented within a project management system.

The requirements are separated into functional, non-functional, project management processes and use cases, which are related to the main functionalities.

In detail, the **non-functional requirements** contain the following issues:

- the software must be open source
- the software must be implemented as a web-based application
- the latest release of the system must have been from 2008 or newer (defined at the beginning of 2009)
- a forum, a mailing list and/or a wiki must exist
- in the forum or wiki the latest entry (from a user or administrator) should not be older than one week
- the software should be easy to install
- the software should be easy to use
- the software supports multiple languages

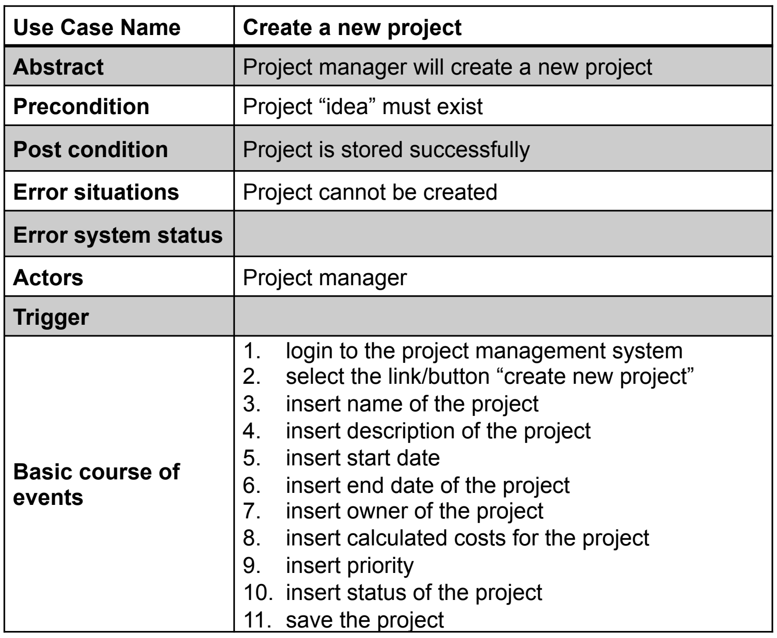
The programming language did not play a decisive role for the evaluation process. On the one hand most of the open source web based project management systems are written in PHP in any case. On the other hand the main focus was lying on the functionalities.

Functional requirements are:

- it should be possible to create and maintain the following information:
  - projects
  - contact information
    - companies
    - persons
  - tasks
  - costs
  - notes
  - time schedule
  - milestones
- other functionalities the system must provide are
  - document upload and download
  - search engine
  - user administration
  - group administration
  - role administration
  - export of information
  - e-mail
  - listing of
    - current projects
    - time schedule
    - tasks
  - management of several projects
  - management of human resources
  - assignment of one person to several projects/tasks

In the evaluation the **project management processes** cover the following processes:

- Planning
  to plan projects before starting
- Estimating
  to calculate the expected costs
- Scheduling
  to define dates and milestones within a project



Use Case Name	Create a new project
Abstract	Project manager will create a new project
Precondition	Project “idea” must exist
Post condition	Project is stored successfully
Error situations	Project cannot be created
Error system status	
Actors	Project manager
Trigger	
Basic course of events	1. login to the project management system 2. select the link/button “create new project” 3. insert name of the project 4. insert description of the project 5. insert start date 6. insert end date of the project 7. insert owner of the project 8. insert calculated costs for the project 9. insert priority 10. insert status of the project 11. save the project

Figure 3.1: Use Case 1: Create a new project

- Reporting
  to compile statistics or graphics

- Controlling
  up to date information of ongoing projects

The processes of the **use cases** are highlighted by the figures 3.1, 3.2 and 3.3. The use cases include the tasks “create a new project”, “a simple keyword search” and “insert a new task allocated to a project member”. I decided to take these basic use cases as I wanted to obtain a general overview of the features and functionalities of the evaluated systems. These requirements are used to evaluate the “powerful functional range” of the systems and to investigate the main focus of the evaluated applications.

The use cases are based on the interview results concerning functionalities of current PM solutions and the literature study. They just illustrate a combination of functionalities, which should be covered by the project management system at least. In addition, they outline the basic functions of the scenarios, illustrated in chapter 4 “Scenarios” and implemented by the prototype.

Evaluation Results:

Table 3.5 highlights the comparison of the non-functional requirements. Only Projectpier is available with a release version below one. Normally you should use a software version above

Use Case Name	**Create a new task and allocate it to a project member**
Abstract	A new task is created including a task owner
Precondition	Project and project member must already exist
Post condition	A task is created and is allocated to a project member
Error situations	Task cannot be created.
Error system status	
Actors	Project manager
Trigger	
Basic course of events	1. login to the project management system 2. select a project 3. select the button/link “create new task” 4. insert name of the task 5. insert description of the task 6. insert start date 7. insert end date of the task 8. insert owner of the task 9. insert calculated costs for the task 10. insert priority 11. insert status of the task 12. select if the task is a milestone 13. define dependencies to other tasks 14. define project members who are assigned to this task 15. save the task

Figure 3.2: Use Case 2: Create a new task and allocate it to a project member

Use Case Name	**Simple keyword search**
Abstract	it will be searched for information
Precondition	Information/Knowledge is stored in the project management system
Post condition	Information is retrieved
Error situations	Search functionality doesn’t work or the information can’t be retrieved
Error system status	
Actors	Project member
Trigger	
Basic course of events	1. login to the project management system 2. search link/button is pressed 3. keyword is entered 4. search button is pressed 5. search results are listed

Figure 3.3: Use Case 3: Simple keyword search

one; as a result the error frequency should be less and the software should be remain stable. In general, all PM systems are open source and include an active forum. In addition, all systems are easy to install and are available as multi language systems, except Projectpier.

PM Systems	dotProject	PHProjekt	WebCollab	Projectpier
Latest release	29.07.2008	23.01.2008	16.04.2009	10.06.2009
Latest release version	2.1.2	5.3 beta	2.50	0.8.5.0 beta
Forum (website)	X	X	X	X
Latest forum entry	29.07.2009	29.07.2009	08.07.2009	28.07.2009
Open source	X	X	X	X
Easy to install	X	X	X	X
Multi-languages	X(modules)	X	X	no

Table 3.5: Overview of the comparison of the non-functional requirements (Last access to the websites on Wednesday 29th of July 2009)

By contrast, table 3.6 displays a comparison of the main functionalities of the evaluated PM systems. In Table 3.6, the terms “projects”, “contacts”, “tasks”, “costs”, “milestones”, “users”, “groups” cover both insertion and editing. While all project management systems cover insert and editing projects, contacts and tasks, only dotProject and PHProjekt include costs. Also milestones are only contained by dotProject and Projectpier. The only functionality, which is not covered by dotProject, is the possibility to edit user groups. Nevertheless, dotProject covers the main and most important functionalities. PHProjekt is also quite interesting, but does not include the function milestone, which is important for the ontology as well as prototype. Thus, comparing the functionalities of the open source systems, dotProject seems the most suitable one.

Table 3.7 outlines “nice-to-have” functionalities, which would be nice to have but are not necessary. I decided what functionalities are main and what are “nice-to-have” ones related to the requirements of the prototype. “Nice-to-have” functionalities are not required for implementing the prototype. dotProject inlcudes a forum, a trouble ticket and links in its software. In addition, it enables archiving projects. This feature is quite important, because several interview partners mentioned that they do not have a standardized data storage. Furthermore, the scenarios (section 4) cover the reuse of archived knowledge. By contrast, PHProjekt contains a forum and chat and enables the import and export of a calender and of entries. On the contrary, WebCollab also inlcudes a forum and facilitates to archive and clone a project. ProjectPier contains an iCalender, tags and RSS feeds. When comparing the “nice-to-have” functionalities of the PM systems, implying the result of table 3.6, dotProject is again the most suitable software. Reason is the possibility to archive projects.

Table 3.8 highlights the comparison of the project management processes. WebCollab and

PM Systems	dotProject	PHProjekt	WebCollab	Projectpier
Projects	X	X	X	X
Contacts	X	X	X	X
Tasks	X	X	X	X
Costs	X	X		
Milestones	X			X
Document upload	X	X	X	X
Document download	X	X	X	X
Users	X	X	X	X
Groups		X	X	
Mails	X	X	X	X
Search	X	X	X (forum)	X
Listing of current projects	X	X	X	X
Listing of tasks/activities	X	X	X	X
Multiple projects	X	X	X	X

Table 3.6: Overview of the comparison of the functional requirements

PM Systems	dotProject	PHProjekt	WebCollab	Projectpier
Forum	X	X	X	
Trouble ticket	X			
Import/export calendar		X		
Chat		X		
Export entries		X		
Archive project	X		X	
Clone project			X	
iCalendar				X
Tags				X
RSS Feeds				X
Links	X			

Table 3.7: Overview of the comparison of further functional requirements

Projectpier do not provide any processes, while PHProjekt covers the scheduling and controlling process. dotProject contains planning, scheduling, reporting and controlling. Reporting is a quite interesting and important process; in fact, statistics of successful and failed projects are possible or just an overview of a current project status. Likewise the other tables, dotProject covers most of the processes and thus, is the most suitable system for the prototype.

	dotProject	PHProjekt	WebCollab	Projectpier
Planning	X	-	-	-
Estimating	-	-	-	-
Scheduling	X	X	-	-
Reporting	X	-	-	-
Controlling	X	X	-	-

Table 3.8: Comparison of the project management processes

While in Table 3.6 the functionalities of the evaluated systems look quite similar, the results of the use cases display differences. Table 3.9 and table 3.10 demonstrate the results of use case 1 and 2 and whether the systems provide the required functionalities, clearly highlighting the differences between the systems. The comparison of the use cases outlines the differences regarding the features of the systems.

Regarding use case 1, while dotProject and PHProjekt contain several input and information fields, WebCollab and Projectpier include just the most important ones, such as project name. For instance, to create a new project within the PM system Projectpier, the user just has to insert the project name, a description of the project, the team members of the project and to decide whether the members receive emails. WebCollab has already some more input fields. There the user has to insert the project name, the project owner, the finish date of the project, the priority, the status of the project (started, finished, . . .), a description, to define the group of users, who are responsible for this project, and the security rules and to decide whether the group members receive emails. By contrast, dotProject and PHProjekt enable much more input fields. For instance, PHProjekt has several input fields regarding the costs of the project. Whereas dotProject deals with the project information in general (project type, budget, . . .).

Concerning the results of use case 2, Projectpier cannot be compared with the others. The process as well as the input fields are totally different. In addition, it has no other input fields as the most required ones. Whereas the other systems are more or less equal. dotProject as well as WebCollab enable an assignment of an employee to a task; by contrast, PHProjekt does not and thus, it is ruled out. Comparing just dotProject and WebCollab, dotProject contains the function to define a task parent of the task. This relation between tasks will also be depicted in the project management ontology; thus, dotProject seems the most suitable software again.

Use case 3 is not compared within a table, because the search functionalities are quite different. For instance, WebCollab only offers a forum search but no search according to project relevant information items, while Projectpier does not have any kind of search functionality. In contrast, dotProject and PHProjekt provide both simple and advanced search functionalities. In both systems, the search results are separated into information items, such as documents or tasks.

	dotProject	PHProjekt	WebCollab	Projectpier
task list name	-	-	-	X
task list description	-	-	-	X
task list milestone	-	-	-	X
private task lists	-	-	-	X
task name	X	X	X	X
task assign to	-	-	-	X
status	X	X	X	-
progress	X	-	-	-
priority	X	X	X	-
milestone	X	-	-	-
task owner	X	-	X	-
task type	X	-	-	-
access	X	-	X	-
web address	X	-	-	-
task parent	X	-	-	-
target budget	X	-	-	-
description	X	X	X	-
end date	-	X	X	-
emails	-	-	X	-
security rules	-	-	X	-
to	-	X	-	-
start date	-	X	-	-
contact	-	X	-	-
project	-	X	-	-
notification	-	X	-	-
project related times	-	X	-	-
opening	-	x	-	-

Table 3.9: Comparison of Use Case 1: Create a new project

	dotProject	PHProjekt	WebCollab	Projectpier
project name	X	X	X	X
project owner	X	X	X	-
company	X	-	-	-
internal division	X	-	-	-
start date	X	X	-	-
target finish date	X	X	X	-
target budget	X	X	-	-
priority	X	X	X	-
short name	X	-	-	-
color identifier	X	-	-	-
project type	X	-	-	-
status	X	X	X	-
actual budget	X	-	-	-
URL	X	-	-	-
staging URL	X	-	-	-
import tasks from	X	-	-	-
description	X	X	X	X
hourly rate	-	X	-	-
category	-	X	-	-
cost centre	-	X	-	-
cost unit	-	X	-	-
contractor	-	X	-	-
subproject of	-	X	-	-
project related times (description, date, hours)	-	X	-	-
opening	-	X	-	-
members	-	X	-	X
contacts	-	X	-	-
group of users	-	-	X	-
send emails	X	X	X	X
security rules	-	-	X	-

Table 3.10: Comparison of Use Case 2: Create a new task and allocate it to a project member

Regarding the prototype, the comparison of the different project management systems has illustrated that dotProject seems to be the most suitable one. It has similar functionalities to PHProjekt, but is more usable. Furthermore, it contains more features than WebCollab and Projectpier, especially in the field of project processes. In addition, dotProject already tackles some functionalities, which are part of the project management ontology as well as prototype. Thus, I decided to use dotProject.

dotProject in Detail

In 2000 dotProject was originally named dotmarketing.org and in 2001 the project was moved to Sourceforge. dotProject is defined as *'a web-based project management application, designed to provide project layout and control functions [25]'*. The latest release, published in January 2011, is the version 2.1.5.

Prerequisites of dotProject are the web server Apache with integrated PHP and MySQL support; the PHP version 5.0.X or higher, at least the MySQL version 4.1 and Apache version 1.3.27 or higher. Apache2 is only recommended starting with the dotProject version 2.0.40 combined with PHP version 4.3.5 or later.

dotProject is divided into *core modules* and *addon modules*. While core modules represent the system and standard functionalities, addon modules are not supported by the original development team of dotProject. They are offered by other developers.

The core modules are the following ones:

- Backup Module
  This module enables a backup of the dotProject installation.
- Calender
  Displays events, important dates of tasks, etc.
- Companies, Departments, Contacts and User
  Companies are the central part of dotProject. A company is able to have as many relations to contacts, users or projects. While a user does not need any relation to a company, contacts and projects must have a connection.
  A user is someone with a login account to dotProject. A contact does not.
- File Module
  Lists all uploaded files within dotProject.
- Forums
  The forum is a simple communication module.
- Gantt charts
  It is a graphical representation of projects and their tasks, showing the time line and assigned resources.

- History Module
  This module keeps track of added, modified and deleted tasks, projects, files as well as user logon/off etc.
- Links Module
  It is a central repository for storing references and web links.
- Project Reports
  This module outlines different information of projects, such as tasks sorted by users or project statistics.
- Projects
  This module handles the project activities, like editing, deleting or listing.
- Resources Module
  It is used to connect non-human resources to a project/task.
- Smartsearch Module
  Smartsearch provides functionalities for searching for words or phrases across an entire dotProject database and if you have parsers available, it can locate strings within attached files.
- Tasks
  A task must be related to a project.

CHAPTER 4

Scenarios

The scenarios highlight important aspects of this thesis and “translate” the research question into “applicable” and “tangible” examples. They reflect common PM problems identified by the interviews and outline the requirements of the ontology as well as the prototype. Beside the interview results basis for the scenarios are also the use cases, which were part of the evaluation process of the open source PM systems. In addition, they are used to evaluate the research question and contributions by serving as base of the evaluation guidelines.

In the following section I describe three scenarios which cover the most common problems, identified in the interviews [19], of current project management solutions [20]. They demonstrate the functionalities of a possible PM system and outline project processes, which are not covered by current project management solutions. The scenarios focus on problems like central and standardized data storage as well as the search functionality to recover already stored information, independent of the type of information, e.g., documents, competences of employees or lessons learned.

Figure 4.1 outlines the most important functionalities, their dependencies and correlations regarding the scenarios. Thus, the basis of the pyramid is the “Central Data Storage” to enable the ’universal’ access to the information and to guarantee the consistency and availability of the different information items. The “Central Data Storage” contains the storage and the access of the information. The second layer contains the “Standardized Data Storage”. This layer is closely intertwined with the first one and is geared towards standard processes (e.g., how to store information correctly) for storing all information regarding project management. “Retrieve Information” deals with information retrieval. The “Cockpit” mainly displays the information of all ongoing projects, such as their status and information of project changes, problems or delays. Implementing those four layers the prototype is becoming a knowledge base including a standardized data storage, related information, retrieval functions and with a cockpit a harmonised view to different projects.

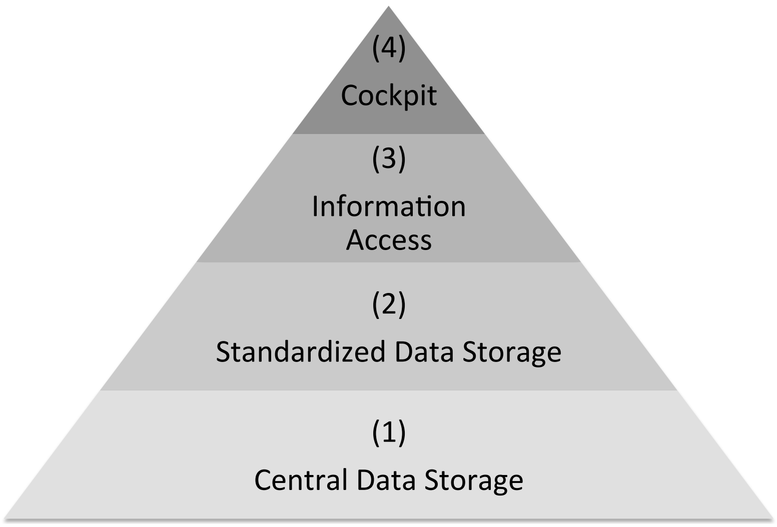


Figure 4.1: Problem Pyramid of missing functionalities improving the project management quality [20]

The focus of the three scenarios is "how do I have to store and collect the information" (layer one and two) and "how do I retrieve the information for reusing it" (information access, layer three and four).

When starting with the work of the scenarios my first version of them were quite general. After two presentations, one in my working group and one at the Karlsruhe Institute of Technology, and some discussions within the groups I reworked my scenarios. The focus was to define concrete and realistic scenarios based on the interview outcome and not only based on functions.

Below, the scenarios are discussed as follows:

1. To cover the main ideas behind the scenarios they are described as use cases.

2. Weaknesses of PM solutions, which are handled by the individual scenarios, are listed.

3. Questions, which are derived from the scenarios, are specified. These questions will be answered/covered by querying and selecting the project management ontology.
   The required concepts, properties and relations (within the PM ontology) to accomplish these scenarios are illustrated in the chapter 5 "Ontologies". To answer the informal questions based on the scenarios, some SPARQL queries are shown in section 7 "Evaluation".

4. Additional comments are given.

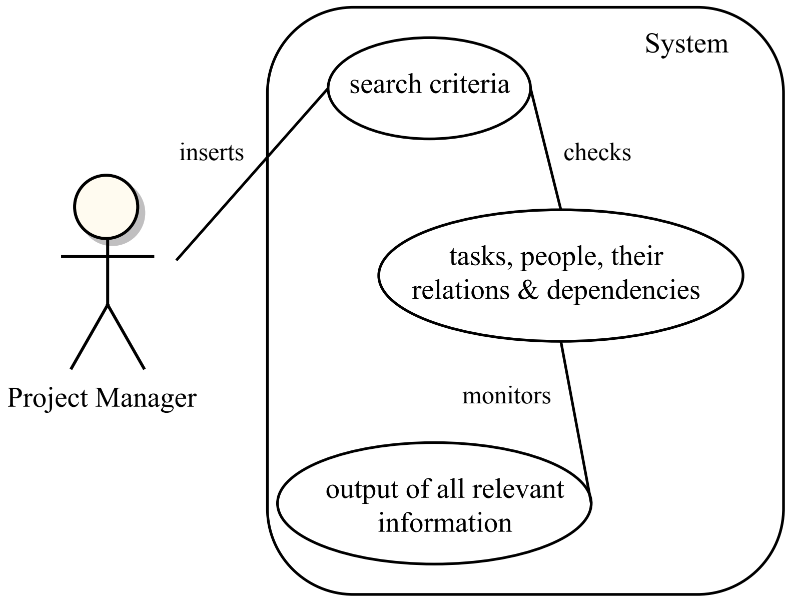

System
search criteria
inserts
checks
tasks, people, their relations & dependencies
Project Manager
monitors
output of all relevant information

Figure 4.2: Operating procedure of the scenario "Bob needs Holidays"

4.1 Scenario 1 - "Bob needs Holidays"

Description: Bob is working as a software developer in the IT company "Holidays". He is currently working for the project "Example" where he is responsible for work package (WP) 4. The actual date is March, the 14th and Bob is thinking if he could go on holidays in August for two weeks. But in August the deliverable 4.3 of WP 4 has to be finished. So, the question is: Is he able to go on vacation in August when he should work to finish the deliverable?

The technical aspect of this scenario is as follows: The project leader inserts the scheduled period of holidays and the name of the employee into the project management system. The system checks the assigned work packages, deliverables, tasks and projects of the employee. Furthermore, it monitors the process status of the work as well as possible milestones, dependencies and relations between different tasks. In addition, the PM system checks employees, who are able to do the tasks of the person. Preconditions are that these employees are not on holidays and have the same or similar competences as the colleague. Furthermore, they must be available during the holiday period of the person. Figure 4.2 displays the operating procedure of this scenario.

Weaknesses: The weaknesses, covered by this scenario, are:

- Detailed information of relationships between projects/tasks, time-management, staff members and competences is missing.
- Current PM systems do not provide up-to-date information; a kind of "cockpit" which

offers all relevant information regarding ongoing projects (e.g.: are the projects in time, is a task not finished in time?) is missing.

Questions: The following items are the results of the first scenario and must be answered to tackle the question whether the employee is able to go on vacation in August?

- What is the process status of the deliverable?
- What is the effective and planned end date of this deliverable?
- Does the deliverable have any dependencies to other deliverables, tasks or work packages?
- Is the deliverable of Bob a milestone?
- Does the company have employees with the same competences as Bob? In which projects do they work? Do they have deliverables and/or tasks during the holidays of Bob?
- Is Bob responsible for any other deliverables, work packages or tasks? What are the effective and planned end dates of these deliverables, work packages and tasks? Do they have dependencies to others?

Additional Comments: Scenario 1 covers the layer "Information Recovery Access" and "Cockpit". "Information Recovery Access", because information about ongoing projects, their tasks, employees and their competences must be searched and recovered. Reasons for the layer "Cockpit" are browsing and screening through the information to find suitable results. The scenario described above can also be applied, for instance, in the field of sick certificates. In this sense the scenario becomes even more relevant, because if an employee gets ill there is often no time to delay appointments. Thus, a colleague with the same competences must be found. In fact, a deliverable or WP must be finished in time. Another domain for this scenario can be employee turnovers. In this connection, the time period to react is shorter than in the case of holiday substitution.

4.2 Scenario 2 - "A new Project"

Description: Frida is the project leader of the new software project "Easy Life" within the IT company "Parallels". She inserts the new project into the project management system. Frida provides all relevant project information, such as name of the project, project description and used technologies. While storing the data all the supplied information is checked against similarities to existing project knowledge. The system seeks similarities between, e.g., task descriptions, task names, lessons learned and competences. If similarities to stored projects are found, the information of, e.g., lessons learned or tasks, will be displayed as advice. Now Frida, the project leader, has the choice to decide whether relevant information is contained in the advice or not. If there is relevant information she connects the new project part to the information advice of the stored project knowledge. This check can also be repeated later on. With this check information loss can be avoided. The operating procedure is depicted in figure 4.3.

