Using Semantic Documentation to Support Software Project Management

Erick Casagrande Bastos, Monalessa Perini Barcellos, Ricardo de Almeida Falbo

Ontology and Conceptual Modeling Research Group (NEMO), Department of Computer Science, Federal University of Espírito Santo– Vitória – ES – Brazil

`{erickcasagrande, monalessa, falbo}@inf.ufes.br`

Abstract. *Project management is a key process for software projects. Nowadays, there are several tools that support project management activities. However, the existence of such tools did not eliminate the use of documents for documenting the outcomes of the project management process. Text documents and spreadsheets are used to support communication between stakeholders and understanding about the project. These documents are generally developed to be understood by humans and not by computers. Thus, the access to the document content typically depends on human intervention. Retrieving information from documents can be not trivial, especially when it is distributed in several documents. A Semantic Documentation approach can be used to deal with this issue. Combining ontologies and documents by adding semantic annotations to documents makes the document content interpretable by computers and can help diminish the burden of gathering information later on. In this paper, we present a semantic documentation approach for supporting software project management, providing a way to get useful information from data recorded in documents and spreadsheets related to scope, time and cost management. For this, we developed the first version of a Software Project Management Ontology, which is integrated into the Software Engineering Ontology Network. This ontology is used to annotate documents and spreadsheets and to develop features to support scope, time and cost management activities. These features were implemented in the Infrastructure for Managing Semantic Documents in Software Project Management, which was evaluated through an experimental study.*

Keywords: Semantic Documentation, Project Management, Ontology, Semantic Annotation

Acknowledgments

This research is funded by the Brazilian Research Funding Agency CNPq (Processes 485368/2013-7 and 461777/2014-2).

Corresponding author: Monalessa Perini Barcellos (monalessa@inf.ufes.br; monalessa@gmail.com)

Using Semantic Documentation to Support Software Project Management

Abstract. *Project management is a key process for software projects. Nowadays, there are several tools that support project management activities. However, the existence of such tools did not eliminate the use of documents for documenting the outcomes of the project management process. Text documents and spreadsheets are used to support communication between stakeholders and understanding about the project. These documents are generally developed to be understood by humans and not by computers. Thus, the access to the document content typically depends on human intervention. Retrieving information from documents can be not trivial, especially when it is distributed in several documents. A Semantic Documentation approach can be used to deal with this issue. Combining ontologies and documents by adding semantic annotations to documents makes the document content interpretable by computers and can help diminish the burden of gathering information later on. In this paper, we present a semantic documentation approach for supporting software project management, providing a way to get useful information from data recorded in documents and spreadsheets related to scope, time and cost management. For this, we developed the first version of a Software Project Management Ontology, which is integrated into the Software Engineering Ontology Network. This ontology is used to annotate documents and spreadsheets and to develop features to support scope, time and cost management activities. These features were implemented in the Infrastructure for Managing Semantic Documents in Software Project Management, which was evaluated through an experimental study.*

Keywords: Semantic Documentation, Project Management, Ontology, Semantic Annotation

1. Introduction

Effective project management helps organizations succeed in businesses (Hillson 2003). It is a critical competency for meeting strategic goals and implementing innovation initiatives (Anantatmula and Rad 2013). In the last decades, several project management techniques have been proposed and supporting tools have been developed (Jessen 2011). However, although there are several tools to support project management, they are not used by all organizations and often do not eliminate the need of using documents, such as text documents and spreadsheets (Villalobos et al. 2011).

In the context of software projects, documents hold a considerable amount of information that is mainly interpreted by humans (Lethbridge et al. 2003). One disadvantage of using documents is the difficulty of obtaining consolidated information from them, especially when information is spread in several documents. Accessing, recovering and managing information recorded in documents usually depend on human intervention and can be laborious and error-prone. Besides, gathering relevant information from different documents can be so wearing that people may tend not to do so (Arantes and Falbo 2010).

Semantic Web deals with a similar issue by providing a way that both humans and computers can interpret the content of web pages. To reach this goal, web pages are annotated with metadata that describe fragments of the page content so that they become available for computer interpretation (Berners-Lee et al. 2001). Since ontologies represent a conceptualization about the domain of interest and establish a common vocabulary to be shared, they are of great value to describe metadata, providing a rich formal semantic structure for their interpretation. Therefore, ontologies are often used as a basis for semantic annotation (Sicilia 2006).