Weaknesses: The weaknesses resolved by this scenario can be described as:

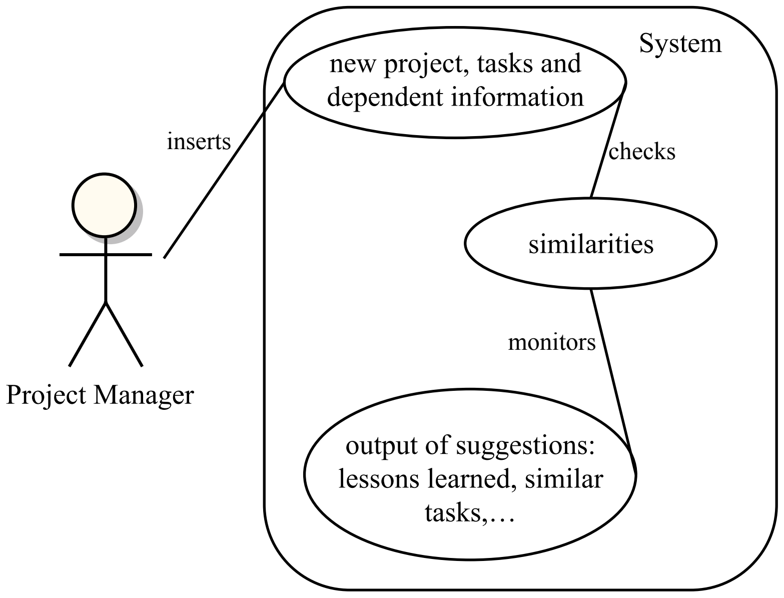

System
new project, tasks and dependent information
inserts
checks
similarities
monitors
Project Manager
output of suggestions: lessons learned, similar tasks,…

Figure 4.3: Operating procedure of the scenario “A new Project”

- Search through archived information is difficult (different storage, no predefined and consistent annotation, no existing relations between different information items)
- The project management life cycle consists of the phases: initiating, ongoing and closing. While the ongoing (execution and monitoring & controlling) part is more or less covered by existing project management systems, the initiating and closing phases are still badly covered.

Questions: Following questions regarding scenario 2 are relevant:

- Are there similar projects/tasks to the project “Easy Life” available? (e.g., name, descriptions, software in use)
- Are there already lessons learned available which can be used for this project?
- Can already stored information be reused for this project? (e.g., software modules, documentation, know how)

Additional Comments: The layer “Central Data Storage” and “Standardized Data Storage” are included in scenario 2. However, layer three “Recover Information” and four “Cockpit” are also part of this scenario. While layer one and two deal with the data storage, layer three and four cover the retrieval of already stored information.

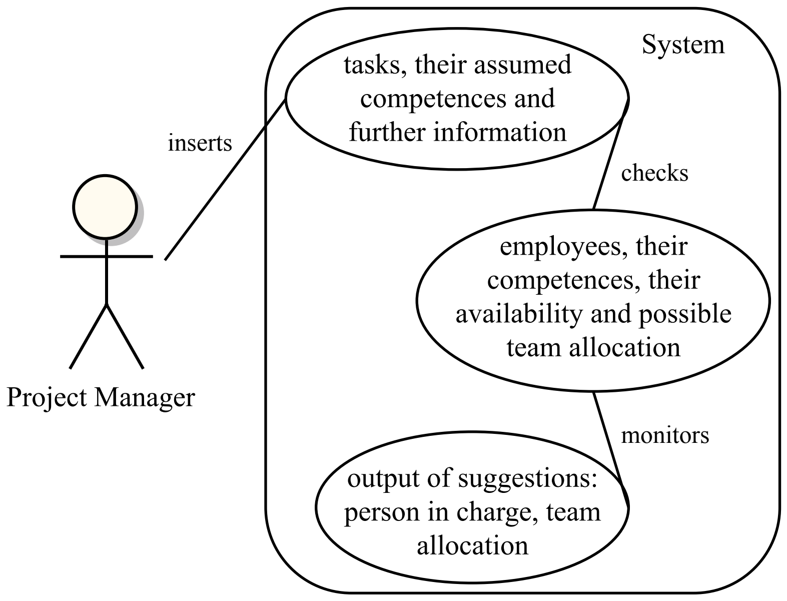

System
tasks, their assumed competences and further information
inserts
checks
employees, their competences, their availability and possible team allocation
Project Manager
monitors
output of suggestions: person in charge, team allocation

Figure 4.4: Operating procedure of the scenario "Team suggestions"

4.3 Scenario 3 - "Team suggestions"

Description: Frida, the project leader of the project "Easy Life", enters the PM system with information regarding work packages, deliverables and tasks. For setting up the optimal team, the system checks all employees, their capacity for teamwork as well as their competences and reports recommendations regarding the project allocation. Furthermore, the system monitors if employees, who might be considered for this project, are already assigned to other projects and tasks. If employees are already completely allocated to another project or task, they will not be considered anymore. If employees are not working with full capacity and their competences fit to the necessary project, they will be listed as available. At the team allocation of the system all employees are assigned to their possible functions in the new project. Furthermore, it is marked if employees collaborate well with each other or if they have some problems while working together.

If Frida does not agree with the recommendation of the system, she is able to search for other proposals regarding the project allocation. For instance, Frida wants someone else as project manager. Figure 4.4 displays the operating procedure.

Weaknesses: The weaknesses, highlighted by this scenario, are:

- Detailed information of relationships between tasks, time-management, staff members and competences is often missing.
- Search through archived information is difficult (different storage, no standardized and

consistent annotation, no existing relations between different information items).

- The project management life cycle consists of the phases: initiating, ongoing (including planning, execution, monitoring & controlling) and closing. While the ongoing phase is more or less covered by existing project management systems, the initiating and closing phases are still badly covered.

Questions: Questions, which can be deduced from the scenario 3, are:

- Which employees have the most adequate competences for the project/tasks?
- Do the employees already work in other projects?
  In which time period do the personnel work in other projects/tasks?
  In which projects and tasks are the employees already working?
  Are they responsible for tasks or projects?
  What are they working in the projects or tasks?
- Do the chosen employees work well with others? What are their competences?
- Do the competences match with competences needed in the new project?

Additional Comments: Scenario 3 deals with all layers depicted in figure 3. To retrieve already stored information, the annotation of information plays a decisive role. In addition, the relation between different information items is also an important part. By means of ontologies the relations can be easily set and the search queries can be optimized. Thereby, the search queries can be more complex and interlaced.

For further work, scenario 2 and 3 are merged to one, because they are intertwined. As soon as the project leader inserts information of a new project into the PM system, he/she wants to know the employees, who are free during the project, have the required competences and if possible work well together.

CHAPTER 5

Ontology

In this section I cover both, the engineering approach of the ontology and the resulting ontology.

5.1 Ontology Engineering

When carrying out a literature study (shown in chapter 2 “Related Work”) Gomez-Perez et al. [33] give an excellent overview of methods for building ontologies. The most important ones, which influenced my approach, are Uschold and King [70] and Grüninger and Fox [34].

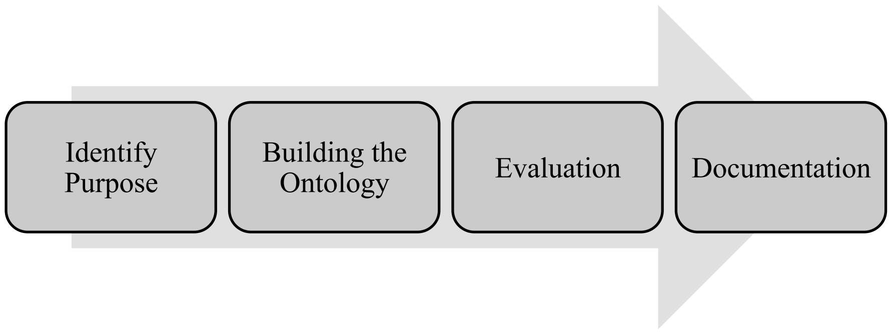


Figure 5.1: Ontological Engineering Approach of Uschold and King [70]

Uschold and King defined the following four steps for ontological engineering [70] (see figure 5.1):

1. **Identify purpose**
   To identify the purpose and the scope. Why is the ontology being built? What is its usage?

2. **Building the Ontology**
   This step is seperated into three steps:

a) *Ontology capture:* To identify key concepts and properties and to specify their terms.

b) *Coding:* To define classes, entities and relations as well as to implement the ontology.

c) *Integrating existing ontologies:* To check whether existing ontologies can be used.

3. **Evaluation**
   To validate the consistency of the ontology code.

4. **Documentation**
   Documentation of the ontology; its concepts, properties and relations as well as its handling.

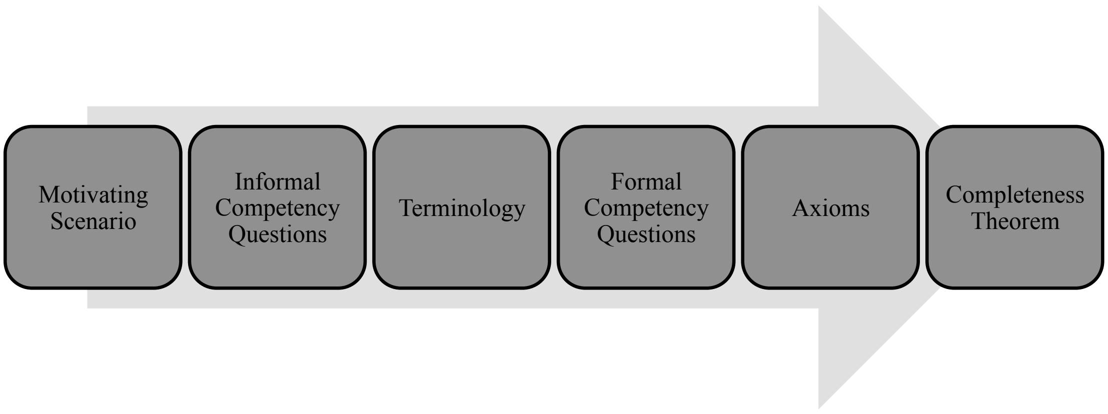


Figure 5.2: Ontological Engineering Approach of Grüninger and Fox [34]

By contrast, Grüninger and Fox described six steps to implement an ontology [34] (shown in figure 5.2):

1. **Motivating scenario**
   "The development of ontologies is motivated by scenarios related to the applications that will use the ontology. Suche scenarios describe a set of the ontology's requirements that the ontology should satisfy after being formally implemented [34]."

2. **Informal competency questions**
   Each scenario elicits questions; informal competency questions. These questions, written in natural language, have to be answered by the ontology.

3. **Terminology**
   From the informal competency questions the terminology of the ontology will be deduced. This terminology will be formally represented by means of concepts, attributes and relations in a first-order logic language. First, the objects should be identified; second, the predicates.

4. **Formal competency questions**
   Formal competency questions will be written in first-order logic. These questions are deduced from the informal competency ones.

5. **Axioms**
   Axioms are used to define the terms of the ontology.

6. **Completeness theorem**
   It defines the conditions under which the solutions to the questions are complete.

For the implementation of the ontology, I used a mix of both approaches. While the engineering approach of Uschold and King [70] is more related to modeling and building up an ontology within a special domain, Grüninger and Fox [34] use a more formal and technical approach which is query centered. By merging both approaches, I obtain a detailed documentation of the ontology, its concepts, properties and relations, its term definitions and an illustration how to extend the ontology. In addition, scenarios are used as ontology requirements (described in Chapter 4 “Scenarios”). Beyond that informal and formal competency questions specify the ontology in more detail and simplify the implementation of the prototype.

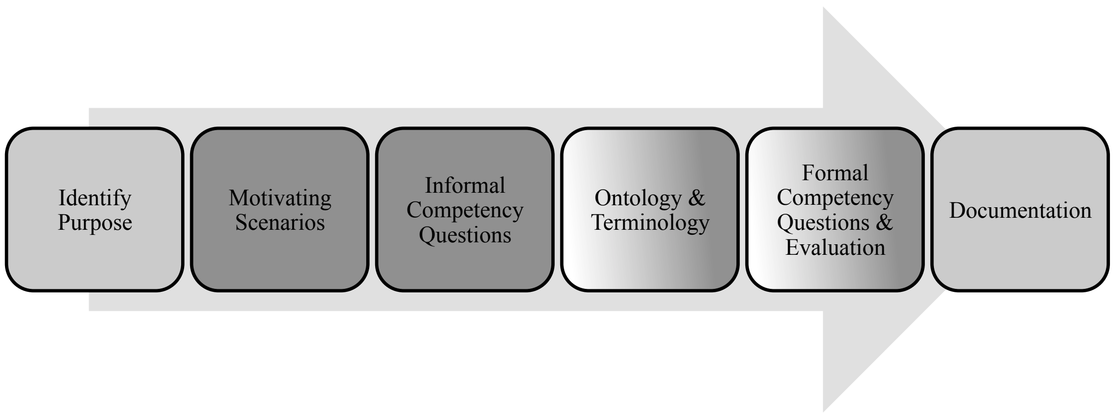


Figure 5.3: Merged Ontological Engineering Approach of my work [21]

Depicted in figure 5.3, my engineering approach contains the following steps in detail [21]:

1. **Identify purpose:**
   The ontology covers current project management problems; e.g., relations between tasks, required competences by the tasks, project members and their competences as well as lessons learned

2. **Motivating scenarios:**
   I identified three scenarios, which deal with current weaknesses of project management

solutions. They describe both, the requirements of the ontology as well as possible solution processes. In addition, they are used for the evaluation of the prototype and the ontology.

3. **Informal competency questions:**
   These questions, derived from the scenarios, are natural language questions. They will be answered by querying the formal questions testing the prototype.

4. **Building the ontology & terminology:**
   Based on the scenario requirements and the informal competency questions I identified the ontology's concepts, properties and relations and their term definitions.

5. **Formal competency questions & Evaluation:**
   To write the competency questions in a formal way, I used SPARQL (SPARQL Protocol And RDF Query Language), a RDF (Resource Description Framework) query language. In addition, to evaluate the ontology I simulate the scenarios by using the prototype and querying the questions.

   To evaluate the ontology I reviewed the consistency by using the consistency check of the used ontology editor TopBraid, the Website consVISor (a tool for checking the consistency of OWL ontologies) and the W3C RDF Validation Service [72]. I carried out all three to ensure the ontology has no errors. For instance, the outcome of the W3C RDF Validation was: "Your RDF document validated successfully".
   The main evaluation will be accomplished by testing the ontology and prototype with the predefined scenarios.

6. **Documentation:**
   To keep track of the ontology, I documented the concepts, properties and relations and their term definitions. In addition, the modularity of the ontology and the interfaces are described. Thus, the ontology is easily extendable and can be used for any other project management system in any other domain.

For engineering the ontology I used the ontology editor TopBraid Composer [69]. Reason for this decision was an internal discussion of available ontology editors and which one should be used within the working group. Therefore, an evaluation of existing ones was carried out. As a result, the working group chose the TopBraid Composer.

When developing the project management ontology I had to decide whether to use OWL (Web Ontology Language) or RDF (Resource Description Framework); and I decided to implement the ontology in OWL; more precisely, OWL Lite. This is because OWL offers more complex functionalities for describing relationships than RDF. Furthermore, OWL has more possibilities to define properties more exactly.

OWL provides three sublanguages: OWL Lite, Description Logic (DL) and Full.

- OWL Lite
  "OWL Lite supports those users primarily needing a classification hierarchy and simple constraints. For example, while it supports cardinality constraints, it only permits cardinality values of 0 or 1. It should be simpler to provide tool support for OWL Lite than its more expressive relatives, and OWL Lite provides a quick migration path for thesauri and other taxonomies. Owl Lite also has a lower formal complexity [52]".

- OWL DL *"OWL DL supports those users who want the maximum expressiveness while retaining computational completeness (all conclusions are guaranteed to be computable) and decidability (all computations will finish in finite time). OWL DL includes all OWL language constructs, but they can be used only under certain restrictions (for example, while a class may be a subclass of many classes, a class cannot be an instance of another class). OWL DL is so named due to its correspondence with description logics, a field of research that has studied the logics that form the formal foundation of OWL [52]".*

- OWL Full *"OWL Full is meant for users who want maximum expressiveness and the syntactic freedom of RDF with no computational guarantees. For example, in OWL Full a class can be treated simultaneously as a collection of individuals and as an individual in its own right. OWL Full allows an ontology to augment the meaning of the pre-defined (RDF or OWL) vocabulary. It is unlikely that any reasoning software will be able to support complete reasoning for every feature of OWL Full [52]".*

"Each of these sublanguages is an extension of its simpler predecessor, both in what can be legally expressed and in what can be validly concluded. The following set of relations hold. Their inverses do not [52]."

- "Every legal OWL Lite ontology is a legal OWL DL ontology.
- Every legal OWL DL ontology is a legal OWL Full ontology.
- Every valid OWL Lite conclusion is a valid OWL DL conclusion.
- Every valid OWL DL conclusion is a valid OWL Full conclusion [52]."

Nevertheless, for the needs of the PM ontology the features of OWL Lite are sufficient; although it has a lower formal complexity than OWL DL. For instance, OWL Lite includes

- property characteristics as object, datatype and inverseOf,
- RDF Schema features, like domain, range and subClassOf or
- datatypes, which is sufficient.

In short, I have chosen OWL Lite, because it contains all features (relations, reasoning) which I required to implement the project management ontology.

5.2 Concepts and their Properties

In the following the concepts and their properties are explained in detail. Then, the relations between the concepts will be described.

Concepts or classes describe concepts in the domain of an ontology. A concept is able to have subclasses, which are more specific than the superconcept. For instance the class "Cats" represents all cats. If I divide the class "Cat" into the subclasses "long hair" and "short hair", the concepts are getting more specific.
Properties describe attributes and features of a concept. For instance the "race" or age of a cat [50]. An object property is defined as a property for which the value is an individual. Whereas the value of a datatype property is a data literal.

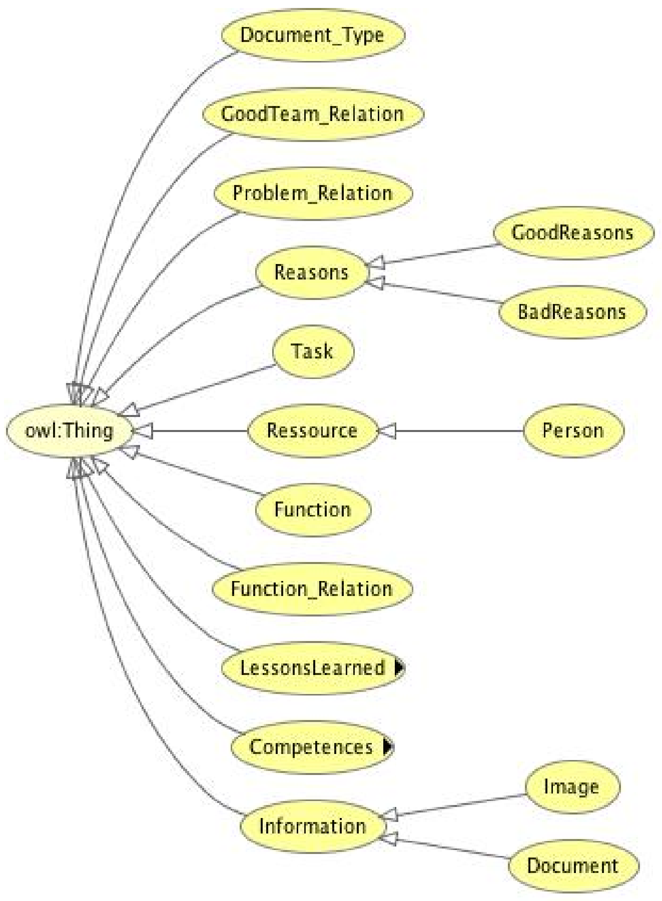


Figure 5.4: All concepts of the project management ontology

Figure 5.4, 5.5 and 5.6 highlight all concepts, main- as well as subconcepts, of the project management ontology.

Figure 5.4 outlines the first level of the project management ontology and the subconcepts of the concepts *Reasons*, *Information* and *Ressource*.

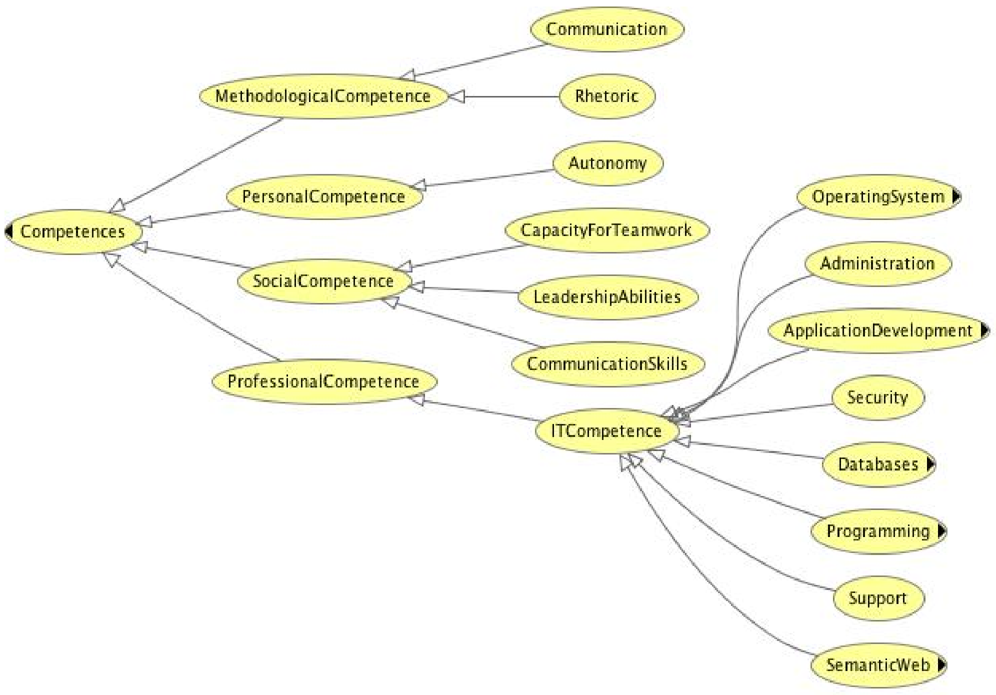


Figure 5.5: All subconcepts of the concept *Competences*

Figure 5.5 illustrates the subconcepts of the concept *Competences*. The second level of the concept *Competences* classifies the kind of competence/skill. The third and fourth level displays categorization examples of potential competences. I did not represent the subconcepts of the fifth level of the classifications of *IT Competence* as a figure due to the fact that it is just another categorisation of skill examples in the domain of IT (but they are listed later on as figure).

The subconcepts of the concept *LessonsLearned* are also a possible classification and can be easily extended; shown in figure 5.6.

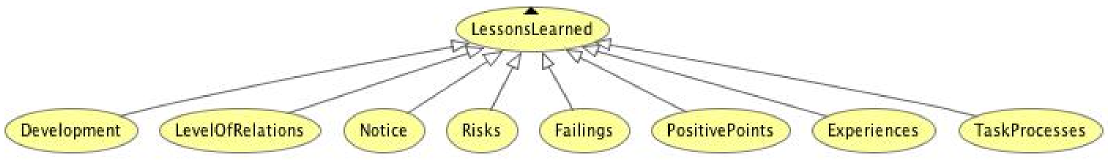


Figure 5.6: All subconcepts of the concept *LessonsLearned*

Task

A task can contain several sub-tasks. The task, which is no sub-task of another task is identified as project; while sub-tasks can be termed as work packages, deliverables or tasks (these are common categories to classify a project). A project is able to consist of several tasks. Each task is related to a person, who is responsible for the task. In addition, the task is related to at least one function, which describes the activities within that task. To each function at least one person is related. Thus, the relation between task, function and person is handled (It will be described in more detail in section 5.3. Relations).

Properties of the concept *Task* (depicted in table 5.1)

Task	
assumes	*Competences*
hasFunction	*Function_Relation*
hasLessonsLearned	*LessonsLearned*
hasPersonInCharge	*Person*
hasSubTask	*Task*
includes	*Information*
isPredeccesorOf	*Task*
isSuccessorOf	*Task*
hasBeginDate	date
hasDescription	string
hasEndDate	date
hasFinishDate	date
hasName	string
hasStartDate	date
hasStatus	string
isMilestone	boolean
isProject	boolean

Table 5.1: Properties of the concept Task (the properties in italic are object properties, which refer to other concepts, whereas the underlined are datatype properties)

Object Properties (relations between concepts):

- *assumes:* a task assumes one or more Competences.

- *hasFunction:* refers to the utility class *Function_Relation* to describe the relation between task, person and function; a person has assigned functions/work within a task.
- *hasLessonsLearned:* a task may have lessons learned
- *hasPersonInCharge:* each task has an employee, who is responsible for this task.
- *hasSubTask:* a task can be subdivided into several sub-tasks.
- *includes:* a task includes documents, such as requirements or project description.
- *isPredecessorOf:* task is able to be a predecessor of another task.
- *isSuccessorOf:* task is able to be a successor of another task.

Datatype Properties:

- *hasBeginDate:* a task has an expected start date.
- *hasDescription:* a task has a detailed description.
- *hasEndDate:* a task has an expected end date.
- *hasFinishDate:* a task has an effective end date.
- *hasName:* name/title of the task.
- *hasStartDate:* task has an effective start date.
- *hasStatus:* defines the process of the task; e.g., in progress, in planning or finished.
- *isMilestone:* task can be a milestone; either true or false. In the PM ontology the definition of a milestone is: a task is a milestone if its a special task that receives special attention.
- *isProject:* task can be a project; either true or false. Whenever a task is no sub-task of another task it is a project.

Resource

The concept *Resource* is a resource to perform a process. For instance, a resource is a person, working fund or working time. For the needs of my work it just contains the subconcept *Person*, but it can be easily extended with other subconcepts like *WorkingFund* or *Hardware*.

Person

A person is any employee. A person can be assigned to a task and within the task to an activity. In addition, a person works well with another person and has several competences.

Properties of the concept *Person* (depicted in table 5.2)

Object Properties:

- *isCreatorOf:* a person made up a document.
- *isResponsibleFor:* a person is responsible for a task.
- *uploads:* a person uploads a document to the project management system.
- *workswell:* a person works well with another person.
- *hasCompetence:* a person has several competences.

Datatype Properties:

- *Company:* company the person is working in.
- *hasBirthday:* birthday of the person.
- *hasFirstName:* first name (given name) of a person.
- *hasLastName:* last name (surname) of a person.

Person	
hasCompetence	*Competences*
isCreatorOf	*Information*
isResponsibleFor	*Task*
uploads	*Information*
workswell	*Person*
Company	string
hasBirthday	date
hasFirstName	string
hasLastName	string

Table 5.2: Properties of the concept Person (the properties in italic are object properties, whereas the underlined are datatype)

Information

The concept *Information* contains any kind of project relevant documents; e.g., project plan or documentation of a software module.

Information	
belongsTo	*Task*
hasCreator	*Person*
hasRelatedInformation	*Information*
hasUploader	*Person*
isType	*Document_Type*
hasContentDescription	string
hasCreationDate	date
hasFilename	string
hasFormat	string
hasUploadDate	date
hasStored	string

Table 5.3: Properties of the concept Information (the properties in italic are object properties, whereas the underlined are datatype)

Properties of the concept Information (depicted in table 5.3)

Object Properties:

- *belongsTo:* the information (document) belongs to a task.
- *hasCreator:* a person is creator of the information.
- *hasRelatedInformation:* a document is able to be related to another document.
- *isType:* defines the type of document (Document_Type). For instance, project handbook, requirements or project proposal.
- *hasUploader:* the document is uploaded by a Person.

Datatype Properties:

- *hasContentDescription:* a detailed description of the information.
- *hasContentTitle:* title of the document.

- *hasCreationDate:* the date of creation of the information.
- *hasFileName:* name of the stored document.
- *hasFormat:* type of document; e.g., pdf, MSWord or MSExcel.
- *hasUploadDate:* the date of uploading the information.
- *isStored:* the path where the information is stored.

Competences

The concept *Competences* contains any kind of competences of a person or a task. The classification (depicted as subconcepts) is as follows, assumed by Erpenbeck and Rosenstiel [28] and Biesalski [8] (already shown in figure 5.5):

- methodological competence,
- personal competence,
- social competence and
- professional competence.

MethodologicalCompetence is the qualification to use specific learn and working methods.

PersonalCompetence is, for instance, accuracy or to work under pressure.

The last subconcept **SocialCompetence** includes social, emotional and intellectual skills.

In the PM ontology the concept **ProfessionalCompetence** contains in my work competences in the field of IT (concept *IT-Competence*). If necessary any other domain can be added.

The subconcept *IT-Competence* implies the following classifications (subconcepts):

- Administration
- Application Development
- Databases
- Operating Systems
- Programming Languages
- Security
- Semantic Web
- Support

This classification is a small selection of possible categorisations of the professional competence IT and was defined by myself. It just outlines a possible classification and it was sufficient to test the ontology and the prototype. When I identified these sub-concepts I tried to cover a wide range of the working areas of the domain of Information Technology. The classification can be extended optionally.

Properties of the concept *Competences* (shown in table 5.4)

Datatype Properties:

- *hasNotation:* a description of the competence.
- *hasValue:* the value of the competence is beween 0.1 and 1. It characterizes the rating of a competence a person has or a task requires.

Competences	
hasNotation	string
hasValue	decimal

Table 5.4: Properties of the concept Competence (the properties in italic are object properties, whereas the underlined are datatype)

LessonsLearned

LessonsLearned is any kind of information of experiences or hints aggregated during an existing project, e.g. a bad team composition or problems during a development process, which helps to improve further projects.

LessonsLearned	
hasComment	string
hasText	string
hasTitle	string

Table 5.5: Properties of the concept LessonsLearned (the properties in italic are object properties, whereas the underlined are datatype)

Properties of the concept *LessonsLearned* (illustrated in table 5.5)

Datatype Properties:

- *hasComment:* if another person wants to write anything else to a lesson learned.
- *hasText:* detailed description of the lesson learned.
- *hasTitle:* name / title of the lesson learned.

Function_Relation

This concept is sort of a utility class. It enables the trinary relation between the concepts *Task*, *Person* and *Function*. A task implies some functions/activities, which will be assigned to at least one person.

Properties of the concept *Function_Relation* (highlighted in table 5.6)

Object Properties:

- *FunctionExecution:* an assignment/a function within a task will be performed by a predefined person.
- *hasFunctionDescription:* an assignment/a function within a task has a predefined specification/explanation.

Datatype Properties:

- *hasPercentage:* estimated amount of work quoted in percent.

Function_Relation	
FunctionExecution	*Person*
hasFunctionDescription	*Function*
hasPercentage	string

Table 5.6: Properties of the concept Function_Relation (the properties in italic are object properties, whereas the underlined are datatype)

Function

The concept *Function* implies the activity within a task. Each task requires one or even more fields of duties.

Properties of the concept *Function* (depicted in table 5.7)

Object Properties:

- *implies:* a task contains function. The function implies predefined competences.

Datatype Properties:

- *Description:* a detailed description of the function.
- *hasFunctionName:* name/title of the function.

Function	
implies	*Competence*
Description	string
hasFunctionName	string

Table 5.7: Properties of the concept Function (the properties in italic are object properties, whereas the underlined are datatype)

Document_Type

Classifies the type of document/information. Types of docments are for instance project handbook, requirements, project proposal or status report.

GoodTeam_Relation

Is related to the concept *LessonsLearned*. This concept is a utility class of the concept *LevelOfRelations*, which is a subconcept of *LessonsLearned*. It describes the trinary relation between two persons and the explanation why these persons are working well together. This concept connects the concepts *Person* and *Reasons*. The object properties are shown in table 5.8.

GoodTeam_Relation	
withPerson1	*Person*
withPerson2	*Person*
withReason	*GoodReason*

Table 5.8: Properties of the concept GoodTeam_Relation (the properties in italic are object properties, whereas the underlined are datatype)

Problem_Relation

Is related to the concept *LessonsLearned*. This concept is a utility class of the concept *LevelOfRelations*, which is a sub-concept of *LessonsLearned*. It describes the trinary relation between two persons and the explanation why these persons are not working well together and what the problems between these persons are. This concept connects the concepts *Person* and *Reasons*; highlighted in table 5.9.

Problem_Relation	
hasPerson1	*Person*
hasPerson2	*Person*
hasReasons	*BadReasons*

Table 5.9: Properties of the concept Problem_Relation (the properties in italic are object properties, whereas the underlined are datatype)

Reasons

This concept is subdivided into the concepts *BadReasons* and *GoodReasons*.

BadReasons describe the explanation why persons cannot work together.

Whereas **GoodReasons** describe the statement why persons work well together exceedingly.

5.3 Relations

All relations between concepts of the PM ontology are listed in the tables 5.10 and 5.11.

Relations		
Domain	**Object Property**	**Range**
Function_Relation	***FunctionExecution***	Person
Function_Relation	***hasFunctionDescription***	Function
Function	***implies***	Competences
GoodTeam_Relation	***withPerson1***	Person
GoodTeam_Relation	***withPerson2***	Person
GoodTeam_Relation	***withReason***	GoodReason
Problem_Relation	***hasPerson1***	Person
Problem_Relation	***hasPerson2***	Person
Problem_Relation	***hasReasons***	BadReasons

Table 5.10: Relations of the project management ontology (Part 1)

The relations between the main concepts are illustrated in figure 5.7 and can be described as follows:

Relations		
Domain	**Object Property**	**Range**
Task	***assumes***	Competences
Task	***hasFunction***	Function_Relation
Task	***hasLessonsLearned***	LessonsLearned
Task	***hasPersonInCharge***	Person
Task	***hasSubTask***	Task
Task	***includes***	Information
Task	***isPredecessorOf***	Task
Task	***isSuccessorOf***	Task
Person	***isCreatorOf***	Information
Person	***isResponsibleFor***	Task
Person	***uploads***	Information
Person	***workswell***	Person
Person	***hasCompetence***	Competences
Information	***belongsTo***	Task
Information	***hasCreator***	Person
Information	***hasRelatedInformation***	Information
Information	***isType***	Document_Type
Information	***hasUploader***	Person

Table 5.11: Relations of the project management ontology (Part 2)

A task is able to contain sub-tasks (*hasSubTask*). This implies a hierarchical structuring of a project/task into smaller working areas, like work packages or deliverables. If a task is no sub-task of another task, it is called a project. Other dependencies between Tasks are (i) successor (*isSuccessorOf*) and (ii) predecessor (*isPredecessorOf*) of a task. In addition, a task has a person, who is responsible for it (*hasPersonInCharge*). The inverse relation is *isResponsibleFor*. Furthermore, a task includes several documents (Information; *includes*, inverse relation *belongsTo*) and implies lessons learned (*hasLessonsLearned*). To implement each task, the task *assumes* competences, which have to be fulfilled by persons. In addition, a task has a function (*hasFunction*), which will be accomplished by a person (*FunctionExecution*) and described by the job/function (*hasFunctionDescription*). Moreover, the function requires competences (*implies*).

Additionally, a person is able to create (*isCreatorOf*, inverse relation is *hasCreator*) and upload (*uploads*, inverse relation: *hasUploader*) documents. These may also be related to other ones (*hasRelatedInformation*). Each document can be classified as certain project document (*isType*), such as project handbook or functional specification. A person is able to work well (*workswell*) with other persons and possesses several competences (*hasCompetence*).

As a result the input of lessons learned is divided into several categories, relations between employees can be specified as good or bad teamwork. If it is a good team relation, employees have to be specified (*withPerson1, withPerson2*) and a short explanation must be given (*withReason*). In fact of a bad teamwork employees have also be mentioned (*hasPerson1, hasPerson2*) as well as a short description (*hasReasons*).

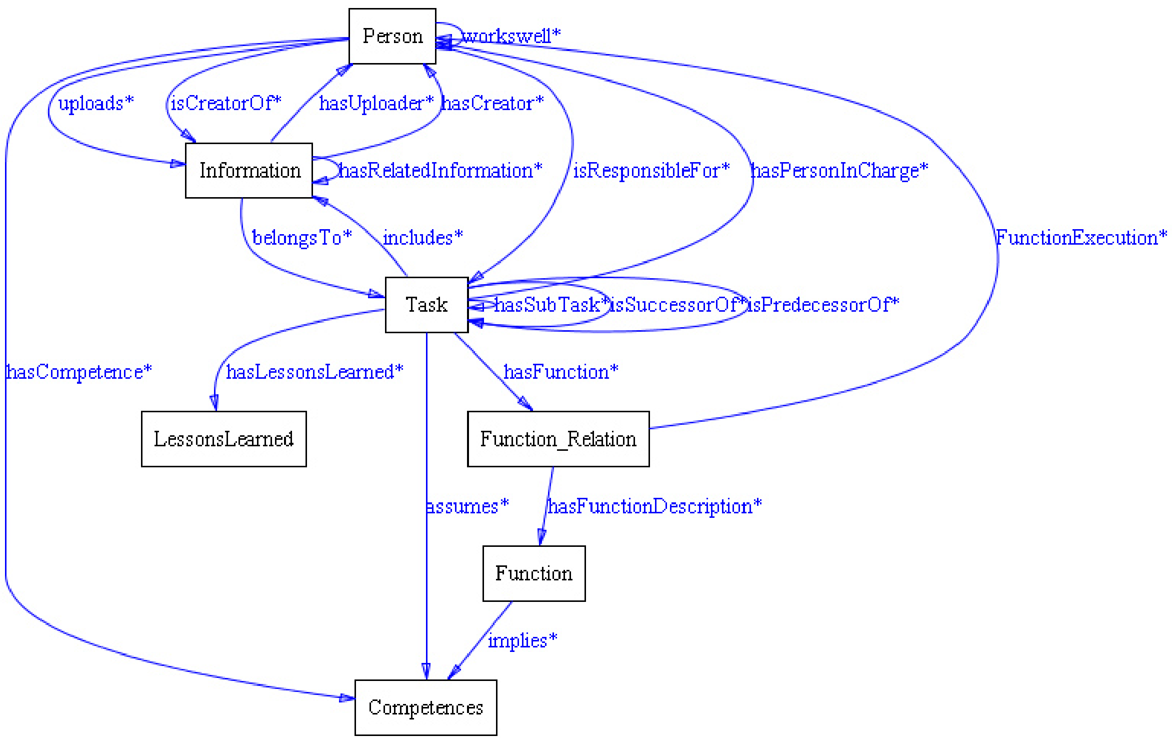


Figure 5.7: Main concepts and their relations of the ontology

Concept, Properties and Relations in the View of the Scenarios

In the following I show that the questions depicted in chapter 4 “Scenarios” are covered by the ontology. As such these questions serve for modeling as well as evaluating the ontology.

The questions of scenario 1 “Bob needs Holidays” are as follows:

1. What is the process status of the deliverable?

2. What is the effective and planned end date of this deliverable?

3. Does the deliverable have any dependencies to other deliverables, tasks or work packages?

4. Is the deliverable of Bob a milestone?

5. Does the company have employees with the same competences as Bob? In which projects do they work? Do they have deliverables and/or tasks during the holidays of Bob?

6. Is Bob responsible for any other deliverables, work packages or tasks? What are the effective and planned end dates of these deliverables, work packages and tasks? Do they have dependencies to others?

Question 1 and 2 are answered by the datatype properties hasStatus, hasEndDate (expected end date) and hasFinishDate (effective end date) of the concept Task. Question 3 relies on the relations isSubTask, is SuccessorOf and isPredecessorOf of the concept Task. All these relations are connections between different task instances. The relation of the concepts Task and Person as well as the datatype property isMilestone of the concept Task answer question 4.

To tackle question 5 the concepts Competences, Person, Task, Function_Relation and Function and their relations are necessary. In addition, some datatype properties are required.

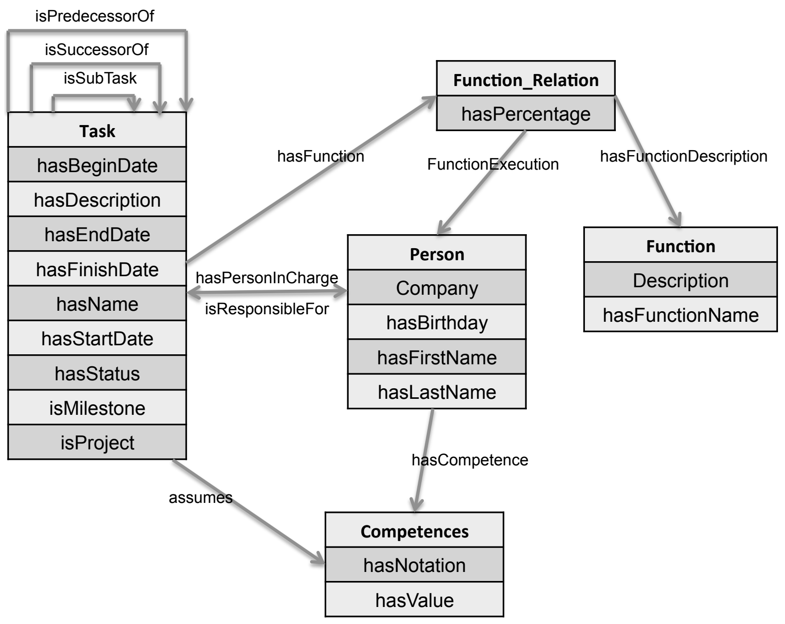


Figure 5.8: Concepts, Properties and Relations of Scenario 1

Question 6 summarizes Question 1 to 3; thus, includes the concepts Task and Person, their relation among themselves as well as Task with each other. In addition, properties about the end and finish dates of the tasks. The relations as well as properties are shown in figure 5.8.

Scenario 2 “A new Project” contains the following questions:

1. Are there similar projects/tasks to the project “Easy Life” available? (e.g., name, descriptions, software in use)
2. Are there already lessons learned available which can be used for this project?
3. Can already stored information be reused for this project? (e.g., software modules, documentation)

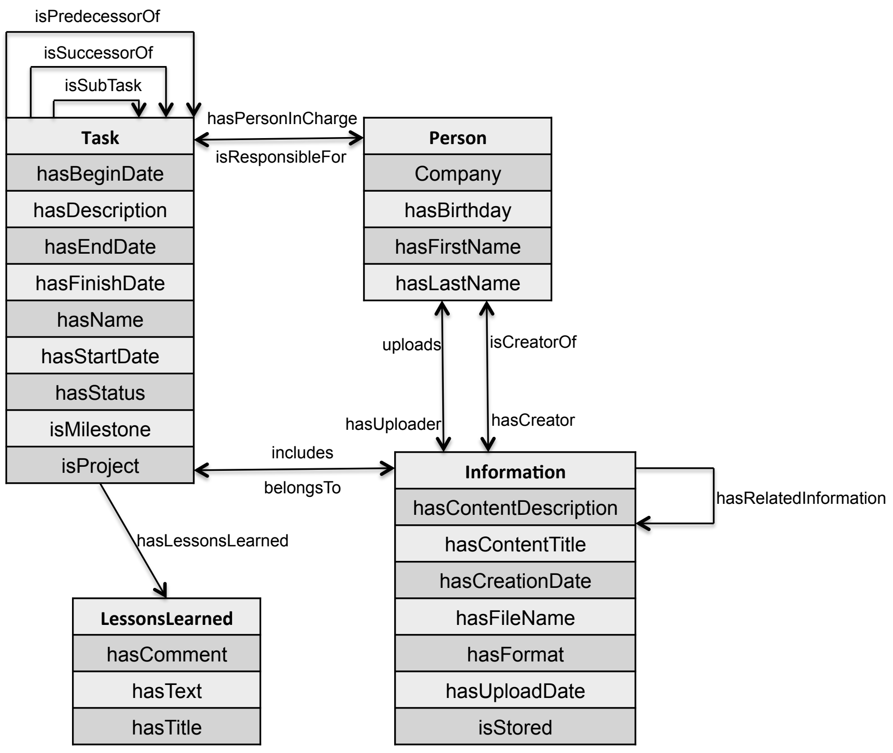


Figure 5.9: Concepts, Properties and Relations of Scenario 2

To answer question 1 the datatype properties hasName and hasDescription of the concept Task are compared with stored instances. Question 2 contains the relation of the concepts Task and LessonsLearned. In addition, the datatype properties hasTitle, hasComment and hasText of available lessons learned are scoured for entries, which will improve and support the new project. To tackle question 3 the concept Information (any kind of document) and its relations to the concepts Task or Person as well as its datatype properties are required. The responsible part of the ontology to solve these questions are illustrated in figure 5.9.

Last, the questions of scenario 3 “Team suggesstions, etc.” are:

1. Which employees have the most adequate competences for the project/tasks?
2. Do the employees already work in other projects?
   In which time period do the personnel work in other projects/tasks?
   In which projects and tasks are the employees already working?
   Are they responsible for tasks or projects?
   What are they working in the projects or tasks?
3. Do the chosen employees work well with others? What are their competences?
4. Do the competences match with competences needed in the new project?

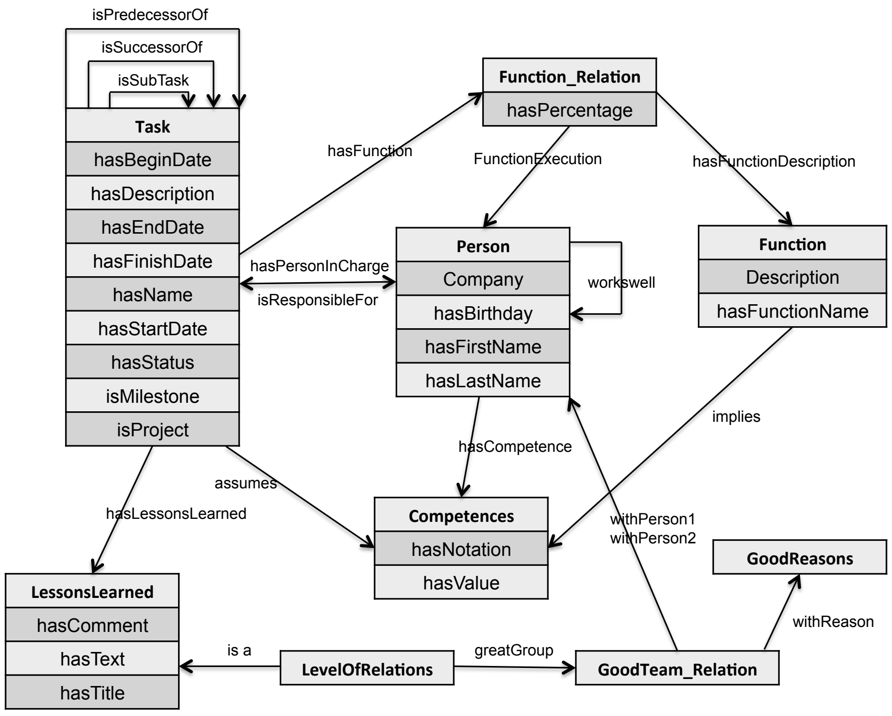


Figure 5.10: Concepts, Properties and Relations of Scenario 3

Question 1 requires the relation of the concepts Person, Competences and Task. In addition, the competences the task assumes and the person has are compared with each other. To answer question 2 the relations of the concepts Person and Task are important. Also, the datatype properties hasBeginDate and hasEndDate of the concept Task are relevant. Likewise, the concepts Function_Relation and Function are an issue to assess the work of a person within a project. Question 3 requires the relations of the concepts Person and Competences as well as the relation to the concept LessonsLearned and to the sub-concept LevelOfRelations. This concept refers to the utility classes GoodTeam_Relation or BadTeam_Relation. The last question is tackled with the outcome of question 1 and 3. The necessary concepts, relations and properties are highlighted in figure 5.10.

By itself the ontology shall not be considered as finished. Although, it covers all necessary aspects of the scenarios. It can be rated as foundation for further work. Thus, the individual parts, for instance, competences and lessons learned, can be easily extended. For instance, if a project management system within the domain of architecture is needed, the ontology can easily be enhanced with the structuring of professional knowledge of, e.g., architecture.

CHAPTER 6

Prototype

In this chapter I give an overview of the prototype, its architecture and structure.

6.1 Architecture

When I started with this PhD, I designed a rough architecture of the prototype, depicted in figure 6.1. There it was already specified that the chosen project management system will be a central server solution; thus, consisting of a web server and a database. Each PM system, identified by an internet research, consisted of a web server and a database. Documents are stored on the same server. As renewal a project management ontology [21] enhances and improves the project management system including interfaces to connect the system and the ontology. This rough architecture describes the main components of the PM system and the added modules.