Semantic Web principles can also be applied to documents (e.g., documents produced in desktop text editors, electronic spreadsheets), giving rise to Semantic Documentation. Semantic Documentation combines documents and ontologies in the same way that Semantic Web, i.e., ontology-based metadata are created and then attached to the document content, resulting in a semantic document (Eriksson 2007). By doing that, syntactic information resources are turned into semantic information resources, which can be interpreted by computers. Once information recorded in the documents is automatically retrieved by computers, it is possible to develop tools to provide consolidated and integrated information for end users, decreasing the human effort necessary to obtain such information. Moreover, semantic annotation approach allows relating annotated contents and using the relationships to extract information from several documents, providing a general view that probably could not be gotten without the annotations (Arantes and Falbo 2010).

During a software project, information regarding planning, execution, progress monitoring and control is recorded in text documents and spreadsheets (e.g., project plan and status reports). Due to the importance of these documents for a project, project members should share them in a common environment and be able to access their content in an easy and efficient way (Talas et al. 2011). Once documents are annotated, it is possible to build repositories containing semantic documents produced in the projects and develop semantic documentation tools able to retrieve consolidated information from them. Besides, semantic documentation helps store and retrieve knowledge acquired during a project and reuse it in other projects.

Tools providing general features[1] to support semantic documentation have been proposed in the literature. They are domain-independent and use ontologies to provide a set of features to manage semantic documents, such as document annotation, storage, indexing and retrieval. Some examples are the Infrastructure for Managing Semantic Documents (IMSD) (Arantes and Falbo 2010), which manages OpenOffice text documents, the PDFTab (Eriksson 2007), which deals with PDF documents, and SDArch (Nesic 2010), which treats MSWord documents.

Project management involves planning, monitoring and controlling aspects related to ten knowledge areas (PMI 2013). Scope, time and cost are considered core areas. They compose the "iron triangle" and represent the main constraints of a project. The use of semantic documentation in these areas can help extract and integrate data recorded in documents, providing an integrated view useful for project managers to make decisions. For instance, by semantically annotating text documents and spreadsheets, information about project schedule recorded in a spreadsheet could be integrated to information about costs of the project resources recorded in a document to provide information about the impact of a delayed schedule on the project costs. Besides, the deliverables of the Work Breakdown Structure (WBS) recorded in a document could be related to the project schedule activities recorded in a spreadsheet, so that it would be possible to get information about the deliverables impacted by the schedule delay. Integrating these pieces of information spread in different documents could help the manager to decide about the actions to be taken due to the schedule delay (e.g., apply some technique to catch schedule delay, renegotiate deadlines or even cancel the project). Semantic annotations could also help keep consistency between project management artifacts. For example, semantic annotations could be used to relate project deliverables to project schedule activities. This would allow verifying if all the project deliverables are associated with activities to produce them, helping keep consistency between WBS and schedule.

In the view of the above, in this paper we explore the use of semantic documentation to support software project management, particularly in aspects related to scope, time and cost. We extended IMSD (Arantes and Falbo 2010) to support project management activities. IMSD is an environment able to manage desktop semantic documents and, different from other proposals (e.g., (Eriksson 2007); (Nesic 2010)), it allows annotating document templates. Thus, documents produced by using the annotated templates are automatically annotated, easing end users work, since they can use the annotated templates to create semantic documents without concern with the annotations to be made.

Since IMSD is domain-independent, only general semantic documentation features are provided. To provide a more effective support to domain tasks, we advocate that it is useful to explore the conceptualization provided by a domain ontology. In other words, it is useful to explore the ontology elements (concepts, relations and properties) and use them to develop domain-specific features. In this sense, to make IMSD able to support semantic documentation in the software project management domain, the conceptualization of a software project management domain ontology should be explored to develop domain-specific features.

The research endeavor discussed in this paper started with an investigation of the state-of-art through a systematic literature review about the use of semantic documentation in project management (Bastos et al. 2015). Six digital libraries were investigated and five proposals were found. The results showed us that this research topic is recent and still not much explored. Besides, from the results, we noticed some gaps: cost and time management has not been explored; only manual annotation mechanisms have been used, and; spreadsheets have not been annotated to support project management.