Concept - General Architecture

My work to improve a project management system with semantic technologies, should be applicable to any PM system. Thus, I designed the core modules of the architecture of the system as depicted in figure 6.2.

The "PM Knowledge Base" displays the environment of the core project management system, including the ontology, the MySQL database and the document storage. The ontology depicts main parts of project management in order to obtain a knowledge base. Such parts are information of projects and tasks, employees (person), competences employees have and projects require, lessons learned of finished projects as well as information (documents). Stored documents are related to the concept information for the purpose of recovering them as easy as possible. The MySQL database is required by the SPARQL API ARC and dotProject.

The "User Interface" illustrates the application for the end-user (project manager or member). If this person wants to insert or change project relevant information (edit/update), the sys-

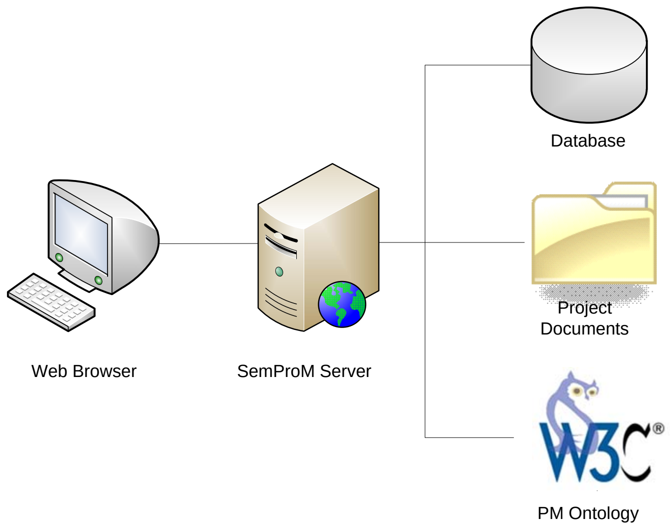


Figure 6.1: Rough architecture of the prototype

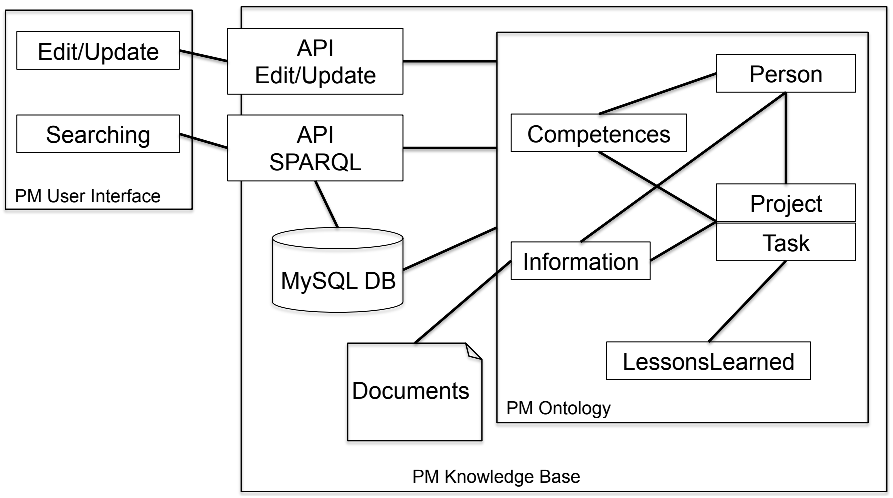


Figure 6.2: Core modules of the PM system

tem activates the interface API - "Edit/Update" to accomplish this work. The "API - SPARQL" is activated in order to query information of the knowledge base.

6.2 'SemProM' Prototype - Approach

First when implementing the prototype I had a deeper look into the structure of dotProject to identify modules and scripts which must be changed and added. dotProject consists of a MySQL database and an Apache webserver. Second, to cover the whole PM ontology within the prototype I compared the MySQL database with the ontology. Objective of this work was to identify tables and columns to match my predefined concepts and properties of the ontology with the dotProject database. Table 6.1 and 6.2 show the results.

Ontology	**dotproject mySQL**	
	table	column
Task		
assumes (Competences)		
hasBeginDate	tasks projects	task_start_date project_start_date
hasDescription	tasks projects	task_description project_description
hasEndDate	tasks projects	task_end_date project_end_date
hasFinishDate	Projects	project_actual_end_date
hasFunction (Function_Relation)		
hasLessonsLearned (LessonsLearned)		
hasName	tasks projects	task_name project_name
hasPersonInCharge (Person)	tasks projects	task_owner project_owner
hasStartDate		
hasStatus	tasks projects	task_status task_percent_complete project_status project_percent_complete
hasSubTask (Task)	tasks	task_parent (←→) task_project
includes (Information)	files	file_project file_task
isMilestone	tasks	task_milestone
isPredecessorOf (Task)		
isSuccessorOf (Task)	tasks	task_parent

Table 6.1: Comparison of the ontology and database properties (Part 1)

The first column (Ontology) outlines the ontology properties. The second column (table) highlights the table of the dotProject database and the third one the column (column) of the table. Table 6.1 illustrates the properties of the concept "Task" of the ontology. For instance, the

Ontology	dotproject mySQL	
	table	column
Person		
Company	contacts	contact_company
hasBirthday	contacts	contact_birthday
hasCompetence		
hasFirstName	contacts	contact_first_name
hasLastName	contacts	contact_last_name
isCreatorOf (Information)		
isResponsibleFor (Task)	contacts	contact_project
uploads (Information)		
workswell (Person)		
Information		
belongsTo (Task)	files	file_project file_task
hasContentDescription	files	file_description
hasContentTitle	files	file_name
hasCreationDate		
hasCreator (Person)	files	file_owner
hasFileName	files	file_real_filename
hasFormat	files	file_type
hasRelatedInformation (Information)	files	file_parent
hasUploadDate	files	file_date
hasUploader (Person)	files	file_owner
isStored	files	file_folder
Function		
Description		
implies (Competences)		

Table 6.2: Comparison of the ontology and database properties (Part 2)

property “hasBeginDate” is equivalent to “task_start_date” and project_start_date” in the table “tasks” and “project” of the database. In contrast “assumes” has no equivalent opposite within the database.

Another important task was to decide whether

- I skip the dotProject database and store all information within the ontology,
- to store the covered information within the ontology and all others within the database or
- to store the information twice, in the ontology as well as database.

Due to the fact I integrated the PM ontology into an existing project management system, I decided to take a mixture of alternative one and two. This is, because the PM ontology does not cover all modules of the system dotProject. For instance, Companies or Departments are not covered. The ontology just contains a property "Company" of the concept "Person", which has the meaning of "the person is working in the company xyz". Thus, I identified the scripts, which contain the functions regarding adding and updating information within the database and extended and modified them with the functions for editing and updating the ontology. This was facilitated by the modularity of the system.

The modified and added scripts of the PM system dotProject are as follows:

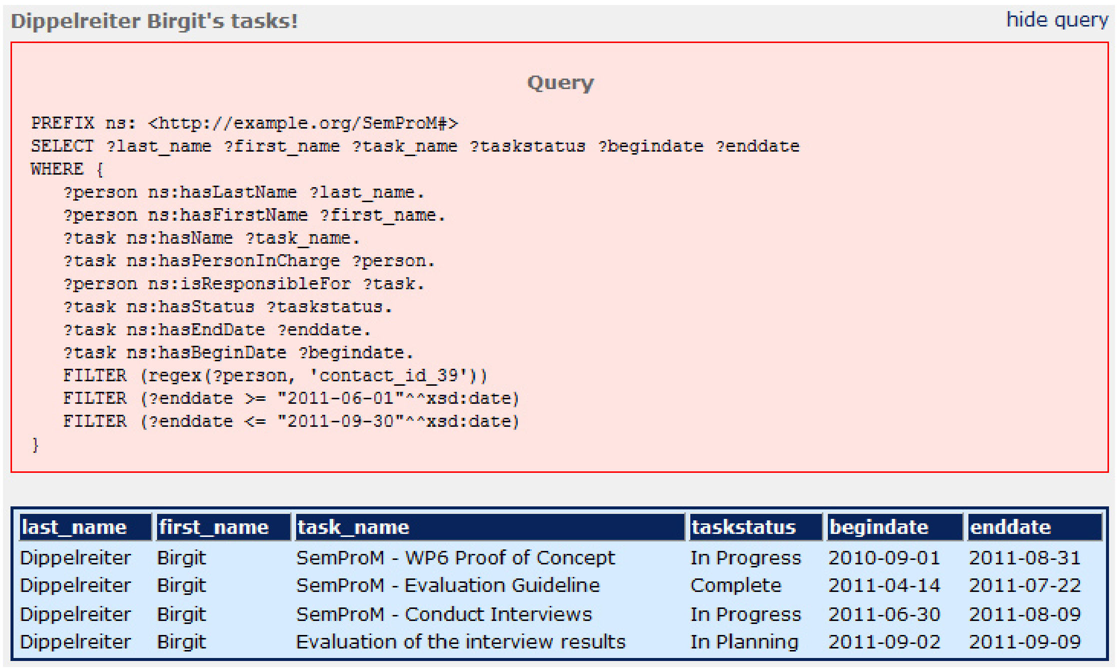
- *Modified*
  - /index.php
    This is the start script of dotProject. There I added the path to the rdf_handler and interface script. This script is in the root folder of dotProject.
  - /modules/<module>/do_<module>_aed.php
    These scripts are responsible for adding, editing and deleting module data. They were modified to call the do_<module>_rdf.php script.
  - /modules/<module>/addedit.php
    These scripts are concerned to define the user interface to add, edit and delete module data. This is necessary, because the input fields must be consistent with the ontology.
  - /modules/<module>/view.php
    The user interface of the output of information regarding specific modules, such as project or task, is covered by the view.php script. It was adapted to display also the information stored in the ontology. The information, stored in the ontology, is read out by the script <module>.functions.php.
- *Added*
  - /RDF/SemProM.owl
    The project management ontology converted in RDF/XML format.
  - /RDF/sparql.php
    The implementation of the SPARQL Endpoint.
  - /RDF/sparql_test.php
    Some tests of the SPARQL Endpoint.
  - /RDF/sparql/*
    This folder includes all files regarding the ARC API (ARC2 scripts).

- /sparql/images/*
  Includes all images concerning the SPARQL module.
- /sparql/index.php
  It is the homepage of the SPARQL module
- /sparql/index.html
  The system dotProject requires such a HTML-File. It is empty.
- /sparql/setup.php
  This script is the setup file for the SPARQL module.
- /classes/SparqlQueryBuilder.class.php
  This class contains the building of SPARQL Queries and the parsing of the results within a HTML table.
- /includes/rdf_handler.php
  This script contains the functionalities to work on the ontology (RDF/XML file). It is enhanced by the scripts do_<module>_rdf.php.
- /includes/rdf_interface.php
  Interface of the do_<module>_rdf.php script.
- /modules/<module>/do_<module>_rdf.php
  This script extends the rdf_handler.php and is responsible to save the RDF data (information) of a module (e.g., project, task or file). The code contains functions to update or add information of a person to the ontology. In addition, to add or update the relation of workswell and of competences.
- /modules/<module>/<module>.functions.php
  It provides module specific features to extract information from the ontology.
- /modules/contacts/ae_competences.php
  This script contains the selection of competences. It is included within the script addedit.php of contacts.
- /modules/projects/ae_competecens.php
  Similar to the ae_competences.php script. This script is contained in the script addedit.php of projects and tasks.
- /modules/projects/ae_functions.php
  Quite similar to the two scripts just mentioned. Instead of competences it comprises the selection of functions. This script is contained in the script addedit.php of projects and tasks.
- /modules/projects/vw_sparql.php
  It is the user interface to create SPARQL queries.
- /modules/projects/vw_sparql_evaluate.php
  User interface to illustrate the results of the script vw_sparql.php.
- /modules/projects/vw_revision.php
  User interface of the task and project revision part.

- /modules/projects/vw_revision_person.php
  User interface for more and detailed information of the revision output of a person.
- /modules/projects/vw_revision_task.php
  It is the user interface for more and detailed information of the revision output of tasks.
- /modules/lessonslearned/*
  Contains the lessons learned module scripts.
- /js/jquery-1.4.3.min.js

<module> is taken for task, project, contact, file or lessonslearned; for the folder as well as script name.

To improve the usability of the prototype I replaced some java scripts by the “popular” package jQuery. It enables me to provide more efficient interfaces and simplifies the handling of the application.


Dippelreiter Birgit's tasks! hide query

Query

```
PREFIX ns: <http://example.org/SemProM#>
SELECT ?last_name ?first_name ?task_name ?taskstatus ?begindate ?enddate
WHERE {
   ?person ns:hasLastName ?last_name.
   ?person ns:hasFirstName ?first_name.
   ?task ns:hasName ?task_name.
   ?task ns:hasPersonInCharge ?person.
   ?person ns:isResponsibleFor ?task.
   ?task ns:hasStatus ?taskstatus.
   ?task ns:hasEndDate ?enddate.
   ?task ns:hasBeginDate ?begindate.
   FILTER (regex(?person, 'contact_id_39'))
   FILTER (?enddate >= "2011-06-01"^^xsd:date)
   FILTER (?enddate <= "2011-09-30"^^xsd:date)
}
```

last_name	first_name	task_name	taskstatus	begindate	enddate
Dippelreiter	Birgit	SemProM - WP6 Proof of Concept	In Progress	2010-09-01	2011-08-31
Dippelreiter	Birgit	SemProM - Evaluation Guideline	Complete	2011-04-14	2011-07-22
Dippelreiter	Birgit	SemProM - Conduct Interviews	In Progress	2011-06-30	2011-08-09
Dippelreiter	Birgit	Evaluation of the interview results	In Planning	2011-09-02	2011-09-09

Figure 6.3: SPARQL Code to review the Query

When starting the implementation of the prototype, I was processing the following way:

1. store and update information in the ontology
2. adapt existing user interfaces & modules

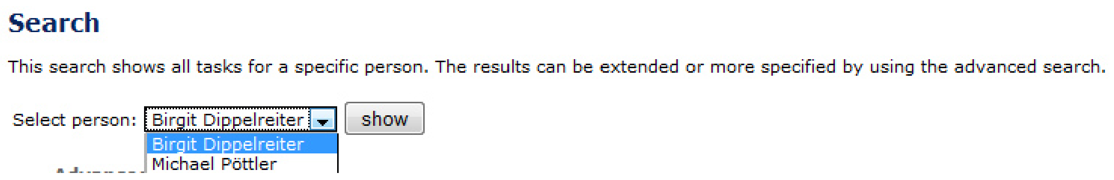

Search
This search shows all tasks for a specific person. The results can be extended or more specified by using the advanced search.
Select person:
Birgit Dippelreiter
show
Birgit Dippelreiter
Michael Pöttler

Figure 6.4: Search functionality - selection of person

3. implement new modules (functionality & user interface)

Adapted modules are, for instance, project or task. Whereas new modules are “lessons learned”, the “search functionality” and the “advanced SPARQL query search”. In its origin dotProject did not offer any “lessons learned” module.

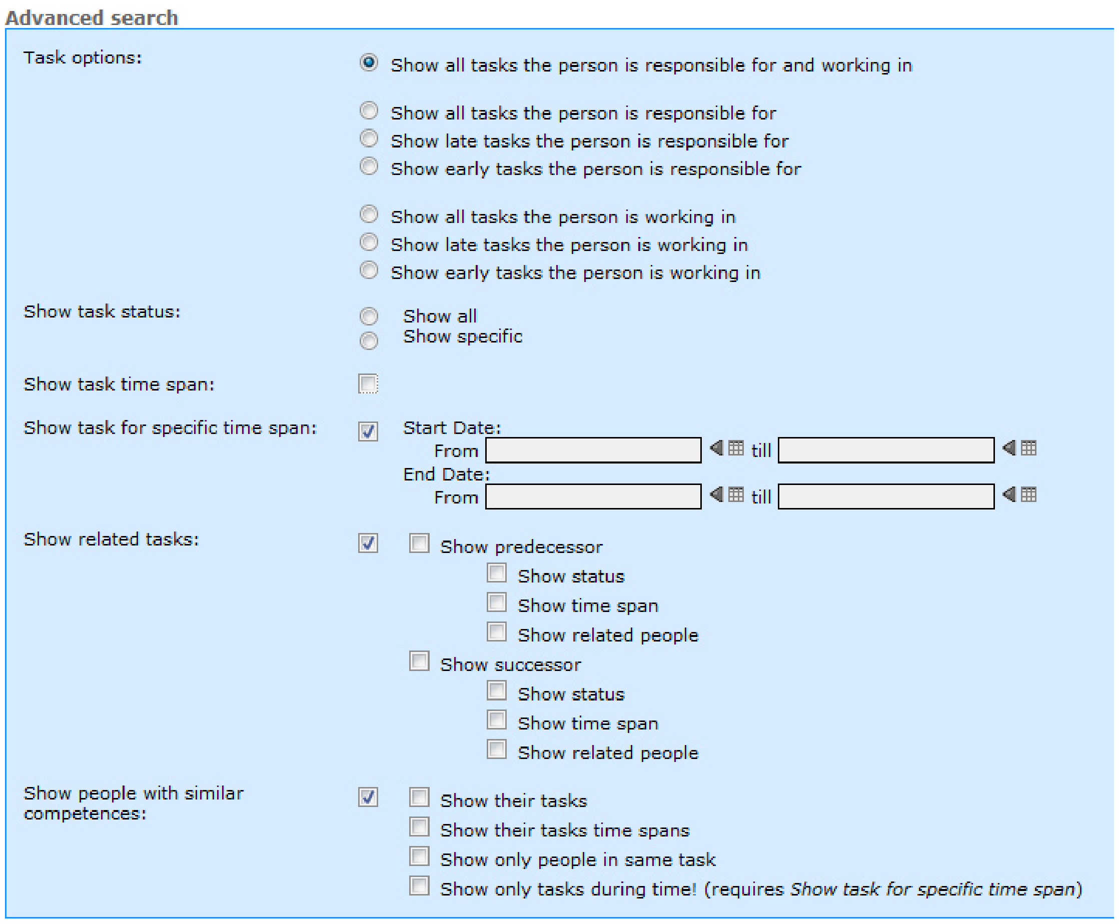
After each implementation part I carried out some functionality tests. To proof the SPARQL queries regarding the search functionality, I added the SPARQL code (as possible display) to the search results; shown in figure 6.3. In addition, I implemented an advanced SPARQL query search. Therewith, users are able to test new SPARQL queries or to search for a not yet implemented search request. Furthermore, during the implementation process of the search functionality this advanced search supported me to check and improve the SPARQL queries concerning the search functionality.

To use the “search functionality” the user does not have to know the syntax of SPARQL queries or SPARQL anyway. The queries within the search functionality cannot be changed by the users. The only possibility to define the query is to select the search criteria via the user interface. The “search functionality” is shown in figure 6.4 and 6.5. Figure 6.4 displays the selection of the person, whereas figure 6.5 highlights the configuration of the SPARQL query.

If the user wants to apply the “advanced SPARQL query search”, he/she must know how to generate SPARQL queries as well as the structure of the project management ontology (shown in figure 6.6).

Interfaces

In order to implement the prototype I required an API - Application Programming Interface - as interface between the PM ontology and dotProject. For this purpose, I carried out an internet research concerning available PHP APIs. After a short evaluation of existing ones I realized that there exists no usable OWL APIs. Thus, I decided to use a RDF API. Therefore, I had to convert the PM ontology to the format RDF XML. Nevertheless, the functionality of the ontology is still the same as in OWL. The chosen API was the RAP [58] - RDF API for PHP - for inserting and updating instances within the ontology. The latest version of RAP is the version 0.9.6 and was published in February 2008.



Advanced search

Task options:
- [✓ Show all tasks the person is responsible for and working in]
- Show all tasks the person is responsible for
- Show late tasks the person is responsible for
- Show early tasks the person is responsible for
- Show all tasks the person is working in
- Show late tasks the person is working in
- Show early tasks the person is working in

Show task status:
- Show all
- Show specific

Show task time span: ☐

Show task for specific time span: ✓
- Start Date: From ____ till ____
- End Date: From ____ till ____

Show related tasks: ✓
- ☐ Show predecessor
  - ☐ Show status
  - ☐ Show time span
  - ☐ Show related people
- ☐ Show successor
  - ☐ Show status
  - ☐ Show time span
  - ☐ Show related people

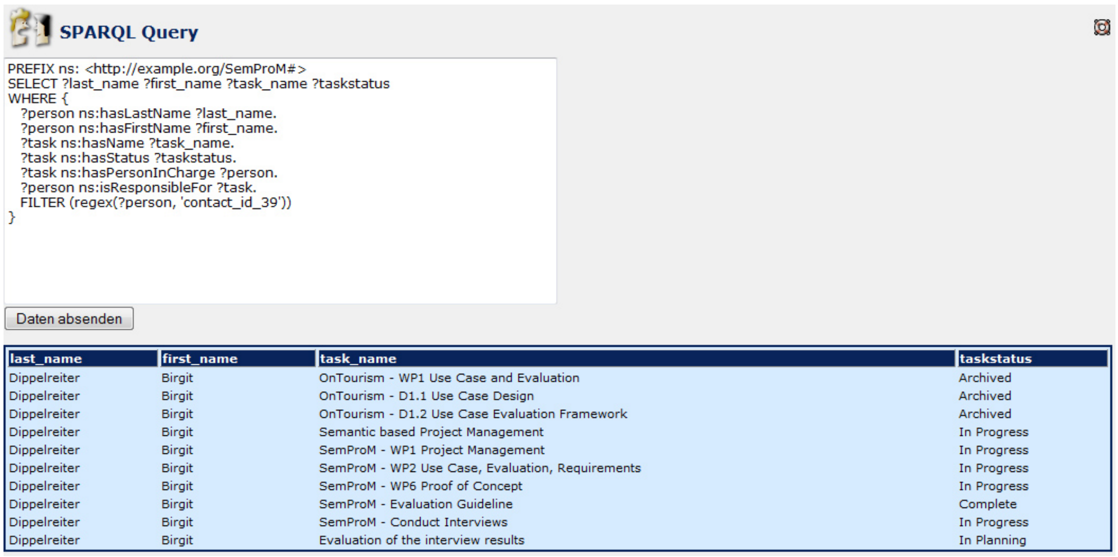
Show people with similar competences: ✓
- ☐ Show their tasks
- ☐ Show their tasks time spans
- ☐ Show only people in same task
- ☐ Show only tasks during time! (requires *Show task for specific time span*)

Figure 6.5: Search functionality - configuration

However, when I tried to integrate RAP to the prototype system I had to realize that different from the assumptions, it did not fullfill the required tasks. Thus, I implemented an API, including a RDF handler class and special module classes, which extended this handler. These modules perform the functions to store and update the instances within the ontology.

For implementing the SPARQL queries I used the API ARC [57]. SPARQL - SPARQL Protocol and RDF Query Language - is a RDF query language to query through the ontology instances. ARC was completely rewritten and is now called ARC2. The latest release is the version 2.0.0. The SPARQL engine of ARC requires a MySQL database.

To include the API ARC into the prototype I compiled a SPARQL Endpoint integrated into the RDF handler. As a result, SPARQL queries activate the SPARQL function within the RDF handler. To build up the SPARQL store I configured ARC to obtain access to the dotProject database, because it requires a database for storing information. For the simple reason that ARC



```
PREFIX ns: <http://example.org/SemProM#>
SELECT ?last_name ?first_name ?task_name ?taskstatus
WHERE {
  ?person ns:hasLastName ?last_name.
  ?person ns:hasFirstName ?first_name.
  ?task ns:hasName ?task_name.
  ?task ns:hasStatus ?taskstatus.
  ?task ns:hasPersonInCharge ?person.
  ?person ns:isResponsibleFor ?task.
  FILTER (regex(?person, 'contact_id_39'))
}
```

last_name	first_name	task_name	taskstatus
Dippelreiter	Birgit	OnTourism - WP1 Use Case and Evaluation	Archived
Dippelreiter	Birgit	OnTourism - D1.1 Use Case Design	Archived
Dippelreiter	Birgit	OnTourism - D1.2 Use Case Evaluation Framework	Archived
Dippelreiter	Birgit	Semantic based Project Management	In Progress
Dippelreiter	Birgit	SemProM - WP1 Project Management	In Progress
Dippelreiter	Birgit	SemProM - WP2 Use Case, Evaluation, Requirements	In Progress
Dippelreiter	Birgit	SemProM - WP6 Proof of Concept	In Progress
Dippelreiter	Birgit	SemProM - Evaluation Guideline	Complete
Dippelreiter	Birgit	SemProM - Conduct Interviews	In Progress
Dippelreiter	Birgit	Evaluation of the interview results	In Planning

Figure 6.6: Advanced SPARQL query search

does not support any HTML output format, I had to implement some parsers to receive the ARC results.