After the literature review, we analyzed IMSD aiming to identify features to support software project management aspects. Until the beginning of this work, IMSD provided general features to semantically annotate text documents, control versions of semantic content extracted from semantic documents, and search the content of semantic document. There was also an extension of IMSD, called

[1] In this paper, we use the "feature" term to denote functionalities or other computational resources provided by tools to their users.

IMSD-Req (Falbo et al. 2014), which explores the conceptualization established by a Software Requirements Reference Ontology to provide features to support some activities of the Requirement Engineering process. From this analysis, we decided to extend IMSD to support scope, cost and time management. Furthermore, considering that in the context of software project management spreadsheets are useful to record information such as budgets and schedules, we extended IMSD to make it able to annotate spreadsheets.

Next, to establish the domain conceptualization to be used as a basis for semantic annotations and for developing domain-specific features for ISDM to support scope, time and cost management, we built the initial version of the Software Project Management Ontology (SPMO), covering these aspects of software project management. SPMO has been developed integrated into the Software Engineering Ontology Network (SEON) (Ruy et al. 2016). SPMO extends SEON's core ontology on Software Processes (SPO). Like all the core and domain ontologies of SEON, SPMO is grounded in the Unified Foundational Ontology (UFO) (Guizzardi 2005; Guizzardi et al. 2008).

Then, taking SPMO as a basis, we developed the Infrastructure for Managing Semantic Documents for Software Project Management (IMSD-SPM), an extension of IMSD with specific features to support project management aspects. From data recorded in semantic documents and spreadsheets related to scope, time and cost produced along projects, IMSD-SPM features provide to managers: (i) a consolidated view of project planning regarding scope, time and cost; (ii) dependency matrices, showing dependency relations among project elements; (iii) consolidated information about project execution, pointing out the differences between planning and execution values; (iv) project performance indicators; (v) estimates for project conclusion; (vi) a global view of the performance of several projects, allowing for comparisons among them; and (vii) non-conformities detected in the semantic documents and spreadsheets content. After extending IMSD, we conducted an experimental study where eight students with theoretical knowledge and practical experience in project management used IMSD-SPM and evaluated it. The results showed that the use of IMSD-SPM as a way to support software project time, scope and cost management activities is feasible and useful.

Our initial ideas about the use of semantic documentation in the project management domain were published in (Bastos et al. 2015). Later, an overview of our work was published in (Bastos et al. 2016). Now, in this paper, we extend the last publication by providing more details about the IMSD extension and presenting the experimental study carried out to evaluate it. Moreover, as an important contribution of this paper, we present SPMO in details and discuss aspects related to its development as a networked ontology of SEON. Aspects concerning SPMO evaluation and implementation are also discussed in this paper.

This paper is organized as follows: Section 2 provides the background for the paper talking briefly about project management and semantic documentation. This section also introduces IMSD and SABiO (Systematic Approach for Building Ontologies) (Falbo 2014), the ontology engineering method we followed to develop SPMO. Section 3 presents SPMO. Section 4 introduces IMSD-SPM and discusses how we used SPMO to develop it. Section 5 regards the experimental study carried out to evaluate IMSD-SPM. Section 6 concerns related works, and Section 7 presents our final considerations.

2. Background

Project management involves the application of knowledge, skills, tools and techniques to project activities aiming to meet project requirements (PMI 2013). According to the PMBOK (Project Management Body of Knowledge) (PMI 2013), there are ten knowledge areas (KAs) to be managed in a project, namely: Integration, Scope, Stakeholder, Human Resource, Time, Cost, Risk, Quality, Communication, and Procurement. In this work, we explore aspects related to Scope, Time and Cost.

Scope management aims to ensure that the project includes all the required work, and only the required work, to be successfully concluded. It is concerned with defining and controlling what is and what is not included in the project (PMI 2013). The Work Breakdown Structure (WBS) is a key artifact produced during scope planning and represents the project scope. It is a hierarchical organization driven by deliverables to be produced to accomplish the project goals and complete the work established to the project (PMI 2013).

Time management focuses on managing the project deadlines and duration. Its main goal is to finish the project on time (PMI 2013). The main artifact of time management is the project schedule, containing the required activities to produce the WBS items, dependencies among the activities, their durations, dates, resources to be used, and human resources responsible for performing the activities.

Cost management, in turn, addresses cost estimates, budgets and cost control. It aims at finishing the project in accordance with the approved budget (PMI 2013). The main artifact is the project budget, in which project costs are defined and distributed over time.

The software project management process involves project planning and project monitoring and control (ISO/IEC 2008; CMMI 2010). During project planning, a plan for the project is established, including the project scope, team, schedule, and budget, among others. Monitoring and control aims to compare the plans with the project execution, identify problems and present solutions.

During monitoring and control, performance indicators can help the project manager. Earned Value Analysis (Fleming and Koppelman 1999) and its indicators, namely Schedule Performance Index (SPI) and Cost Performance Index (CPI), can be used to provide information about project progress and performance. These indicators consider three dimensions (scope, time and cost) to indicate schedule (SPI) and budget (CPI) compliance. They can also be used to compute cost and time estimations for project conclusion. Three types of estimation can be calculated: optimistic, realistic and pessimistic. Optimistic estimation considers that deviations occurred in the project until now are not a pattern of behavior and will not happen anymore. Realistic estimation considers that the current deviation rate will continue during all the project. Finally, pessimistic estimation considers that even being late, the project will be delivered on time, but the cost will be much higher than the initially established, because more people or overtime will be necessary to catch up on the project work.

Project managers work with several pieces of information, which are captured from different artifacts, such as WBS, project schedule and budget, generally put together in a project plan. For properly monitoring and controlling a software project, the project manager has to handle diverse pieces of information extracted from those artifacts. Moreover, the information must be communicated to customers and other stakeholders. There are several available project management software tools that support project management activities, such as MS Project and Teamwork. However, many software organizations (in particular, very small, small and medium organizations) do not use such tools or use them combined with documents. Spreadsheets, for instance, are widely used by organizations that have limited access to sophisticated project management tools (Villalobos et al. 2011). However, dealing with those documents (text documents and spreadsheets) is not an easy task, since it is hard to properly extract information spread in several documents. Accessing, recovering and managing information recorded in documents depend heavily on human intervention and can be a time-consuming task (Arantes and Falbo 2010). A semantic documentation approach can be useful in this context.

Semantic Documentation refers to the use of semantic annotation to make the semantics of the information inside the documents explicit. By doing that, the document content becomes interpretable by computers. Metadata, which in general concern "data about data", are the main mechanism used for expressing the semantics of the information (Sicilia 2006). By adding semantic annotations to documents, we can reach "intelligent" documents. A semantic document knows about its own content so that automated processes can know what to do with it (Uren et al. 2006). Semantic documents allow providing services such as advanced search, reasoning using document metadata, and knowledge management services, like document repositories and document management.

For semantically annotating documents, we need shared representations of knowledge to establish the basic vocabulary from which metadata statements can be asserted (Sicilia 2006). Since ontologies aim at capturing the intended meaning of a portion of shared knowledge, many researchers defend their use to semantically annotate information resources (Arantes and Falbo 2010; Sicilia 2006; Eriksson 2007; Nesic 2010; Uren et al. 2006; Graaf et al. 2012).

Successful use of semantic documents strongly depends on the effort required to manipulate them, since document authors cannot daily suffer the overhead associated with annotating the documents. The task of manually adding metadata can consume a considerable amount of time (Eriksson and Bang 2006) and be susceptible to errors (Uren et al. 2006). Thus, it is necessary to provide tools to reduce the effort needed to semantically annotate the documents (Uren et al. 2006; Tallis 2003).

Figure 1 illustrates the semantic annotation process for generating a semantic document. The person responsible for adding metadata to documents uses a supporting tool to semantically enrich the document. For each annotation, the tool creates semantic metadata relating a fragment of the document (referring to general information or document content) to an element of the ontology. Usually, metadata are kept in the semantic document.