To implement the prototype I obtained help of a master student. This work was part of a project work. For his master thesis he implemented a RDF API for PHP [5], which was a further development of the API of this thesis.

Figure 6.7 outlines the application programming interface (API) between the project management system dotProject and the PM ontology. As already mentioned, I used two different APIs for the ontology access. The ARC API for SPARQL (Simple Protocol And RDF Query Language) Queries and the rdf_handler (*rdf_handler.php*) for editing and updating instances into the ontology. Whenever an information item, like a project or a task, is stored the system dotProject activates the script *do_<module>_aed.php*, which again includes the script *do_<module>_rdf.php*. This script contains functionalities to store new information items. If an information item will be changed, updated or viewed the system executes the script *<module>.functions.php*. (<module> is taken for task, project, contact, file or lessonslearned.)

Figure 6.8 displays the process "Insert a new Project". Thus, the script *do_project_aed.php* includes the script *do_project_rdf.php*, which contains the functionalities to insert a new project. This script again, activates *rdf_handler.php* to store the project in the ontology (SemProM.owl). The successful storage is dsiplayed by *rdf_interface.php*.
The process of the search functionality is shown in figure 6.9. The *SPARQL.php* script activates the *SparqlQueryBuilder.class.php*, which again includes the *rdf_handler.php*. This script activates the *ARC2.php*.

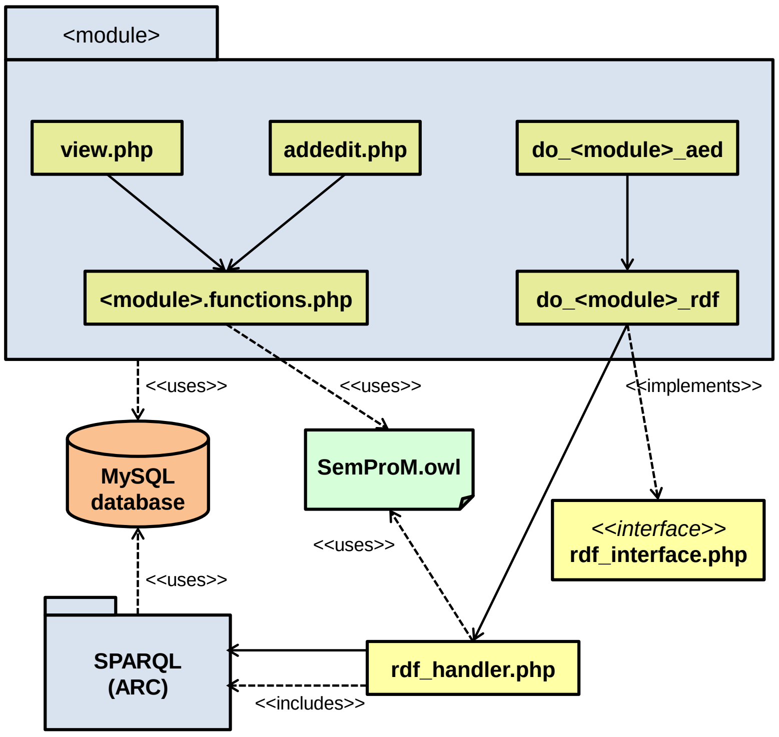


Figure 6.7: API architecture of the prototype

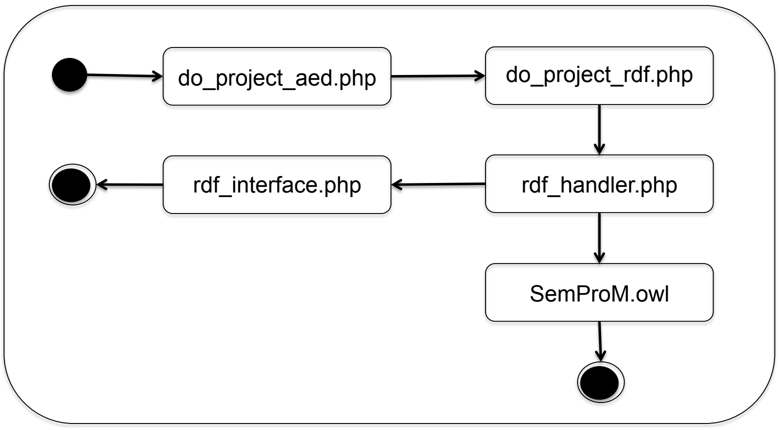


Figure 6.8: Activity Diagram "Insert a new Project"

SPARQL

Due to the fact that the SPARQL queries were part of the implementation of the prototype as well as of the evaluation process of my work, I decided to put their description to the evaluation chapter. Nevertheless, to obtain an impression of the SPARQL queries figure 6.10 highlights a

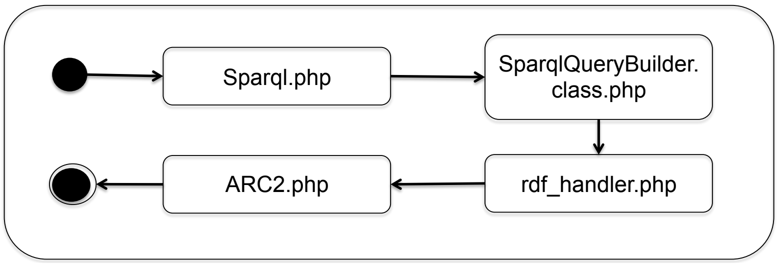
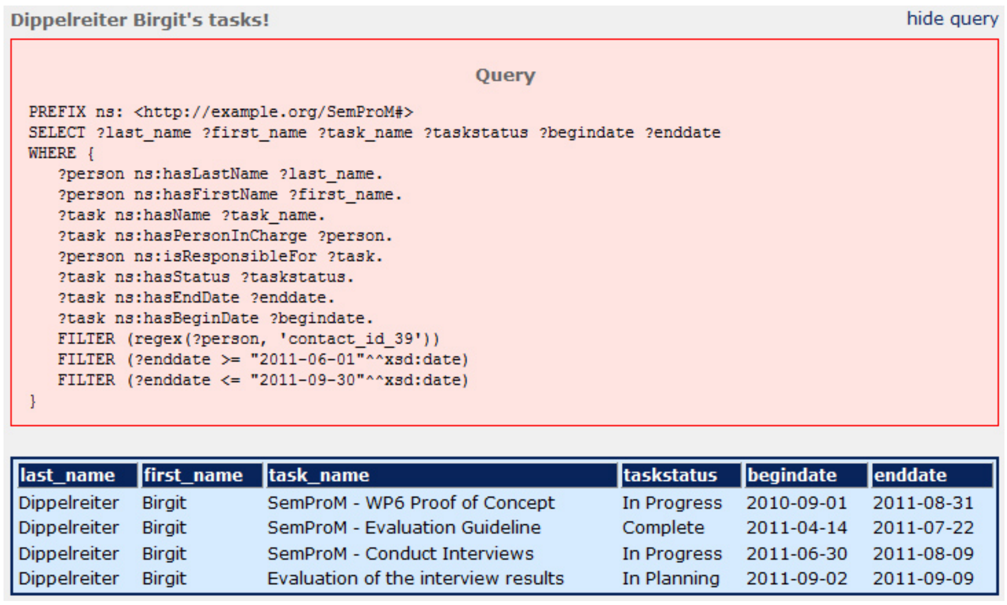


Figure 6.9: Activity Diagram “Search Functionality - SPARQL”

sample SPARQL query and the related output of:

- What are the projects and tasks “Birgit Dippelreiter” is currently responsible for?
- What is the process status of the projects/tasks?
- What is the planned end date of them?



Dippelreiter Birgit's tasks! hide query

Query

```
PREFIX ns: <http://example.org/SemProM#>
SELECT ?last_name ?first_name ?task_name ?taskstatus ?begindate ?enddate
WHERE {
   ?person ns:hasLastName ?last_name.
   ?person ns:hasFirstName ?first_name.
   ?task ns:hasName ?task_name.
   ?task ns:hasPersonInCharge ?person.
   ?person ns:isResponsibleFor ?task.
   ?task ns:hasStatus ?taskstatus.
   ?task ns:hasEndDate ?enddate.
   ?task ns:hasBeginDate ?begindate.
   FILTER (regex(?person, 'contact_id_39'))
   FILTER (?enddate >= "2011-06-01"^^xsd:date)
   FILTER (?enddate <= "2011-09-30"^^xsd:date)
}
```

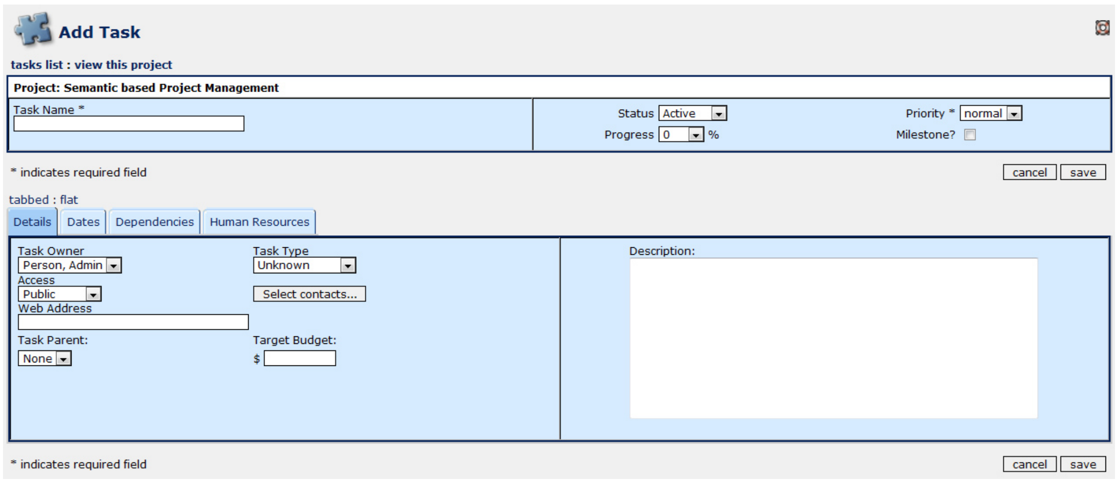
last_name	first_name	task_name	taskstatus	begindate	enddate
Dippelreiter	Birgit	SemProM - WP6 Proof of Concept	In Progress	2010-09-01	2011-08-31
Dippelreiter	Birgit	SemProM - Evaluation Guideline	Complete	2011-04-14	2011-07-22
Dippelreiter	Birgit	SemProM - Conduct Interviews	In Progress	2011-06-30	2011-08-09
Dippelreiter	Birgit	Evaluation of the interview results	In Planning	2011-09-02	2011-09-09

Figure 6.10: SPARQL query and results of informal competency questions of scenario 1 (Part 1 of 3)

The upper part of the figure covers the query while the result is illustrated in the lower part. The query requests all projects and tasks “Birgit Dippelreiter” is responsible for within a predefined end date period. The output lists all projects/tasks, including their name, status, begin and end date.

A detailed decription of the SPARQL queries and their outcome is presented in Chapter 7 “Evaluation”.

User Interface


Add Task

tasks list : view this project

Project: Semantic based Project Management

Task Name *

Status: Active
Progress: 0 %
Priority *: normal
Milestone?

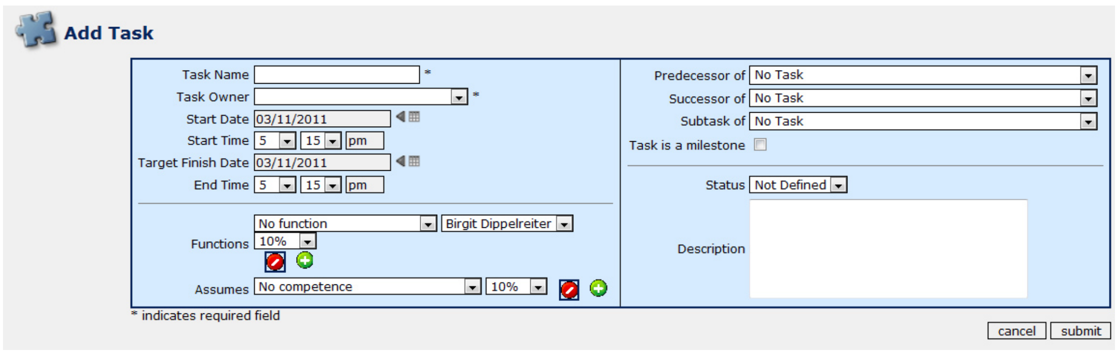
* indicates required field

tabbed : flat

Task Owner: Person, Admin
Task Type: Unknown
Access: Public
Web Address
Task Parent: None
Target Budget: $
Description:

* indicates required field

Figure 6.11: Screenshot: dotProject user interface “Add Task”


Add Task

Task Name *
Task Owner *
Start Date: 03/11/2011
Start Time: 5 15 pm
Target Finish Date: 03/11/2011
End Time: 5 15 pm
Functions: No function, Birgit Dippelreiter, 10%
Assumes: No competence, 10%

Predecessor of: No Task
Successor of: No Task
Subtask of: No Task
Task is a milestone
Status: Not Defined
Description

* indicates required field

Figure 6.12: Screenshot: SemProM user interface “Add Task”

Figure 6.11 depicts the original user interface of “Add Task” of dotProject, whereas figure 6.12 highlights the user interface of the SemProM prototype. The main difference of the user interface is that I removed all input fields, which are not covered by the PM ontology and put all input fields to one single page, whereas dotProject has four pages (Details, Dates, Dependencies and Human Resources) to add a new task. As figure 6.12 illustrates my focus was on the relation task, function and person as well as what competences are necessary to work within this task.

This is shown by the input fields *Functions* and *Assumes*. The information of the percentage beside the input field *Functions* declares that the assigned employee has a workload of a specific percentage for this task. The percentage information beside the *Assumes* input field indicates the rating of the competences, which are required to accomplish this task.

CHAPTER 7

Evaluation

This chapter contains the evaluation process, the evaluation results and the answer to the research question, including the description of the contributions.

7.1 Evaluation Process

In the evaluation phase I followed the aproach of Hevner et al. guidelines [39] and carried out a descriptive and observational method. Thus, I prepared an evaluation guideline based on the pre-defined scenarios (shown in chapter 4 "Scenarios") and accomplished a case study with project managers and members within a business environment.

Summarized, the main steps of my evaluation process are:

1. In order to obtain a serious and applied evaluation of my work I did interviews with project managers and members of IT projects, mostly in IT companies. As such, I used an observational method.
2. The next step I had to accomplish was to generate a test guideline. Based on the scenarios, this guideline covered "real" use cases. In addition, the guideline is a detailed manual how to use the prototype and how to carry out the use cases; a detailed step by step instruction. (The evaluation guideline is given in "Appendix B - Evaluation Guideline".) Reason for this decision was to simplify the usage of the prototype for the test persons. In addition, the main focus of the evaluation process was on the functionalities and the information outcome and not on the handling of the prototype.
3. I conducted six evaluation discussions. Three of the test persons were already consulted for the inteviews I carried out for getting information of strengths and weaknesses in current PM solutions. Likewise the interview results, the evaluation results are also anonymized. Table 7.1 outlines the domains of the companies. All test persons were project managersas well as project members.

Company	Sector
1	Bank sector
2	IT sector
3	IT sector
4	Public / IT sector
5	Public / IT sector
6	IT sector

Table 7.1: Overview of the companies' domains, which attended the evaluation process

4. After conducting the case study with project managers and members, I had to analyze and interpret the outcome. I divided the results into the classification of functional, usability and general output. This was the most consistent terminology during the discussions/case studies; issues corresponding to the functionality of the prototype, the user interface and usability and just general remarks concerning the prototype and the idea of the work.

5. My last step of the evaluation process was to review my research question and contributions based on the outcome of the test persons. Honestly, I was surprised about the consistently positive reactions of the project managers and members and thus, about the acceptance of my proposed solution.

In order to enable a review of my work, I give a general view of the natural competency questions concerning the scenarios and how they are solved by using SPARQL queries. I decided to put these work to the evaluation chapter due to the fact that the queries are the outcome of the predefined scenarios. They highlight the ontology results. Thus, provide the evaluation of my work.

Scenario 1

The description of the real use case is as follows:

The employee "Birgit Dippelreiter" is working in the project "SemProM" (Semantic based Project Management). In July she wants to go on holidays for two weeks. But during this month the evaluation of the SemProM prototype has to be finished. Is Birgit able to go on vacation?

To enable a better use of to the prototype and ensure an understanding of the composition of the queries, I divided scenario 1 into six steps. Following these steps each test person could execute easier the queries and interprete their results.

The six steps of scenario 1 deal with the following topics:

- Step 1 queries and outputs the tasks and their status Birgit Dippelreiter is responsible for as well as related tasks.
- Step 2 highlights the tasks and their status Birgit Dippelreiter is working in as well as related tasks.
- Step 3 displays employees with similar competences as Birgit Dippelreiter and the tasks and projects they are working in.
- Step 4 is similar to process 3. Beyond that the number of employees is restricted to those, who are working in the same projects and tasks as Birgit.
- Step 5 displays projects and tasks, which end dates are between a predefined time span.
- Step 6 is similar to process 5. In addition, it outlines employees with similar competences as Birgit and their tasks and projects during the predefined time span.

These steps contain, for instance, these informal natural language questions:

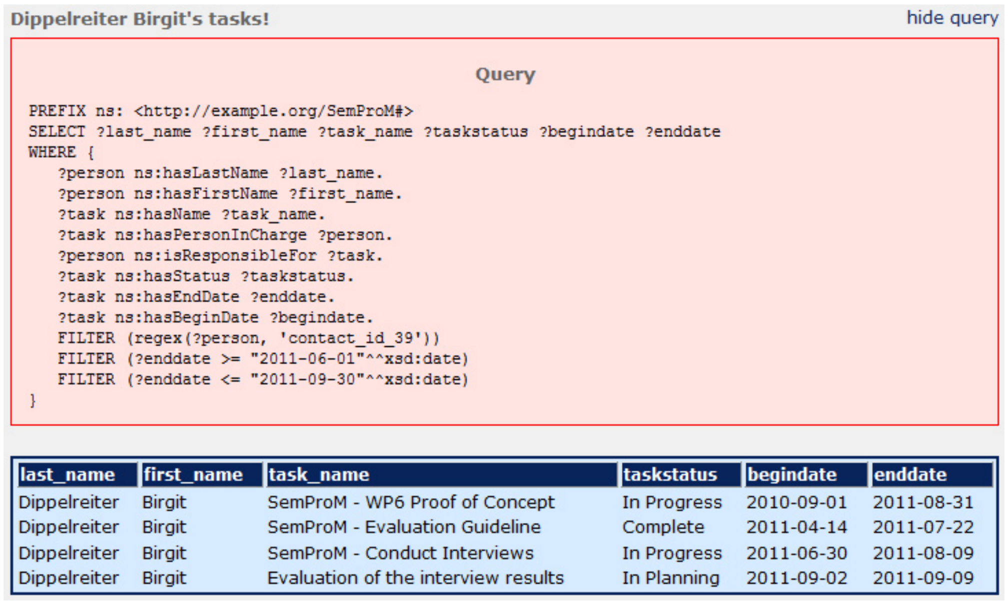
1. What are the projects and tasks "Birgit Dippelreiter" is currently responsible for?
   What is the process status of the projects/tasks?
   What is the planned end date of them?
2. Does the company have employees with the same competences as "Birgit Dippelreiter"?
   Do they work in tasks and projects "Birgit Dippelreiter" is currently responsible for?
   What is their function within the tasks and projects?
   What is the time span of the tasks and projects?
   What is the percentage allocation of the employee to the task/project he/she is working in?
   Does the deliverable have any dependencies to other deliverables, tasks or work packages?
3. Does the company have employees with the same competences as "Birgit Dippelreiter"?
   Are they responsible for tasks and projects "Birgit Dippelreiter" is working in?
   What is the time span of the tasks and projects?

These questions, "translated" to SPARQL queries, are shown in the following.

SPARQL Queries - Scenario 1

Figure 7.1 highlights the SPARQL query and the related output of question 1. The query requests all projects and tasks "Birgit Dippelreiter" is responsible for within a predefined end date period. The output lists all projects/tasks, including their name, status, begin and end date.