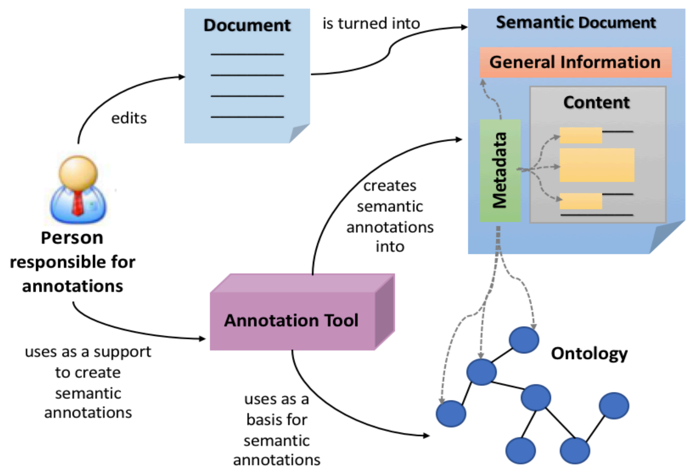

Document
is turned into
Semantic Document
General Information
Content
Metadata
edits
creates semantic annotations into
Person responsible for annotations
Annotation Tool
Ontology
uses as a support to create semantic annotations
uses as a basis for semantic annotations

Figure 1 – Creating semantic documents (adapted from (Eriksson and Bang 2006), (Eriksson 2007) and (Arantes and Falbo 2010)).

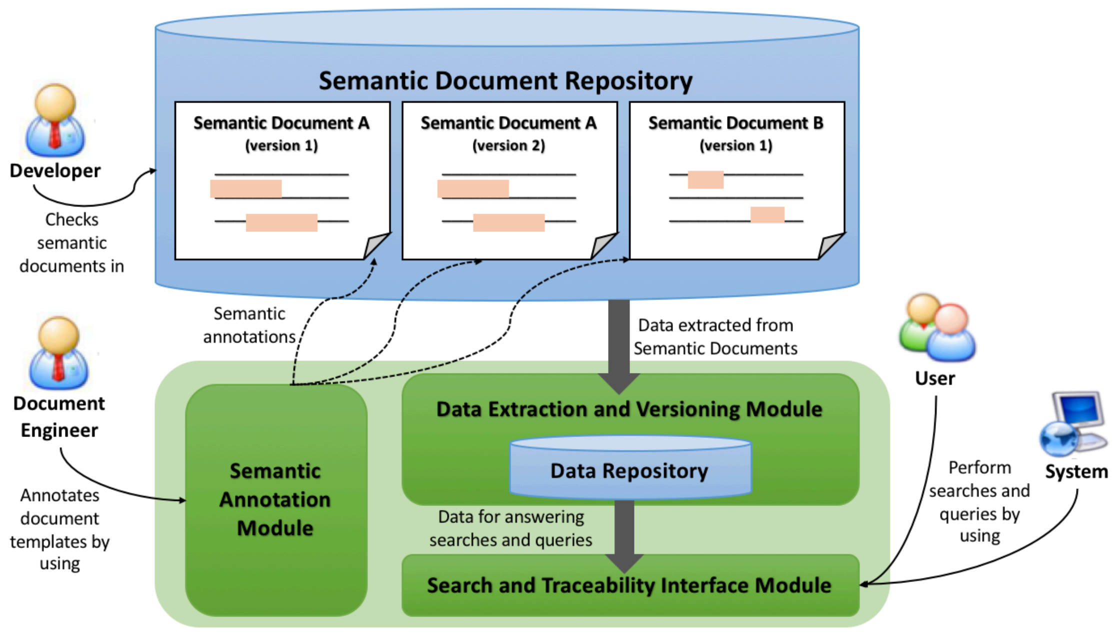
Once documents are annotated, it is possible to extract knowledge and link contents from different documents according to the shared ontology. By integrating content extracted from several documents, it is possible to achieve a more holistic view of the available knowledge (Arantes and Falbo 2010).

Several tools have been developed intending to support semantic document management (Arantes and Falbo 2010; Eriksson 2007; Popov et al. 2003). As discussed in the Introduction of this paper, this work uses the Infrastructure for Managing Semantic Document (IMSD) (Arantes and Falbo 2010). The main reason that led us to select ISMD is that it semantically annotates document templates instead of documents. Adding semantic annotations to templates can be a simple approach to diminish the overhead users may found related to document annotation. Once templates are annotated, users can create documents by using the semantic templates. By doing that, the resulting documents are semantic documents that carry the annotations made in the used template. Another reason that contributed to our choice is that we have got access to the ISMD source code.

IMSD is a domain-independent tool which provides the following features to support semantic documentation: *(i)* a way to semantically annotate document templates; *(ii)* a mechanism for controlling versions of semantic content extracted from semantic document versions, and therefore providing a way for tracking the evolution of the data embedded inside a semantic document; and *(iii)* data visibility to end-users allowing for searches and data change notification subscription for aiding developers to get up-to-date information about something they are interested in.

Figure 2 shows an overview of the IMSD architecture, comprising two main elements: the *Semantic Document Repository*, which is responsible for storing semantic documents, and the *Main Module*, which is composed of three sub-modules:

- *Semantic Annotation Module*: responsible for allowing document engineers to semantically enrich document templates.
- *Data Extraction and Versioning Module*: responsible for extracting the semantic content from an annotated document whenever a new version of that document is checked in the Semantic Document Repository. After extraction, the semantic content is stored, along with version information in another repository, called Data Repository, which is also part of this module.
- *Search and Traceability Interface Module*: responsible for providing an API (Application Programming Interface) that allows users and other systems to perform ontology-based searches and data traceability towards the Data Repository.



Semantic Document Repository
Semantic Document A (version 1)
Semantic Document A (version 2)
Semantic Document B (version 1)
Developer
Checks semantic documents in
Semantic annotations
Data extracted from Semantic Documents
User
Document Engineer
Annotates document templates by using
Semantic Annotation Module
Data Extraction and Versioning Module
Data Repository
Data for answering searches and queries
Search and Traceability Interface Module
Perform searches and queries by using
System

Figure 2 - Overview of the ISDM Architecture.

In a nutshell, document engineers annotate document templates using the Semantic Annotation Module. Later, developers may instantiate a template (i.e., use it to create a document), generating a semantic document. This semantic document can be checked in the Semantic Document Repository. When a new version of a semantic document is available at the Semantic Document Repository, the Data Extraction and Versioning Module unwraps the semantic content of that version and stores it into the Data Repository, making the information of that semantic content available. At this point, it is possible to further enhance the integration of information scattered throughout various semantic documents stored in the Semantic Document Repository. Finally, users and other systems may interact with the Search and Traceability Interface Module to perform searches and queries about the evolution of a particular semantic content stored in the Data Repository.

In this work, we extended IMSD to support software project management. Aiming at annotating documents related to scope, time and cost management and developing domain-specific features, we developed the Software Project Management Ontology (SPMO). SPMO development followed SABiO, a Systematic Approach for Building Ontologies (Falbo 2014).

SABiO has been used and improved for more than ten years and prescribes an iterative process for ontology engineering. Its development process comprises five main phases: (i) *purpose identification and requirements elicitation*, which involves identifying the ontology purpose and its intended uses, eliciting its requirements, and modularizing the ontology (when necessary); (ii) *ontology capture and formalization,* which regards capturing the ontology concepts, relations, properties and constraints, and creating conceptual models, dictionary of terms and formal axioms; (iii) *design*, dealing with technological aspects needed to implement the ontology in a particular machine-readable ontology language (e.g., OWL); (iv) *implementation*, consisting of implementing the ontology in the chosen operational language; and (v) *test*, referring to dynamic verification and validation of the operational ontology. As the result of the two first phases, a reference ontology (i.e., a solution-independent conceptual model making the best possible description of the domain (Guizzardi 2007)) is produced. The three other phases produce an operational ontology (i.e., a machine-readable implementation version of the ontology (Guizzardi 2007)).

3. The Software Project Management Ontology (SPMO)

By following SABiO, we started by defining SPMO purpose and intended uses. The purpose of SPMO is to establish a common conceptualization about the software project management domain, initially focusing on scope, time and costs planning and monitoring. Regarding costs, at first, only costs associated with human resources are considered. SPMO is intended to be used as a reference model for: (i) supporting human learning on the software project management domain; (ii) integrating project management supporting tools; (iii) structuring and representing knowledge related to software project management; and (iv) annotating project management related resources in a semantic documentation approach. This last intended use is the one addressed in this work. Briefly, here, SPMO is used to assign semantics to documents and spreadsheets content, and also to help develop features to support software project management using a semantic documentation approach. Thus, in this work, concepts related to techniques or methods applied to project management (e.g., Gantt and PERT charts) are out of the ontology scope.