By contrast figure 7.2 outlines the request concerning employees with similar competences as "Birgit Dippelreiter", who are working in projects and tasks "Birgit Dippelreiter" is responsible for. In addition, the SPARQL query returns information of the function the employees are working in the task or project, their begin and end date and the percentage allocation of the



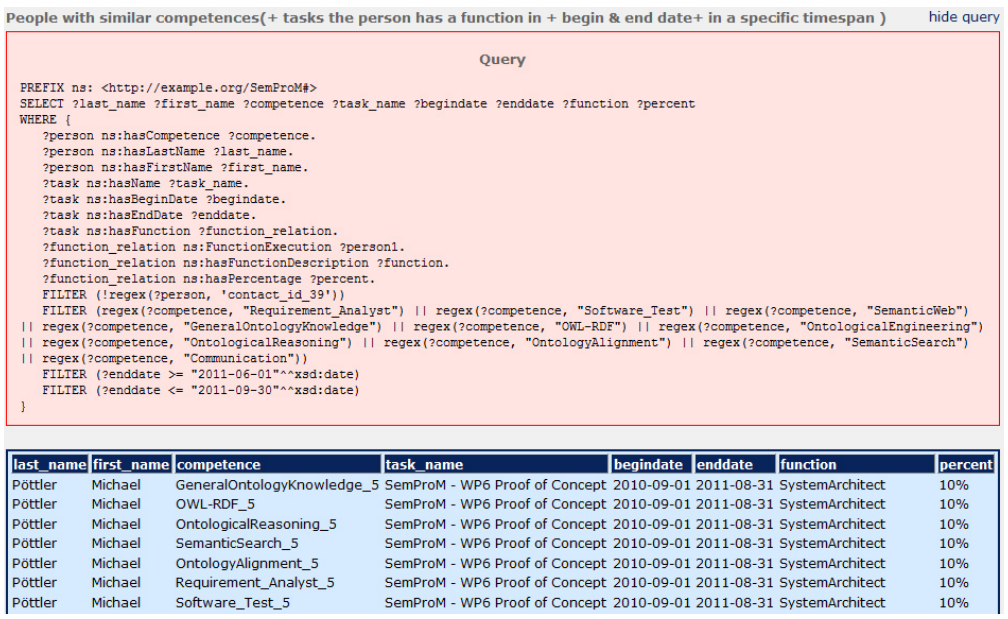
Dippelreiter Birgit's tasks! hide query

Query

```
PREFIX ns: <http://example.org/SemProM#>
SELECT ?last_name ?first_name ?task_name ?taskstatus ?begindate ?enddate
WHERE {
   ?person ns:hasLastName ?last_name.
   ?person ns:hasFirstName ?first_name.
   ?task ns:hasName ?task_name.
   ?task ns:hasPersonInCharge ?person.
   ?person ns:isResponsibleFor ?task.
   ?task ns:hasStatus ?taskstatus.
   ?task ns:hasEndDate ?enddate.
   ?task ns:hasBeginDate ?begindate.
   FILTER (regex(?person, 'contact_id_39'))
   FILTER (?enddate >= "2011-06-01"^^xsd:date)
   FILTER (?enddate <= "2011-09-30"^^xsd:date)
}
```

last_name	first_name	task_name	taskstatus	begindate	enddate
Dippelreiter	Birgit	SemProM - WP6 Proof of Concept	In Progress	2010-09-01	2011-08-31
Dippelreiter	Birgit	SemProM - Evaluation Guideline	Complete	2011-04-14	2011-07-22
Dippelreiter	Birgit	SemProM - Conduct Interviews	In Progress	2011-06-30	2011-08-09
Dippelreiter	Birgit	Evaluation of the interview results	In Planning	2011-09-02	2011-09-09

Figure 7.1: SPARQL query and results of informal competency questions of scenario 1 (Part 1 of 3)



People with similar competences(+ tasks the person has a function in + begin & end date+ in a specific timespan) hide query

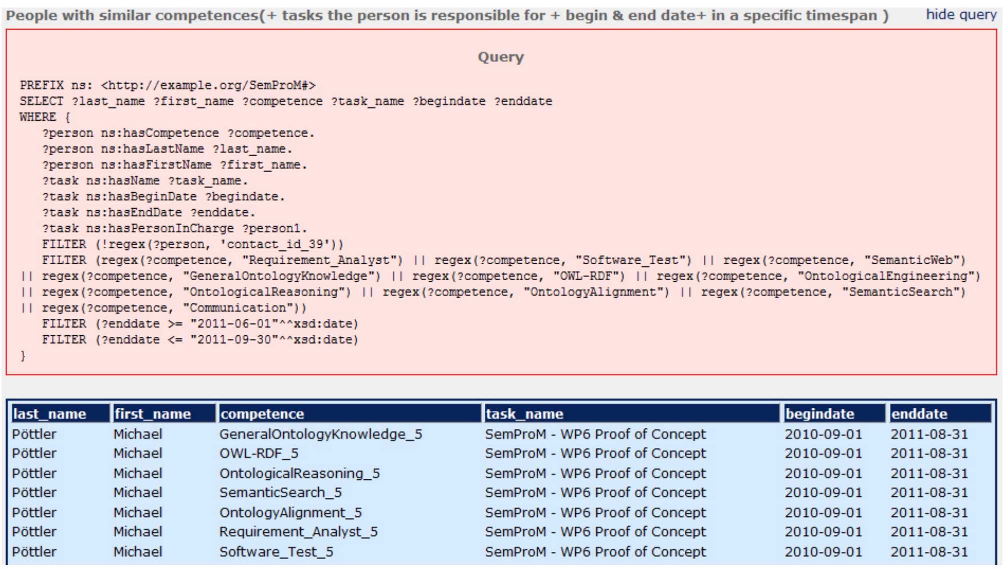
Query

```
PREFIX ns: <http://example.org/SemProM#>
SELECT ?last_name ?first_name ?competence ?task_name ?begindate ?enddate ?function ?percent
WHERE {
   ?person ns:hasCompetence ?competence.
   ?person ns:hasLastName ?last_name.
   ?person ns:hasFirstName ?first_name.
   ?task ns:hasName ?task_name.
   ?task ns:hasBeginDate ?begindate.
   ?task ns:hasEndDate ?enddate.
   ?task ns:hasFunction ?function_relation.
   ?function_relation ns:FunctionExecution ?person1.
   ?function_relation ns:hasFunctionDescription ?function.
   ?function_relation ns:hasPercentage ?percent.
   FILTER (!regex(?person, 'contact_id_39'))
   FILTER (regex(?competence, "Requirement_Analyst") || regex(?competence, "Software_Test") || regex(?competence, "SemanticWeb")
|| regex(?competence, "GeneralOntologyKnowledge") || regex(?competence, "OWL-RDF") || regex(?competence, "OntologicalEngineering")
|| regex(?competence, "OntologicalReasoning") || regex(?competence, "OntologyAlignment") || regex(?competence, "SemanticSearch")
|| regex(?competence, "Communication"))
   FILTER (?enddate >= "2011-06-01"^^xsd:date)
   FILTER (?enddate <= "2011-09-30"^^xsd:date)
}
```

last_name	first_name	competence	task_name	begindate	enddate	function	percent
Pöttler	Michael	GeneralOntologyKnowledge_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31	SystemArchitect	10%
Pöttler	Michael	OWL-RDF_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31	SystemArchitect	10%
Pöttler	Michael	OntologicalReasoning_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31	SystemArchitect	10%
Pöttler	Michael	SemanticSearch_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31	SystemArchitect	10%
Pöttler	Michael	OntologyAlignment_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31	SystemArchitect	10%
Pöttler	Michael	Requirement_Analyst_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31	SystemArchitect	10%
Pöttler	Michael	Software_Test_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31	SystemArchitect	10%

Figure 7.2: SPARQL query and results of informal competency questions of scenario 1 (Part 2 of 3)

employees to the task/project.


People with similar competences(+ tasks the person is responsible for + begin & end date+ in a specific timespan) hide query

Query

```
PREFIX ns: <http://example.org/SemProM#>
SELECT ?last_name ?first_name ?competence ?task_name ?begindate ?enddate
WHERE {
   ?person ns:hasCompetence ?competence.
   ?person ns:hasLastName ?last_name.
   ?person ns:hasFirstName ?first_name.
   ?task ns:hasName ?task_name.
   ?task ns:hasBeginDate ?begindate.
   ?task ns:hasEndDate ?enddate.
   ?task ns:hasPersonInCharge ?person1.
   FILTER (!regex(?person, 'contact_id_39'))
   FILTER (regex(?competence, "Requirement_Analyst") || regex(?competence, "Software_Test") || regex(?competence, "SemanticWeb")
|| regex(?competence, "GeneralOntologyKnowledge") || regex(?competence, "OWL-RDF") || regex(?competence, "OntologicalEngineering")
|| regex(?competence, "OntologicalReasoning") || regex(?competence, "OntologyAlignment") || regex(?competence, "SemanticSearch")
|| regex(?competence, "Communication"))
   FILTER (?enddate >= "2011-06-01"^^xsd:date)
   FILTER (?enddate <= "2011-09-30"^^xsd:date)
}
```

last_name	first_name	competence	task_name	begindate	enddate
Pöttler	Michael	GeneralOntologyKnowledge_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31
Pöttler	Michael	OWL-RDF_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31
Pöttler	Michael	OntologicalReasoning_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31
Pöttler	Michael	SemanticSearch_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31
Pöttler	Michael	OntologyAlignment_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31
Pöttler	Michael	Requirement_Analyst_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31
Pöttler	Michael	Software_Test_5	SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31

Figure 7.3: SPARQL query and results of informal competency questions of scenario 1 (Part 3 of 3)

The last example of a SPARQL query regarding scenario 1 is shown in figure 7.3. The query displays employees with similar competences as “Birgit Dippelreiter”; similar to the request above. In contrast, it outlines employees, who are responsible for a task or project and “Birgit Dippelreiter” is assigned to a function within the project or task. The request shows the first and last name of the employee; his/her competences, the task/project the person is responsible for and the begin and end date of the project or task.

Scenario 2 & 3

Scenario 2 “A new Project” and 3 “Team suggestions, etc.” are merged to one use case, because they engage one another. Scenario 3 extends scenario 2. Thus, the description of the real use case is as follows:

The project manager “Birgit Dippelreiter” of the project “Semantic based Project Management” inserts the new task “Evaluation of the interview results”.

Informal competency questions of use case 2 and 3 are for instance:

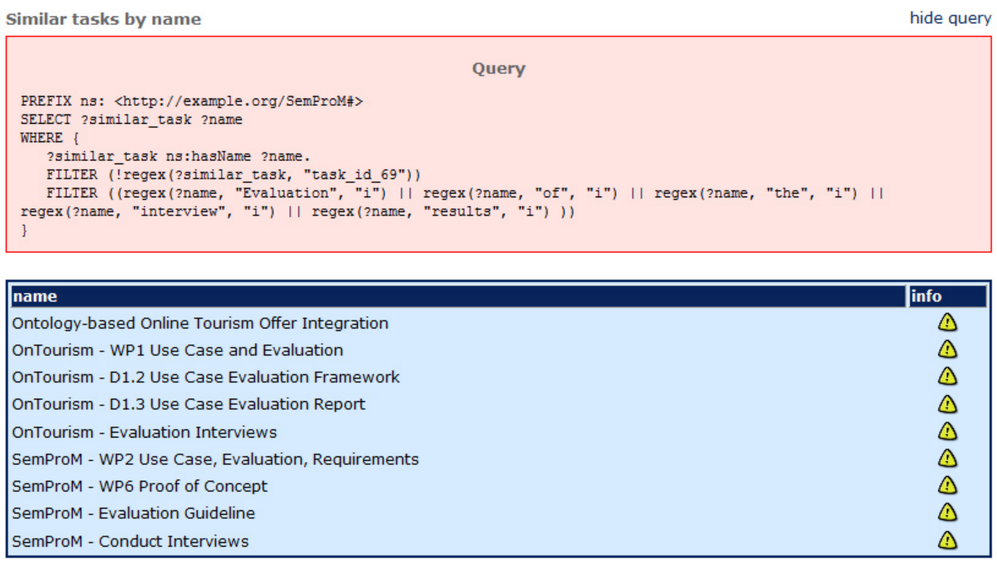
1. What are similar tasks or projects to the just stored project/task?
2. What are similar tasks or projects by assumed competences?
3. Are there alrady lessons learned, which arised out of similar projects?

4. Are there employees with fitting competences, the project/task requires?

5. Do the chosen employees work well with others? What are their competences?

SPARQL Queries - Scenario 2 & 3

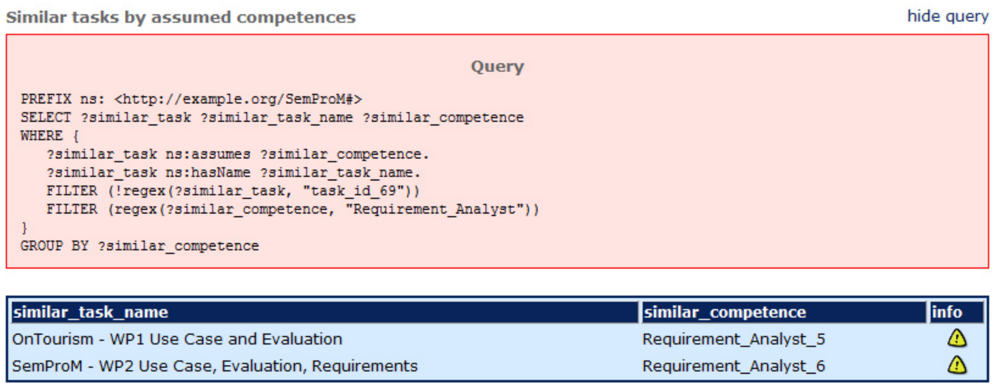
The translation of the informal competency questions to the formal ones of use case 2 and 3 are depicted in the figures 7.4 to 7.7.



```
PREFIX ns: <http://example.org/SemProM#>
SELECT ?similar_task ?name
WHERE {
   ?similar_task ns:hasName ?name.
   FILTER (!regex(?similar_task, "task_id_69"))
   FILTER ((regex(?name, "Evaluation", "i") || regex(?name, "of", "i") || regex(?name, "the", "i") ||
regex(?name, "interview", "i") || regex(?name, "results", "i") ))
}
```

name	info
Ontology-based Online Tourism Offer Integration	⚠
OnTourism - WP1 Use Case and Evaluation	⚠
OnTourism - D1.2 Use Case Evaluation Framework	⚠
OnTourism - D1.3 Use Case Evaluation Report	⚠
OnTourism - Evaluation Interviews	⚠
SemProM - WP2 Use Case, Evaluation, Requirements	⚠
SemProM - WP6 Proof of Concept	⚠
SemProM - Evaluation Guideline	⚠
SemProM - Conduct Interviews	⚠

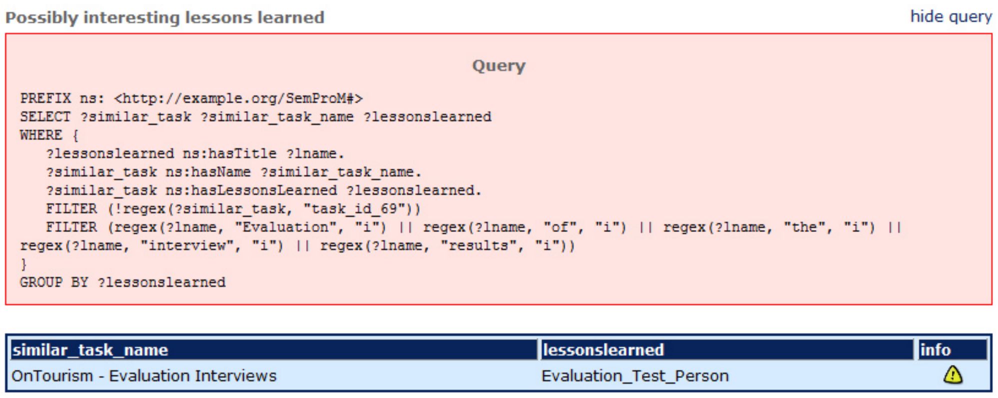
Figure 7.4: SPARQL query and results of informal competency questions of scenario 2 & 3 (Part 1 of 5)



```
PREFIX ns: <http://example.org/SemProM#>
SELECT ?similar_task ?similar_task_name ?similar_competence
WHERE {
   ?similar_task ns:assumes ?similar_competence.
   ?similar_task ns:hasName ?similar_task_name.
   FILTER (!regex(?similar_task, "task_id_69"))
   FILTER (regex(?similar_competence, "Requirement_Analyst"))
}
GROUP BY ?similar_competence
```

similar_task_name	similar_competence	info
OnTourism - WP1 Use Case and Evaluation	Requirement_Analyst_5	⚠
SemProM - WP2 Use Case, Evaluation, Requirements	Requirement_Analyst_6	⚠

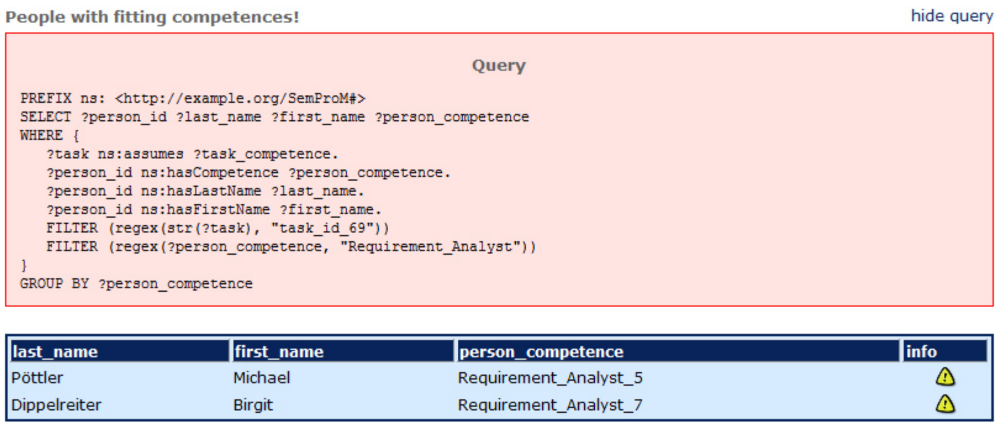
Figure 7.5: SPARQL query and results of informal competency questions of scenario 2 & 3 (Part 2 of 5)



```
PREFIX ns: <http://example.org/SemProM#>
SELECT ?similar_task ?similar_task_name ?lessonslearned
WHERE {
   ?lessonslearned ns:hasTitle ?lname.
   ?similar_task ns:hasName ?similar_task_name.
   ?similar_task ns:hasLessonsLearned ?lessonslearned.
   FILTER (!regex(?similar_task, "task_id_69"))
   FILTER (regex(?lname, "Evaluation", "i") || regex(?lname, "of", "i") || regex(?lname, "the", "i") ||
regex(?lname, "interview", "i") || regex(?lname, "results", "i"))
}
GROUP BY ?lessonslearned
```

similar_task_name	lessonslearned	info
OnTourism - Evaluation Interviews	Evaluation_Test_Person	⚠

Figure 7.6: SPARQL query and results of informal competency questions of scenario 2 & 3 (Part 3 of 5)



```
PREFIX ns: <http://example.org/SemProM#>
SELECT ?person_id ?last_name ?first_name ?person_competence
WHERE {
   ?task ns:assumes ?task_competence.
   ?person_id ns:hasCompetence ?person_competence.
   ?person_id ns:hasLastName ?last_name.
   ?person_id ns:hasFirstName ?first_name.
   FILTER (regex(str(?task), "task_id_69"))
   FILTER (regex(?person_competence, "Requirement_Analyst"))
}
GROUP BY ?person_competence
```

last_name	first_name	person_competence	info
Pöttler	Michael	Requirement_Analyst_5	⚠
Dippelreiter	Birgit	Requirement_Analyst_7	⚠

Figure 7.7: SPARQL query and results of informal competency questions of scenario 2 & 3 (Part 4 of 5)

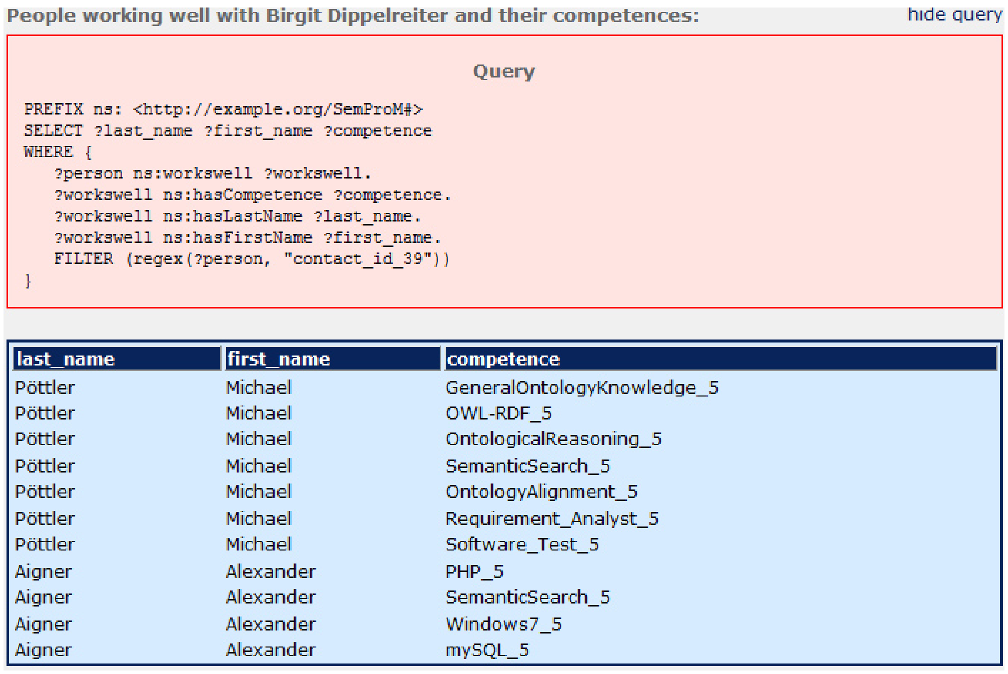
Query one (figure 7.4) selects tasks and projects, which are similar to a just stored project/task by name. Up to now, it is just a comparison of words in the project/task title. The outcome is a list of projects/tasks. For more information regarding a project/task, the user has to click the triangle beside the listed items.

Figure 7.5 covers the question whether there are already stored projects or tasks, which assumes the same or similar competences. The number beside the required competence indicates the level of the skill. 1 is the lowest, while 10 the highest.

The request if there are already lessons learned available, which would improve the new project or task, is handled in figure 7.6. To obtain more information, the triangle beside the listed items has to be clicked.

Figure 7.7 displays employees with fitting competences as the project or task assumes. As already mentioned above, the number beside the required competence indicates the level of the skill. (1 is the lowest, while 10 the highest.)

The last question whether an assigned employee (to a task) works well with others, who they are and what their competences are is illsutrated in figure 7.8. The outcome is a list with the employees and their competences.



People working well with Birgit Dippelreiter and their competences: hide query

Query

```
PREFIX ns: <http://example.org/SemProM#>
SELECT ?last_name ?first_name ?competence
WHERE {
   ?person ns:workswell ?workswell.
   ?workswell ns:hasCompetence ?competence.
   ?workswell ns:hasLastName ?last_name.
   ?workswell ns:hasFirstName ?first_name.
   FILTER (regex(?person, "contact_id_39"))
}
```

last_name	first_name	competence
Pöttler	Michael	GeneralOntologyKnowledge_5
Pöttler	Michael	OWL-RDF_5
Pöttler	Michael	OntologicalReasoning_5
Pöttler	Michael	SemanticSearch_5
Pöttler	Michael	OntologyAlignment_5
Pöttler	Michael	Requirement_Analyst_5
Pöttler	Michael	Software_Test_5
Aigner	Alexander	PHP_5
Aigner	Alexander	SemanticSearch_5
Aigner	Alexander	Windows7_5
Aigner	Alexander	mySQL_5

Figure 7.8: SPARQL query and results of informal competency questions of scenario 2 & 3 (Part 5 of 5)

7.2 Evaluation Results

As already mentioned, I carried out the prototype evaluation with six test persons. Table 7.2 and 7.3 highlight the main evaluation results concerning scenario 1 “Bob needs Holidays”. In addition, table 7.4 outlines the results regarding scenario 2 “A new Project” and scenario 3 “Team suggestions, etc.”.

For discussing the results, I summarized the results into the clusters functionality, usability/user interface and general.

Scenario 1 “Bob needs Holidays“			
Person	**Usability**	**Functionality**	**General**
Test person 1	•Part 1: Identification of relations between tasks in different tables (e.g., predecessor or successor) is difficult ➔ Indexation of tasks •Part 3: ranking of the hits would make sense; e.g., by degree of accordance to others or by availability	•Part 1: Task status: possibility to select the type of status should be given •Part 3: request is precise and clearly •Part 3: Numeric relation of the tasks would make sense	•Part 5: assignment of a new task to a person is easier than of an existing, ongoing task to a person •Requests are easy and simple; intuitive •Test person needs 1-2 requests to understand the system •Conceptually conventions takes some getting used to •Integrable into other systems
Test person 2	•Part 1: Task status: in percentage •Part 1: Relation of the tasks with graphic arrows? •Part 3: Listing of the output should be classified/bundled; Person ➔ Task ➔ Competences •Listing of the Competences of the “wanted” person would make sense •Part 6: classification of the results is necessary •Concrete recommendation of a possible person would make sense; who can do the work of another one	•Part 1: Task status: possibility to select the type of status should be given •Part 2: put together “responsible for” and “working in” •Part 2: what happens if a task is not finished by the predefined end date and the related persons starts with a new task? Who is doing the “old” task? •“show people with similar competences in same task” ➔ makes sense	•Positive: listing of the competences of the persons •Scenario 1 is well known; same problem in the company •Nice idea •Part 5: practical •Scenario fits perfect to the real world •Test person would like to use the system within his/her company
Test person 3		•Part 1: “Show related Tasks: “all” is missing •Part 3: Tasks should be ongoing tasks; no “in archive” or “finished” •How much does a person work in different tasks ➔ percentage output	•Questions are answered by the search functionality •Output is chaotic; lost of information in a small area ➔ graphic design •Positive: output of persons, who have similar competences

Table 7.2: Evaluation results concerning scenario 1 “Bob needs Holidays” (Part 1 of 2)

Functionality

One test person told me that they currently have a similar problem concerning team members are going on vacation and who can substitute them.

Another response regarding scenario 1 was to take holidays of other employees into account. In addition, to get more precise results of possible substitutions.