Considering its purpose and intended uses, as functional requirements, we defined the following set of competency questions to be answered by SPMO:

CQ1. Which are the processes and activities defined for a given project?
CQ2. How is a project process structured in terms of sub-processes and activities?
CQ3. Which is the software organization that commits to perform a given project process and its activities?
CQ4. Which other project activities does a project activity depend on?
CQ5. Which stakeholders are allocated to perform a project activity?
CQ6. What are the estimated schedule, duration and cost for a project process?
CQ7. What are the estimated schedule, duration and cost for a project activity?
CQ8. What are the deliverables to be produced in the context of a project?
CQ9. What are the deliverables to be produced by a project activity?
CQ10. What are the estimated schedule, duration and cost of a stakeholder allocation?
CQ11. Which were the processes and activities performed in a given project?
CQ12. How was a performed process structured in terms of sub-processes and activities?
CQ13. Which other performed activity did a given performed activity depend on?
CQ14. Which stakeholders participated in a performed activity?
CQ15. What was the actual time frame, duration and cost of a performed process?
CQ16. What was the actual time frame, duration and cost of a performed activity?
CQ17. What was the actual time frame, duration and cost of a stakeholder participation?
CQ18. Which intended process caused a given performed process?
CQ19. Which intended activity caused a given performed activity?
CQ20. On which stakeholder allocation is a stakeholder participation based?
CQ21. Regarding time frame, duration and cost, was a performed process executed according to the intended process that caused it?
CQ22. Regarding time frame, duration and cost, was a performed activity executed according to the intended activity that caused it?
CQ23. Regarding time frame, duration and cost, was a stakeholder participation performed according to the stakeholder allocation to which it is based on?

Besides the functional requirements, posed as competency questions, we also elicit non-functional requirements for SPMO, namely:

NFR1. SPMO should take the main standards of project management into account, in particular, PMBoK (PMI 2013). This non-functional requirement is aligned to the characteristic of beautiful ontologies of being based on international standards (d'Aquin and Gangemi 2011).
NFR2. SPMO should be grounded in a foundational ontology. This non-functional requirement is aligned to the characteristic of beautiful ontologies of reusing foundational ontologies (d'Aquin and Gangemi 2011).
NFR3. Since the software project management domain is very complex, SPMO should be extensible, modular and should reuse already existing related ontologies. This non-functional requirement is aligned to the characteristic of beautiful ontologies of being modular or embedded in a modular framework (d'Aquin and Gangemi 2011).

To address both NFR2 and NFR3, we decided to develop SPMO as a networked ontology of the Software Engineering Ontology Network (SEON) (Ruy et al. 2016). An ontology network is a collection of ontologies related together through a variety of relationships, such as alignment, modularization, and dependency. A networked ontology, in turn, is an ontology included in such a network, sharing concepts and relations with other ontologies (Suárez-Figueroa et al. 2012). SEON is designed seeking to: (i) take advantage of well-founded ontologies (all its ontologies are ultimately grounded in the Unified Foundational Ontology – UFO (Guizzardi 2005; Guizzardi et al. 2008)); (ii) provide ontology reusability and productivity, supported by core ontologies organized as Ontology Pattern Languages (OPLs) (Falbo et al. 2013); and (iii) solve ontology integration problems by providing integration mechanisms, as well as favoring an incremental ontology development through reuse and extension. In its current version, SEON includes a core ontology for software processes, as well as domain ontologies for the main technical software engineering subdomains, namely requirements, design, coding and testing, and for some management subdomains, namely configuration management, measurement and quality assurance.

Managing projects requires the effective management of suitable processes (PMI 2013). During software project planning, knowledge about software processes is useful in several situations, such as software process definition (including defining sub-processes and activities, as well as artifacts and required resources), scheduling of activities, and allocation of people to these activities (Bringuente et al. 2011). Thus, SPMO was developed by extending the SEON's core ontology on Software Processes (SPO) (Bringuente et al. 2011). Fragments of two parts of SPO were reused: the Intended Software Process Definition sub-ontology and the Software Process Execution sub-ontology. The first deals with the processes defined for specific projects. These processes are intended processes in the sense that they are only planned, i.e., they are commitments to perform actions. The last is about processes that were already performed, i.e., performed processes are actions performed to meet the commitments. Besides, both SPO and SPMO are grounded in the Unified Foundational Ontology (UFO) (Guizzardi 2005; Guizzardi et al. 2008). Since SPMO extends SPO, SPO competency questions relevant to the domain of interest were reused. Moreover, SPMO was modularized based on the SPO sub-ontologies. As Figure 3 shows, in its current version, SPMO is divided into two sub-ontologies: Estimated Process and Tracked Process.