Regarding the improvement of the search function, a selection of the project/task status was

Scenario 1 “Bob needs Holidays“			
Person	**Usability**	**Functionality**	**General**
Test person 4	•Part 1: graphic preparation of the output is missing •Part 3: what are the competences of the “wanted” person? ➔ output is missing •Part 3: miss the percentage relation of a person to a task •Part 4: miss to define the degree of competence •Part 5: date input field; possibility to write the date manually is missing	•Part 1: Select type of status is missing; “archived” is redundant •Part 2: why do we separate “responsible for” and “working in”? ; PM Austria defines it as one modul •Part 2: Tasks: unique identifier ➔ relation of tasks is easier detectable •Part 2: system is not able to identify what kind of dependencies between tasks is given	•Part 1: definition of the notion •User interface and the graphic preparation of the output must be improved •Listing of competences of employees is only useful within a big company •System just usable if the data administration is working •Resource management just usable if data administration is working and the information is complete
Test person 5	•Part 2: Sum um of all tasks; “responsible for” as well as “working in” ➔ within one table •Part 2: Listing: Task ➔ Function ➔ ... •Part 3: Listing of persons; click a person if you want to know more details about this person •Part 3: “Show task status” can’t be re-activated •Part 3: “Show” button should also be at the end of the user interface •Part 5: date input field; possibility to write the date manually is missing •Part 6: classification of the results is necessary	•Part 5: “responsible for” vs. “working in”: both is relevant •Part 6: holidays of others should also be taken into account	•Part 2: What is the benefit of a predecessor of a task if you want to answer the question if a person can go on holidays? •User interface and the graphic preparation of the output must be improved •Neat arrangement of the information is necessary ➔ Person X has Competences within this tasks
Test person 6	•Part 3: radio buttons can’t be re-activated •Part 6: listing of information is not meaningful	•Part 6:what is the workload of person X concerning all tasks •Part 6: output of all tasks, which start, are already ongoing or closing within a predefined time period; currently the output concerns just tasks, which have their end date within a predefined time period •Who of the persons, who are available, has the right competences	•Intuitive user guidance and usability are poor •Function: search for a substitute of Birgit within the time period X; who is available? Who has the right competences? •Ranking is missing: who has the most similar competences to Birgit? •Scenario 1 is in need of improvement

Table 7.3: Evaluation results concerning scenario 1 “Bob needs Holidays” (Part 2 of 2)

Scenario 2 & 3 - “A new Project“ & “Team suggestions, etc.”			
Person	**Usability**	**Functionality**	**General**
Test person 1	•“Responsible for” should be shown in the first table •Preparation of the output can be improved •Information should be shown graphical	•Listing of the information can be refined; some additional information to select •Access authorization •Interfaces to other systems, like budgetary information, planning or resource management	•Intuitive •Lots of information •Positive: qualification/ competences and relations •Who is collecting the information
Test person 2		•Team building: useful; normally, just the team leader knows who is working well together and who has what competences	•Positive: social skills; work well together •Positive: lessons learned •Positive: social level
Test person 3	•Nice to have: picture of the person •Usability difficult; lots of tables •Positive: each table has the same structure •Negative: heading quite technical	•Personal data are missing; contact information (work-related/official)	•good idea; implementation quite prototypically •Wording: competence = qualification •Would be cool to have such a system in our company
Test person 4	•Percentage information of the workload of a person within a task is missing	•Persons with a workload of 100% should not be shown as possible substitute or team member •Relations between different information items is useful	•Positive: lessons learned •Interesting within big companies
Test person 5	•Buttons have no standardized position •Listing of persons and their availability	•Priority of projects and tasks is missing •Just ongoing tasks should be shown •Separation of archived and ongoing tasks •If persons cannot work together, they should not be listed as possible team members •Search for projects is missing	•System usable: it is an efficient way to figure out who is applicable for what task •Simple usability •Negative: presentation of the output •Negative: too much information at once
Test person 6	•Sorting of tasks should be improved; chronological process	•Description of tasks is missing	•Presentation of the output must be improved •Scenario 2 & 3 really good

Table 7.4: Evaluation results concerning scenario 2 “A new Project” and scenario 3 “Team suggestions, etc.”

required. Furthermore, a percentage output of the workload of a person within a task was requested. Both adapations are already implemented.

Concerning scenario 2 & 3 all test persons were positive about the abundance of information.

Of course some critical aspects were also mentioned. For instance, official contact information of persons or the priority of projects and tasks is missing. In addition, persons with a workload of 100% should not be listed as possible team member or substitute. In addition, a total workload of all projects/tasks a proposed person is working in would be interesting.

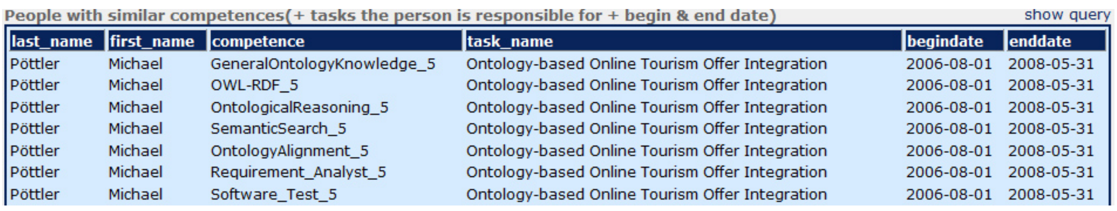
Positive remarks are that the relations between different information items are useful and necessary or the output for possible team building is really interesting and applicable.

The search functionality enables to query tasks/projects a person is "responsible for" or "working in". Some test persons mentioned that it should be possible to query for both at once and not one or the other. This remark is already implemented.

Another notation is that a ranking is missing to identify the persons with the most similar competences to a predefined person.

User interface/Usability

Regarding the preparation of the information output all test persons considered that this part must be improved. Currently, the information is listed in different tables. Thus, it complicates to detect on the one hand, the relation between different information items, and on the other hand, the main information. A feasible solution is to illustrate the information graphically and aggregated. A concrete example is the screenshot in figure 7.9.


People with similar competences(+ tasks the person is responsible for + begin & end date) show query

last_name	first_name	competence	task_name	begindate	enddate
Pöttler	Michael	GeneralOntologyKnowledge_5	Ontology-based Online Tourism Offer Integration	2006-08-01	2008-05-31
Pöttler	Michael	OWL-RDF_5	Ontology-based Online Tourism Offer Integration	2006-08-01	2008-05-31
Pöttler	Michael	OntologicalReasoning_5	Ontology-based Online Tourism Offer Integration	2006-08-01	2008-05-31
Pöttler	Michael	SemanticSearch_5	Ontology-based Online Tourism Offer Integration	2006-08-01	2008-05-31
Pöttler	Michael	OntologyAlignment_5	Ontology-based Online Tourism Offer Integration	2006-08-01	2008-05-31
Pöttler	Michael	Requirement_Analyst_5	Ontology-based Online Tourism Offer Integration	2006-08-01	2008-05-31
Pöttler	Michael	Software_Test_5	Ontology-based Online Tourism Offer Integration	2006-08-01	2008-05-31

Figure 7.9: Screenshot of the person Michael Pöttler, his competences and the project he is responsible for

It displays people with similar competences. In this special case, competences of the person "Michael Pöttler" as well as the task he is responsible for and its begin and end dates are listed. A simple improvement is to aggregate these seven lines to one single line. In addition, the information is: the person "Michael Pöttler" is responsible for the task "Ontology-based Online Tourism Offer Integration" with the begin date "2006-08-01" and the end date "2008-05-31". His competences are “GeneralOntologyKnowledge_5, OWL-RDF_5, OntologicalReasoning_5, Semantic-Search_5, OntologyAlignment_5, Requirement_Analyst_5 and Software_Test_5”. (The number beside the required competence indicates the level of the skill. 1 is the lowest, while 10 the highest.) All these competences will be aggregated into one line and column.

Another point of criticism is the notation of the terms. Up-to-date, they are quite technical. In the future, they have to be translated into a common notation.

Some test persons compared the prototype and its functionalities with MS Project. Thus, they critized the usability a lot. In addition, they reviewed the categorization of tasks and task status and mentioned that the intuitive user guidance and usability are poor.

When programming the prototype my focus was on implementing the functionalities, including the ontology and ontological reasoning and thus, I simply forgot to take care of the usability. As a result, the test persons were completely right to criticize the user interface and usability of the prototype.

General

The main statement of the test persons was that the functionality of the prototype is really interesting, useful and makes sense. Most of them were surprised of the quantity of information items and the relations between them. Some of them mentioned that they would like to have such a system in their company. Further positive comments are that the connection of competences, tasks and persons is efficient or that it is an efficient way to figure out who is applicable for which task. Moreover, the relations to lessons learned is really practicable.

Another remark was that the information about competences are only necessary if the company is a big one. In addition, information regarding competences has to be up-to-date and complete. Otherwise, employee proposals and team allocation do not make any sense.

Of course all of them also criticised the usability and preparation of the information output. Even so, my main focus during the implementation of the prototype was the functionality of the system and the relation of the information (ontology and ontological reasoning). One also mentioned that she/he gets too much information at once.

7.3 Answering the Research Question and the Contributions

With these evaluation results, I am able to review the research question and contributions. As discussed in the introduction, the research question and contributions are as follows:

Research Question:

"Do semantic technologies improve current project management processes?"

Contribution 1:

Ontologies and ontological reasoning enable a better management of assignment of personnel to tasks as well as of competences, personnel and tasks.

Contribution 2:

With the use of semantic technologies, possible upcoming problems can be fixed by using knowledge of finished projects in advance. In addition, it reduces time and cost in fact of reusing knowledge. Thus, if up-to-date information of ongoing projects is shown in a kind of cockpit, project managers and members are able to deal in time with upcoming problems.

Contribution 3:

Ontological description of project related and relevant information (of completed, archived and current projects) improves the search functionality in order to recover and reuse knowledge of finished projects, especially to identify related and similar information. With the use of an ontology and ontological description a project management system is becoming a knowledge base in the scope of project management.

Contribution 4:

Ontologies and ontological reasoning improve the start and end phases of a project management life cycle.

Contribution 1

Ontologies and ontological reasoning enable a better management of assignment of personnel to tasks as well as of competences, personnel and tasks.

This contribution is covered by the scenario 1 "Bob needs Holidays" and scenario 3 "Team suggestions, etc.".

This contribution is validated. Due to the fact that the search functionality in scenario 1 depicts relations between information, the outcome concerning vacation replacement suggestions is improved. Furthermore, detailed information of possible holiday substitutions is given. For instance, their work load and competences.

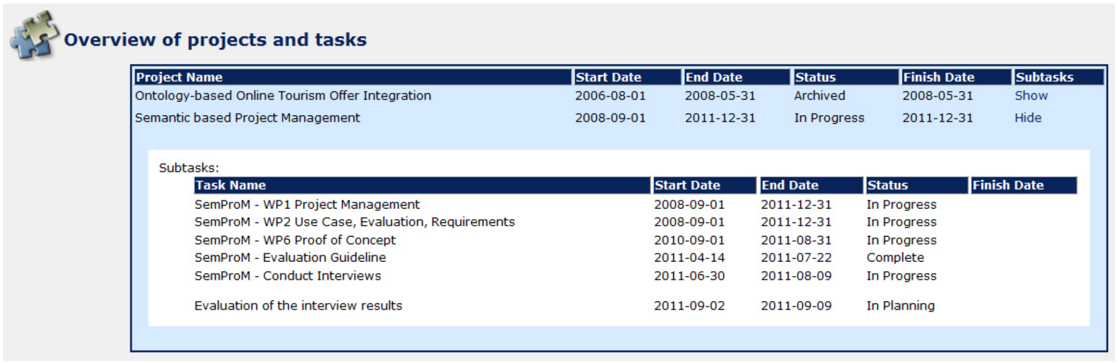
Some improvement proposals concerning scenario 1 have also be mentioned. While doing the evaluation of the prototype, I as well as the test persons realized that the scenario "Bob needs Holidays" should be extended. It should be enhanced by the information of the vacations of other employees. In addition, knowledge of other systems, like sick calls or resignation of a job, should also be integrated. Nevertheless, scenario 1 validates contribution 1.

Scenario 3 highlights the feasible power of the prototype. The suggestions of, e.g., similar projects/tasks and possible team members displays the relations of information and the power of a semantic knowledge base. In addition, the simplicity of information retrieval and information reuse is shown. Thus, scenario 3 also validates contribution 1.

Contribution 2

With the use of semantic technologies, possible upcoming problems can be fixed by using knowledge of finished projects in advance. In addition, it reduces time and cost in fact of reusing knowledge. Thus, if up-to-date information of ongoing projects is shown in a kind of cockpit, project managers and members are able to deal in time with upcoming problems.

All three scenarios ("Bob needs Holidays", "A new Project" and "Team suggestions, etc.") deal with contribution 2. In addition, while implementing the prototype a "cockpit" of ongoing projects was put into practice. This cockpit displays all ongoing projects and their tasks; their planned end dates and if they are in or out of time; displayed in figure 7.10. Of course, the cockpit also has to be improved regarding the usability. Nevertheless, for a first draft this version proved to be sufficient.



Overview of projects and tasks

Project Name	Start Date	End Date	Status	Finish Date	Subtasks
Ontology-based Online Tourism Offer Integration	2006-08-01	2008-05-31	Archived	2008-05-31	Show
Semantic based Project Management	2008-09-01	2011-12-31	In Progress	2011-12-31	Hide

Subtasks:

Task Name	Start Date	End Date	Status	Finish Date
SemProM - WP1 Project Management	2008-09-01	2011-12-31	In Progress	
SemProM - WP2 Use Case, Evaluation, Requirements	2008-09-01	2011-12-31	In Progress	
SemProM - WP6 Proof of Concept	2010-09-01	2011-08-31	In Progress	
SemProM - Evaluation Guideline	2011-04-14	2011-07-22	Complete	
SemProM - Conduct Interviews	2011-06-30	2011-08-09	In Progress	
Evaluation of the interview results	2011-09-02	2011-09-09	In Planning	

Figure 7.10: Screenshot: Cockpit - Overview of projects and their tasks

Although several interview persons mentioned that they need a kind of "cockpit" in their project management systems to identify awaking problems, I realized that certain available PM systems already support these feature. Nevertheless, with the use of all three scenarios and adding the cockpit to answer this contribution, the contribution is validated only partly.

This is, because scenario 1 highlights current information of ongoing projects. For instance, relations of employees to projects and their workload within the projects. In addition, information of employees and their ability to work in a team is also highlighted.

By contrast, scenario 2 and 3 outline information retrieval concerning the improvement of starting and ongoing projects. They offer information of similar projects and tasks, suggestions regarding project teams, etc.

However, on the basis of the time factor I was not able to collect enough information of finished projects to review if reusing knowledge minimizes upcoming problems as well as time and cost. Thus, this contribution is just validated partly.

Contribution 3

Ontological description of project related and relevant information (of completed, archived and current projects) improves the search functionality in order to recover and reuse knowledge of finished projects, especially to identify related and similar information. With the use of an ontology and ontological description a project management system is becoming a knowledge base in the scope of project management.

Contribution 3 is covered by scenario 2 “A new Project” and 3 “Team suggestions, etc.”.

Both scenarios deal with knowledge of finished and ongoing projects and how these knowledge is usable for new projects and/or tasks. They offer different kinds of information, such as documents or lessons learned; in addition, knowledge of each employee, like her/his competences, working relationships to other employees or her/his assignments to projects and tasks and her/his workload.

Storing a new project in the PM system, the system checks all similarities to existing knowledge. Thus, similar and related information is identified and displayed to the project manager. For this reason, the project manager is able to reuse knowledge in order to support the new project. Thereby, contribution 3 is validated.

Contribution 4

Ontologies and ontological reasoning improve the start and end phase of a project management life cycle.

As already mentioned above, the ongoing phase of a project management life cycle is already covered by existing PM systems. Whereas the start and end phase of a PM life cycle are treated more or less like “stepchildren” by current systems. For instance, knowledge of finished projects is not stored centrally or lessons learned are not specified.

My prototype covers the start and end phase of a PM life cycle. As a result, knowledge of finished projects is stored centrally and structured as well as related to other information. In addition, this knowledge is used to improve new projects and tasks by giving suggestions regarding, for instance, formation of project groups or lessons learned of similar projects. Scenario 2 "A new Project" and 3 "Team suggestions, etc." deal with the start and end phase of a project management life cycle.

For the same reason as in contribution 2, the time factor, this contribution is only validated partly. I did not have enough time to collect sufficient knowledge of projects and thus to check whether the starting phase of the PM life cycle is really getting improved. Nevertheless, I was able to claim that with the use of a smart end phase the start phase will be enhanced. This is, because of the SPARQL query results. They outline promising results.

Research Question

Regarding the research question whether *semantic technologies improve current project management processes*, I can answer this question positively. Semantic technologies improve project management, because with the use of semantic technologies a kind of knowledge base is built. As a result, information retrieval and reuse of existing knowledge is getting far easier.

My contributions show that semantic technologies enhances current project management. They outline that

- it simplifies information retrieval and reuse,
- semantic technologies improves everyday issues, like vacation replacement or sick leave cover and
- it enhances the start and end date of the project management life cycle.

A main difference to existing web based project management systems is, among others, the structured information and the relation to other stored information. In addition, the access to knowledge is easier.

CHAPTER 8

Conclusion & Outlook

8.1 Conclusion

In this thesis I deal with the question "whether semantic technologies improve current project management processes". To answer this question, I carried out the following work. First, I did a literature review about similar work concerning project management enhanced with semantic technologies, existing ontologies and methods, which can be used for my approach. Second, I conducted interviews to get an overview of current PM solutions as well as their weaknesses, strengths and standards. Third, an evaluation of open source project management systems was accomplished to analyze the capabilities of current solutions and to select a system for prototyping. The next step was the specification of scenarios. They highlight common shortcomings of current PM. In addition, they outline the requirements of the ontology as well as prototype. They were also carried out to evaluate the prototype and ontology and in further work to evaluate the research question and contributions. Fifthly, I designed a project management ontology containing aspects of, e.g., project management, competences of employees, lessons learned and document information. Sixthly, I implemented the prototype, containing the project management ontology and the APIs to insert and update instances within the ontology and to request instances. Last, I evaluated my approach using an evaluation guideline by evaluating the prototype and the ontology based on the scenarios.

I propose that semantic technologies are able to tackle the shortcomings of current project management. Especially improving the start and end phase of a project management life cycle. Therefore, I suggested the prototype called "Semantic based Project Management" - SemProM - which enables a better search functionality and reuse of already existing knowledge by using semantic technologies, reducing thus time and costs. In addition, it reduces the effort of project management and increases the probability of project success. information of already finished projects can be used as input for planning new projects and to monitor progress and risks of projects underway. That is, because of better storage of project relevant information. Due to explicitly describing the relationships between various project documents in machine accessible form, better administration of projects and easier identification of relevant information is possible.

While conducting the interviews with the companies, I realized that project management includes a wide range of the interpretation of the term “project management”. Some companies just use a MS Excel document for all relevant steps of project processes, others had predefined processes and self implemented systems to handle each kind of project. Another interesting point regarding the interview outcomes is that some interviewee noted that a kind of “cockpit” management would be necessary to survey all relevant information, such as milestones, important dates and costs of projects. A further outcome of the interviews is that most of the interviewed companies do not have a standardized archive and therefore the stored information documents cannot be reused as easily.

In addition to the interviews, a functional evaluation of project management systems was done. An interesting point is that the evaluated tools show large differences regarding the features while operating use cases. While dotProject and PHProjekt offer nearly the same functionalities and processes, WebCollab and Projectpier just provide the most needed features.

To implement a prototype dotProject seemed to be most suitable. It has similar functionalities like PHProjekt, but offers a better usability. Furthermore, it contains more features than WebCollab and Projectpier, especially in the field of project processes.

A similar approach is used by Semantic Desktop, which is a Personal Information Management system. The analogy to a project management system is given. While the Semantic Desktop uses semantic technologies, such as ontologies to link and store personal information for a better structuring and reusing of the information the project management system will use them for project related information.

Another issue of current project management solutions, validated by the interviews, is the data storage. Up to now lots of companies do not have a central data storage and/or no structured stored information to reuse it. If they use a central data storage and structure and annotate their stored information with the use of ontologies, they will get a knowledge base with nearly all information related to project management. For example, lessons learned, competences and documents, but all information related to projects, while they are related to other projects in fact of, e.g., similar lessons learned.

Based on the interviews, I developed three scenarios, “Bob needs Holidays”, “A new Project” and “Team suggestions, etc.”; used as requirements for a semantic based project management prototype, and additionally, used for evaluating the approach.

The scenarios outline the following shortcomings of current PM solutions:

- PM systems do not contain a central data storage.

- There exists no standardization how to store and annotate project relevant information.
- Project manager and members often miss an up-to-date information cockpit to get all relevant information of ongoing projects.
- Often, stored information cannot be retrieved, because it is not stored central and cannot be retrieved due to less annotations.

To tackle these problems and cover the scenarios, I extended the open source project management system dotProject with semantic technologies; a PM ontology, semantic annotations and ontological reasoning. The self developed project management ontology [6] represents the relations between projects, tasks, documents, employees and their competences and lessons learned.

While implementing the project management ontology I had to define the ontological engineering approach and to decide whether to use existing ontologies and merge them or to implement a new one. Relating the ontology engineering approach I discussed Uschold and King [70] and Grüninger and Fox [34]. As a result, I combined both approaches. By merging both, I obtain a detailed documentation of the ontology, its concepts, properties and relations, its term definitions and an illustration how to extend the ontology.

To decide whether to use existing ontologies or not, I carried out an in-depth study of existing ontologies, such as project management, document and time/date ontologies. Based on the required needs I decided to implement an individual PM ontology due to the fact that the evaluated ones are partially too general for our purpose.

When implementing the prototype, I had to connect the PM system dotProject and the project management ontology. Therefore, I required interfaces to store instances in the ontology and to perform queries. The first part was accomplished by the API RAP. Unfortunately, the further development of this API stopped, I had to carry out lots of changes and adaptations. In addition, I implemented a new API. Regarding the second part, querying, I decided to use the API ARC, which is still under development. To follow an approach during the implementation of the prototype, I started to store and update instances in the ontology. The next step was to adapt the existing user interface; and the last one was to implement new modules, including the functionality and user interface.

To evaluate the prototype and additionally the project management ontology, I carried out an evaluation guideline based on the scenarios. In addition, the evaluation was done with six test persons, working as project managers in companies. Most of them related to the domain of IT. These persons had to go through the evaluation guideline carrying out the predefined scenarios.

The feedback of the test persons were consistently positive concerning the functionalities, the search results and the suggestions for new projects. Regarding the usability and user interface some improvement suggestions were commented. With the evaluation results the research question and the contributions are also validated. Fact is, semantic technologies improve current project management systems. They enable a kind of project management knowledge base and simplify information retrieval and reuse.

8.2 Outlook

This work just outlines a small area of project management; its problems, strengths and weaknesses and how to improve project management.

As further work my approach should be tested in “real world” to gather enough information to show the “full” power.

When carrying out this project my main focus was on project management in the domain of IT. Thus, an opportunity for further development would be just another domain. For instance, projects in the domain of medicine or architecture. In this case, the project management ontology must be extended in the field of professional competences.

A further development of the ontology would be the extension of the competences in general; not only the professional competences. In my work, I just give an outline of the classification of competences and possible sub-categories. I do not provide detailed sub-categories and choices in each category. For instance, merging an existing competence ontology with the project management ontology would be a possible solution.

Another work, which would increase the improvement of current PM, is to extend the scenarios, but also to define new ones. As an example, the adaptation of scenario 1 “Bob needs Holidays” would be useful. To extend this scenario about the information of vacation planning of other employees in the company is an interesting and also important improvement. The added value of further information would be significant. In addition, a subsitute in case of illness will also be an interesting use case. This case is compoundable with the scenario “Bob needs Holidays”.

While evaluating the prototype most of the test persons mentioned that the user interface and the preparation of the output are not user-friendly. Thus, the user interface should be improved. The output preparation should be more structured and categorized as well as graphically redesigned.

Currently, the prototype just provide to edit and change instances of the ontology. A convenient adaptation would be to edit and change concepts of the ontology. Thus, the PM system will be extensible by itself.

Another interesting work would be to extend the project management ontology by dependencies between different projects and tasks. Up to now the ontology just covers relations of tasks within a project, such as subtask, predecessor or successor of another task. Of course, by querying through the ontology similar projects are found out and potential relationships can be identified; but up to now they are not directly related as “dependent”.

Up to now documents are just annotated with metadata and stored in the PM system. Text mining would improve the system significant; in further consequence, information retrieval and reuse. Text mining is the extraction of knowledge out of documents. By means of this the starting and closing phase of a project management life cycle will be significant improved. This is, because text mining prepares and analyzes unstructured data and tries to find patterns, rules and relations of information. Combined with the already implemented semantic technologies, the project management system will be improved further on.

Regarding the “cockpit”, up-to-date information of ongoing projects, this should also be improved and developed further. Up to now it is just a listing of ongoing projects, including the information of start and end date and the status of the projects. In further work, a kind of traffic light could be inserted to identify the status of the project. In this case, green would be “in time”, yellow would be “take care” and red would be “alarm”.

Bibliography

[1] *A Guide To The Project Management Body Of Knowledge (PMBOK Guides)*. Project Management Institute, 2004.

[2] *Research Methods for Leisure and Tourism: a Practical Guide*. Prentice Hall International, 2006.

[3] S. Abels, F. Ahlemann, A. Hahn, K. Hausmann, and J. Strickmann. Promont - a project management ontology as a reference for virtual project organizations. In Robert Meersman, Zahir Tari, and Pilar Herrero, editors, *OTM Workshops (1)*, volume 4277 of *Lecture Notes in Computer Science*, pages 813–823. Springer, 2006.

[4] F. Ahlemann and K. Backhaus. *Project Management Software Systems Requirements, Selection Process and Products*. OYGON Verlag, Munich, 4., edition edition, 2006.

[5] A. Aigner. Easy rdf for php (erp) api - a php api for processing rdf resources. Master's thesis, Technische Universitt Wien, Fakultät für Informatik, 2012.

[6] C.J. Anumba, J. Pan, R.R.A. Issa, and I. Mutis. Collaborative project information management in a semantic web environment. *Engineering, Construction and Architectural Management*, 15:78 – 94, 2008.

[7] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific Am.*, 284(4):34–43, 2001.

[8] E. Biesalski. *Unterstützung der Personalentwicklung mit ontologiebasiertem Kompetenzmanagement*. PhD thesis, Universität Karlsruhe, 2006.

[9] G. Boloix and P. N. Robillard. A software system evaluation framework. *Computer*, 28:17–26, December 1995.

[10] J. Bortz and N. Döring. *Forschungsmethoden und Evaluation: für Human- und Sozialwissenschaftler*. Springer, Heidelberg, 4., überarb. edition, 2006.

[11] N. Cerpa and J. M. Verner. Why did your project fail? *Commun. ACM*, 52:130–134, December 2009.

[12] Chandler. http://chandlerproject.org. August 2011.

[13] L. Chen, N. R. Shadbolt, and C. A. Goble. A semantic web-based approach to knowledge management for grid applications. *IEEE Trans. Knowl. Data Eng.*, 19(2):283–296, 2007.

[14] A. Cheyer, J. Park, and R. Giuli. Iris: Integrate. relate. infer. share. In Stefan Decker, Jack Park, Dennis Quan, and Leo Sauermann, editors, *Proc. of Semantic Desktop Workshop at the ISWC, Galway, Ireland, November 6*, volume 175, November 2005.

[15] S. Colucci, T. Di Noia, E. Di Sciascio, F. M. Donini, G. Piscitelli, and S. Coppi. Knowledge based approach to semantic composition of teams in an organization. In *Proceedings of the 2005 ACM symposium on Applied computing*, SAC '05, pages 1314–1319, New York, NY, USA, 2005. ACM.

[16] DBin. http://www.dbin.org. August 2011.

[17] S. Decker and M. Frank. The social semantic desktop. In *WWW 2004 Workshop Application Design, Development and Implmentation Issues in the Semantic Web*, 2004.

[18] DeepaMetha. http://www.deepamehta.de. August 2011.

[19] B. Dippelreiter, C. Grün, and M. Pöttler. A 'state of the art' evaluation of pm-systems exploring their missing functionalities. In *Proceedings of the 5th International Conference on Project Management*, pages 90–101, Tokyo, Japan, oct 2010.

[20] B. Dippelreiter and M. Pöttler. Scenarios for evaluating a semantic project management approach. *Scientific Journal of Riga Technical University*, Applied Computer Systems:7, 2011.

[21] B. Dippelreiter and M. Pöttler. Semantic technologies for the project management life cycle improvement. In *8th Extended Semantic Web Conference (ESWC2011), Workshop on Semantic Business Process Management*, 2011.

[22] Doe, R.. The Standish Group - Chaos Report. http://www.projectsmart.co.uk/docs/chaos-report.pdf. May 2011.

[23] H. Dong, F. K. Hussain, and E. Chang. Multi-site project organisation knowledge sharing ontology. *Wireless and Mobile Communications, International Conference on*, 0:18, 2007.

[24] dotProject. http://www.dotproject.net. August 2011.

[25] dotProjectWiki. http://docs.dotproject.net/index.php?title=main_page. August 2011.

[26] K. El Emam and A.G. Koru. A replicated survey of it software project failures. *Software, IEEE*, 25(5):84 –90, sept.-oct. 2008.

[27] R. F. Erlandson. System evaluation methodologies: combined multidimensional scaling and ordering techniques. *SIGMETRICS Perform. Eval. Rev.*, 9:52–58, April 1980.

[28] J. Erpenbeck and L. von Rosenstiel. *Handbuch Kompetenzmessung - Erkennen, verstehen und bewerten von Kompetenzen in der betrieblichen, pdagogischen und psychologischen Praxis*. Schffer-Poeschel, 2003.

[29] M. Fernandez-Lopez, A. Gomez-Perez, and N. Juristo. Methontology: from ontological art towards ontological engineering. In *Proceedings of the AAAI97 Spring Symposium*, pages 33–40, Stanford, USA, March 1997.

[30] M. Fernández-López, A. Gómez-Pérez, A. Pazos, and J. Pazos. Building a chemical ontology using methontology and the ontology design environment. *IEEE Intelligent Systems*, 14:37–46, January 1999.

[31] R. Garcia, R. Gil, J. M. Gimeno, T. Granollers, J. M. Lopez, M. Oliva, and A. Pascual. Semantic wiki for quality management in software development projects. *IET SOFTWARE*, 4(6), 2010.

[32] Gnowsis. http://www.gnowsis.com/about/. July 2011.

[33] A. Gomez-Perez, M. Fernandez-Lopez, and O. Corcho. Ontological engineering: With examples from the areas of knowledge management, e-commerce and the semantic web, 1st edition. 2004.

[34] M. Grüninger and M. S. Fox. Methodology for the design and evaluation of ontologies. In *International Joint Conference on Artificial Inteligence (IJCAI95), Workshop on Basic Ontological Issues in Knowledge Sharing*, 1995.

[35] Hausmann K. Häusler S. Hahn, A. and J. Strickmann. Using ontologies to model and understand product development. *IBIS - Interoperability in Business Information Systems*, Heft 5, 2007.

[36] M. J. Hall, R. Hall, and J. Zeleznikow. A process for evaluating legal knowledge-based systems based upon the context criteria contingency-guidelines framework. In *Proceedings of the 9th international conference on Artificial intelligence and law*, ICAIL '03, pages 274–283, New York, NY, USA, 2003. ACM.

[37] D. Hansch and H. Schnurr. Practical applications of semantic mediawiki in commercial environments - case study: semantic-based project management. In *ESTC 3rd Annual European Semantic Technology Conference*, 2009.

[38] Haystack. Haystack - http://simile.mit.edu/hayloft. August 2011.

[39] March S.T. Park J. Hevner, A.R. and S. Ram. Design science in information systems research. *MIS Quarterly*, 28:75–105, 2004.

[40] R. Hewett and J. Coffey. Xprom: a collaborative knowledge-based project management tool. In *Proceedings of the 13th international conference on Industrial and engineering applications of artificial intelligence and expert systems: Intelligent problem solving: methodologies and approaches*, IEA/AIE '00, pages 406–413, Secaucus, NJ, USA, 2000. Springer-Verlag New York, Inc.

[41] V. Heyse and J. Erpenbeck. *Kompetenztraining, 64 Informations- und Trainingsprogramme*. -Poeschel Verlag Stuttgart, 2004.

[42] Y. Honda, J. Ogawa, and K. Tatebe. Committee meeting for exit criteria at early phase of it project. In *Proceedings of the Society of Project Management ProMAC Symposium*, pages 132–137, 2009.

[43] D. Huynh, D. R. Karger, D. Quan, and V. Sinha. Haystack: a platform for creating, organizing and visualizing semistructured information. In *IUI '03: Proceedings of the 8th international conference on Intelligent user interfaces*, pages 323–323, New York, NY, USA, 2003. ACM.

[44] IRIS. http://www.openiris.org. August 2011.

[45] KACTUS (1996). The KACTUS Booklet version 1.0.Esprit Project 8145 KACTUS. http://www.swi.psy.uva.nl/projects/newkactus/reports.html. March 2010.

[46] M. Kwon, H. Ryu, G. Kim, and S. Ha. Design of owl ontology in r&d project management meeting. In *Proceedings of the 2006 International Conference on Hybrid Information Technology - Volume 01*, ICHIT '06, pages 417–423, Washington, DC, USA, 2006. IEEE Computer Society.

[47] D. B. Lenat and R. V. Guha. *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project*. Addison-Wesley Pub (Sd), April 1990.

[48] S. Mohammadi and A. Khalili. A semantic web service-oriented model for project management. In *Proceedings of the 2008 IEEE 8th International Conference on Computer and Information Technology Workshops*, CITWORKSHOPS '08, pages 667–672, Washington, DC, USA, 2008. IEEE Computer Society.

[49] C. Niculescu and S. Trausan-Matu. An ontology-centered approach for designing an interactive competence management system for it companies. *Romanian Journal of Human-Computer Interaction*, 2(1):73 – 88, 2009.

[50] Noy, N.F. and McGuinness, D.L.: Ontology Development 101: A Guide to Creating Your First Ontology. http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html. March 2012.

[51] Ontoprise. http://wiki.ontoprise.com/wiki/index.php/main_page. August 2011.

[52] OWL Web Ontology Language. http://www.w3.org/tr/2004/rec-owl-features-20040210/s1.3. March 2012.

[53] PHProjekt. http://www.phprojekt.com. July 2011.

[54] J.K. Pinto and Jr. Mantel, S.J. The causes of project failure. *Engineering Management, IEEE Transactions on*, 37(4):269 –276, nov 1990.

[55] J.K. Pinto and D.P. Slevin. Critical factors in successful project implementation. *Engineering Management, IEEE Transactions on*, 34:22–27, 1987.

[56] Projectpier. http://www.projectpier.org. July 2011.

[57] RDF API ARC. https://github.com/semsol/arc2/wiki. October 2010.

[58] RDF API RAP. http://www4.wiwiss.fu-berlin.de/bizer/rdfapi/. October 2010.

[59] G. Reif, T. Groza, S. Handschuh, C. Mesnage, M. Jazayeri, and R. Gudjonsdottir. Collaboration on the social semantic desktop. In *Proceedings of the Ubiquitous Mobile Information and Collaboration Systems Workshop (UMICS 2007), CAISE 2007, Trondheim, Norway*, 2007.

[60] J. Richter, M. Vlkel, and H. Haller. Deepamehta - a semantic desktop. In S. Decker, J. Park, D. Quan, and L. Sauermann, editors, *Proceedings of the 1st Workshop on The Semantic Desktop. 4th International Semantic Web Conference (Galway, Ireland)*, volume 175. CEUR-WS, November 2005.

[61] L. Sauermann. The semantic desktop - a basis for personal knowledge management. In *Proceedings of the I-KNOW*, pages 294–301, 2005.

[62] L. Sauermann, G. A. Grimnes, M. Kiesel, C. Fluit, H. Maus, D. Heim, D. Nadeem, B. Horak, and A. Dengel. Semantic desktop 2.0: The gnowsis experience. In *The Semantic Web - ISWC 2006*, volume Volume 4273/2006, pages 887–900. Springer Berlin / Heidelberg, 2006.

[63] J. Scholtz and M. P. Steves. A framework for real-world software system evaluations. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, CSCW '04, pages 600–603, New York, NY, USA, 2004. ACM.

[64] A. Th. Schreiber, B. J. Wielinga, and W. Jansweijer. The kactus view on the 'o' world. In *Skuce D (ed) IJCAI95 Workshop on Basic Ontological Issues in Knowledge Sharing*, pages 15.1 – 15.10, 1995.

[65] G. Schreiber, H. Akkermans, A. Anjewierden, R. de Hoog, N. Shadbolt, W. van de Velde, and B. Wielinga. *Knowledge engineering and management. The CommonKADS Methodology*. MIT Press, Cmabridge, Massachusetts, 1999.

[66] Semantic Desktop. http://semanticweb.org/wiki/semantic_desktop. August 2011.

[67] Steffen Staab, Rudi Studer, Hans-Peter Schnurr, and York Sure. Knowledge processes and ontologies. *IEEE Intelligent Systems*, 16:26–34, January 2001.

[68] Wei Sun, Qin-yi Ma, Tian-yi Gao, and Li Guo. Applications of semantic web technologies for ontology-based knowledge management in product development. In *4th International Conference on Wireless Communications, Networking and Mobile Computing, 2008. WiCOM '08.*, 2008.

[69] TopBraid. http://www.topquadrant.com/products/tb_composer.html. June 2010.

[70] M. Uschold and M. King. Towards a methodology for building ontologies. In *Workshop on Basic Ontological Issues in Knowledge Sharing, held in conduction with IJCAI-95*, Montreal, Canada, 1995.

[71] M. Uschold, M. King, S. Moralee, and Y. Zorgios. The enterprise ontology. *The Knowledge Engineering Review*, 13(01):31–89, 1998. Cambridge Univ Press.

[72] W3C RDF Validation Service. http://www.w3.org/rdf/validator/. December 2011.

[73] WebCollab. http://www.webcollab.sourceforge.net. July 2011.

[74] J. Westland. *The Project Management Life Cycle: A complete step-by-step methodology for initiating, planning, executing & closing a project successfully*. Kogan Page, Great Britian, new edition edition, 2007.

[75] K. Xu and H. Muñoz avila. Cabma: a case-based reasoning system for capturing, refining, and reusing project plans. *Knowl. Inf. Syst.*, 15:215–232, May 2008.

APPENDIX A

Appendix A - Questionnaire

General:

- Information of the company: Name, Size (number of the employees)

General: Projects

- Do you manage small or big projects? How many projects? Which kind of projects do you manage? IT Projects?
- Did projects already fail in your company? Which ones? How much? Why? (internal communication, missing control, . . .) Did the company learn of failed projects? Can your project management be improved?
- Do you have statistics of your projects (success and disappointment)?

Projects: Tools & Functionalities

- Do you use project management systems?
- Why do you use such systems? What were the motives?
- Which software do you use for project management (MS Office, MS Project, Client based, Server based)?
- Which functionalities does the software have?
- Are you satisfied with your systems? Why? What could be better?
- Do you miss any functionality? Which? Why?

- For choosing your PM system did you design a criteria catalogue? Who did the criteria catalog define?
- Which functionalities do company-wide projects have? Do they have different ones?

Projects: Processing & Control

- Do you have rules for the processing of projects?
- Do you have different project management systems for different projects? Or do you use the same system for all projects?
- Do you have different processes for different projects (small/big)? Do you use different project management systems? Why?
- Do you have special employees as project manager? Why? Do they need a special education/training?
- Do employees have to record their working hours?
- What are typical processes in your project management?
- How do you realize company-wide projects? Do you use applications? Which ones? Why?
- Do you have rules for company-wide projects?
- Which categories could be improved in your project managing?
  - communication
  - standardized terminology
  - faster process handling
  - better process quality
  - transfer of knowledge
  - cost saving
  - improvement of the capacity for teamwork
  - to be able to fall back on already existing knowledge
  - better integration between different tasks
- What are the strengths of your current solution?
- What are the weaknesses of your current solution?

Projects: Old/Existing Knowledge

- Do you use knowledge of old projects for current ones? What information? (results of projects, documents, persons/competences)

- Do you need (sometimes) information of old projects?
- Do you have a kind of archive for information of old/closed projects? Where and how do you archive the information? In what format do you archive the information? (pdf, word, images)

Project management:

- Do you know project management standards? Which one? Do you use such a standard in your company?

APPENDIX B

Appendix B - Evaluation Guideline

Login:
Username: admin
Password: passwd

Scenario 1 - “Bob needs Holidays”

General Description:
Bob is working as software developer in the IT company “Holidays”. He is currently working for the project “Example” where he is responsible for work package (WP) 4. The actual date is March, 14th and Bob is thinking if he could go on holidays in August for two weeks. But in August the deliverable 4.3 of WP 4 has to be finished. So, the question is: Is he able to go on vacation in August when he should work to finish the deliverable?

The technical aspect of this scenario is as follows:
The project leader inserts the scheduled period of holidays and the name of the employee into the project management system. The system checks the assigned work packages, deliverables, tasks and projects of the employee. Furthermore, it monitors the process status of the work as well as possible milestones, dependencies and relations between different tasks. In addition, the PM system checks employees, who are able to do the tasks of the person.

Use Case description:
The employee Birgit Dippelreiter is working in the project SemProM (Semantic based Project Management). In July she wants to go on holidays for two weeks. But during this month the evaluation of the SemProM prototype has to be finished. Is Birgit able to go on holidays or does she have to work?

Questions:

- Does Birgit have any other tasks beside the evaluation of the SemProM prototype?
- What is the status of the tasks?
- Are there any employees with similar competences as Birgit?
- Are these employees related to tasks?
- What are the time spans and status of the tasks?

The requests to answer the main question are separated into six request guidelines. As a result, a better understanding of the different options will be given.

Guideline Part 1 of Scenario 1:
Guideline Part 1 outlines the tasks and their status Birgit is responsible for and related tasks (shown in figure B.1).

Step #	Working Steps
1.1	Press the link “Projects”
1.2	Press the icon “SPARQL”
1.3	Press the link “Show advanced search!”
1.4	Select person “Birgit Dippelreiter”
1.5	Select “Task options” “Show all tasks the person is responsible for”
1.6	Select “Show task status”, “show all”
1.7	Select “Show related tasks”
1.8	Select “Show predecessor”
1.9	Select “Show status”
1.10	Select “Show time span”
1.11	Select “Show related people”
1.12	Select “Show successor”
1.13	Select “Show status”
1.14	Select “Show time span”
1.15	Select “Show related people”
1.16	Press the “show” button

Table B.1: Guideline Part 1 of Scenario 1

Guideline Part 2 of Scenario 1:
Guideline Part 2 highlights the tasks and their status Birgit is working in. In addition, related tasks are shown (depicted in figure B.2).

Step #	Working Steps
2.1	Press the link “Projects”
2.2	Press the icon “SPARQL”
2.3	Press the link “Show advanced search!”
2.4	Select person “Birgit Dippelreiter”
2.5	Select “Task options” “Show all tasks the person is working in”
2.6	Select “Show task status” , “show all”
2.7	Select “Show related tasks”
2.8	Select “Show predecessor”
2.9	Select “Show status”
2.10	Select “Show time span”
2.11	Select “Show successor”
2.12	Select “Show status”
2.13	Select “Show time span”
2.14	Press the “show” button

Table B.2: Guideline Part 2 of Scenario 1

Guideline Part 3 of Scenario 1:
Guideline Part 3 displays people with similar competences as Birgit. Furthermore, it outlines people with similar competences and their tasks they are working in (illustrated in figure B.3).

Step #	Working Steps
3.1	Press the link “Projects”
3.2	Press the icon “SPARQL”
3.3	Press the link “Show advanced search!”
3.4	Select person “Birgit Dippelreiter”
3.5	Select “Task options” “Show all tasks the person is responsible for”
3.6	Select “Show people with similar competences”
3.7	Select “Show their tasks”
3.8	Select “Show their tasks time spans”
3.9	Press the “show” button

Table B.3: Guideline Part 3 of Scenario 1

Guideline Part 4 of Scenario 1:
Guideline Part 4 (shown in figure B.4) highlights people with similar competences as Birgit. Furthermore, it outlines people with similar competences and their tasks they are working in. The number of the people is restricted to those, who are working in the same task as Birgit.

Step #	Working Steps
4.1	Press the link “Projects”
4.2	Press the icon “SPARQL”
4.3	Press the link “Show advanced search!”
4.4	Select person “Birgit Dippelreiter”
4.5	Select “Task options” “Show all tasks the person is responsible for”
4.6	Select “Show people with similar competences”
4.7	Select “Show their tasks”
4.8	Select “Show their tasks time spans”
4.9	Select “Show only people in same task”
4.10	Press the “show” button

Table B.4: Guideline Part 4 of Scenario 1

Guideline Part 5 of Scenario 1:
Guideline Part 5 displays tasks, which are between a predefined time span. Illustrated in figure B.5.

Step #	Working Steps
5.1	Press the link “Projects”
5.2	Press the icon “SPARQL”
5.3	Press the link “Show advanced search!”
5.4	Select person “Birgit Dippelreiter”
5.5	Select “Task options” “Show all tasks the person is responsible for”
5.6	Select “Show task for specific time span”
5.7	Enter “End Date: From:” “01/06/2011”
5.8	Enter “End Date: till:” “30/09/2011”
5.9	Press the “show” button

Table B.5: Guideline Part 5 of Scenario 1

Guideline Part 6 of Scenario 1:
Guideline Part 6 outlines tasks, which are between a predefined time span. In addition, it displays people with similar competences their tasks during the predefined time span (shown in figure B.6).

Step #	Working Steps
6.1	Press the link "Projects"
6.2	Press the icon "SPARQL"
6.3	Press the link "Show advanced search!"
6.4	Select person "Birgit Dippelreiter"
6.5	Select "Task options" "Show all tasks the person is responsible for"
6.6	Select "Show task for specific time span"
6.7	Enter "End Date: From:" "01/06/2011"
6.8	Enter "End Date: till:" "30/09/2011"
6.9	Select "Show people with similar competences"
6.10	Select "Show their tasks"
6.11	Select "show their tasks time spans"
6.12	Select "Show only tasks during time"
6.13	Press the "show" button

Table B.6: Guideline Part 6 of Scenario 1

Scenario 2 - "A new Project"

General Description:
Frida is the project leader of the new software project "Easy Life" within the IT company "Parallels". She inserts the new project into the project management system. Therefore, Frida provides all relevant project information, such as name of the project, project description and used technologies. While storing the data all the supplied information is checked against similarities to existing project knowledge. The system seeks similarities between, e.g., task descriptions, task names, lessons learned and competences.
If similarities to stored projects are found, the information of, e.g., lessons learned or tasks, will be displayed as advice. Now Frida, the project leader, has the choice to decide whether relevant information is contained in the advice or not.
This check can also be repeated later on. With this check information loss can be avoided.

Scenario 3 - "Team Suggestions, etc."

General Description:
Frida, the project leader of the project “Easy Life”, enters the PM system with information regarding work packages, deliverables and tasks. For setting up the optimal team, the system checks all employees, their capacity for teamwork as well as their competences and reports recommendations regarding the project allocation. Furthermore, the system monitors if employees, who might be considered for this project, are already assigned to other projects and tasks. If employees are already completely allocated to another project or task, they will not be considered anymore. If employees are not working with full capacity and their competences fit to the necessary project, they will be listed as available.
At the team allocation of the system all employees are assigned to their possible functions in the new project. Furthermore, it is marked if employees collaborate well with each other or if they have some problems while working together.
If Frida does not agree with the recommendation of the system, she is able to search for other proposals regarding the project allocation. For instance, Frida wants someone else as project manager.

Use Case Description:
Scenario 2 and 3 are merged to one use case. Thus, the use case is as follows:
The project manager **“Birgit Dippelreiter”** of the project “Semantic based Project Management” inserts the Task “Evaluation of the interview results” (shown in figure B.7).

Step #	Working Steps
7.1	Press the link “Projects”
7.2	Press the link “All”
7.3	Press the (project-) link “Semantic based Project Management”
7.4	Press the button “new task”
7.5	Enter the “Task Name” “Evaluation of the interview results”
7.6	Enter the “Task Owner” “Birgit Dippelreiter”
7.7	Enter “Start Date” “02.September 2011”
7.8	Enter “End Date” “09.September 2011”
7.9	Enter “Assumes” “Requirement_Analyst” and “40%”
7.10	Enter “Successor of” “SemProM - Conduct Interviews”
7.11	Enter “Subtask of” “WP6 Proof of Concept”
7.12	Enter “Status “In planning”
7.13	Press the “submit” button

Table B.7: Guideline of Scenario 2 & 3