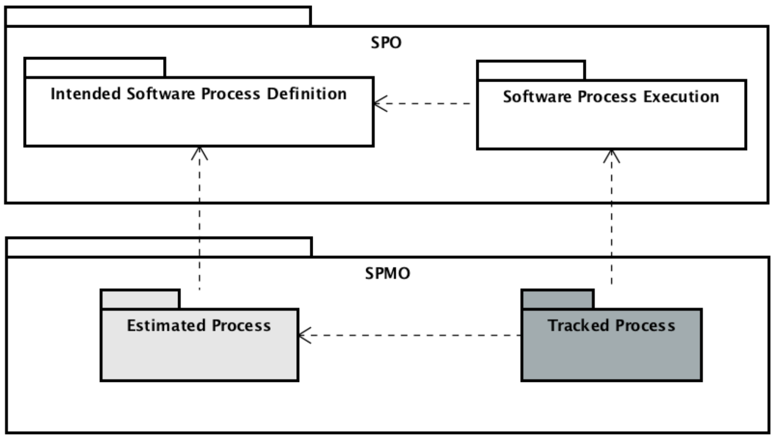

SPO
Intended Software Process Definition
Software Process Execution
SPMO
Estimated Process
Tracked Process

Figure 3 - SPMO Modularization.

The Estimated Process sub-ontology addresses scope, time and costs estimation. Estimated processes and activities are processes and activities not yet accomplished, and thus they refer to intentions (in terms of UFO) to perform actions. This sub-ontology extends concepts and relations from SPO's Intended Software Process Definition sub-ontology. The Tracked Process sub-ontology, in turn, deals with already performed activities and, thus, refer to actions (in terms of UFO). This sub-ontology extends concepts and relations from SPO's Software Process Execution sub-ontology.

For each sub-ontology, we accomplished the *ontology capture and formalization* phase of SABiO. By extending SPO and considering the established competency questions, we produced conceptual models, descriptions for each concept in natural language, and First Order Logic axioms for capturing constraints not captured by the conceptual models. The conceptual models are developed using the Unified Modeling Language (UML), with the concepts grounded in concepts of UFO. For sake of simplicity, in this paper we do not discuss in detail this grounding nor present UFO concepts in the diagrams. For details on this, see SEON Specification available at http://nemo.inf.ufes.br/seon.

Next, for each SPMO sub-ontology, we present its conceptual model, descriptions and some axioms. Since SPMO extends SPO, first we present the fragments of SPO that were reused. SPO has been developed in successive refinements, and the current version is an extension of the one presented in (Bringuente et al. 2011).

3.1 Software Process Ontology

In its current version, SPO is organized in four main sub-ontologies: (i) the Standard Software Process Definition sub-ontology deals with generic processes (types of processes, in fact) defined by an organization, establishing the basic requirements for processes to be performed in that organization; (ii) the Intended Software Process Definition sub-ontology addresses the definition of intended processes to be accomplished in the scope of a specific software project; (iii) the Software Process Execution sub-ontology is about processes already performed and their activities, and the stakeholders, resources, artifacts and procedures involved in their executions; (iv) the Process Assets sub-ontology talks about the elements (planned to be) used or handled by the activities during a software process. These sub-ontologies are further split into smaller sub-ontologies. Here, we address only the SPO fragment relevant for this paper. For details, see the SEON Specification.

As Figure 3 shows, SPMO extends parts of two SPO sub-ontologies: Intended Software Process Definition sub-ontology and Software Process Execution sub-ontology. The first addresses competency questions CQ1 to CQ5, which relate to core aspects of planned processes and their activities. The last addresses competency questions CQ11 to CQ19, which regard to the execution of processes and activities. Figure 4 shows the SPO view used in this paper, containing the concepts and relations that are later reused or extended in SPMO.

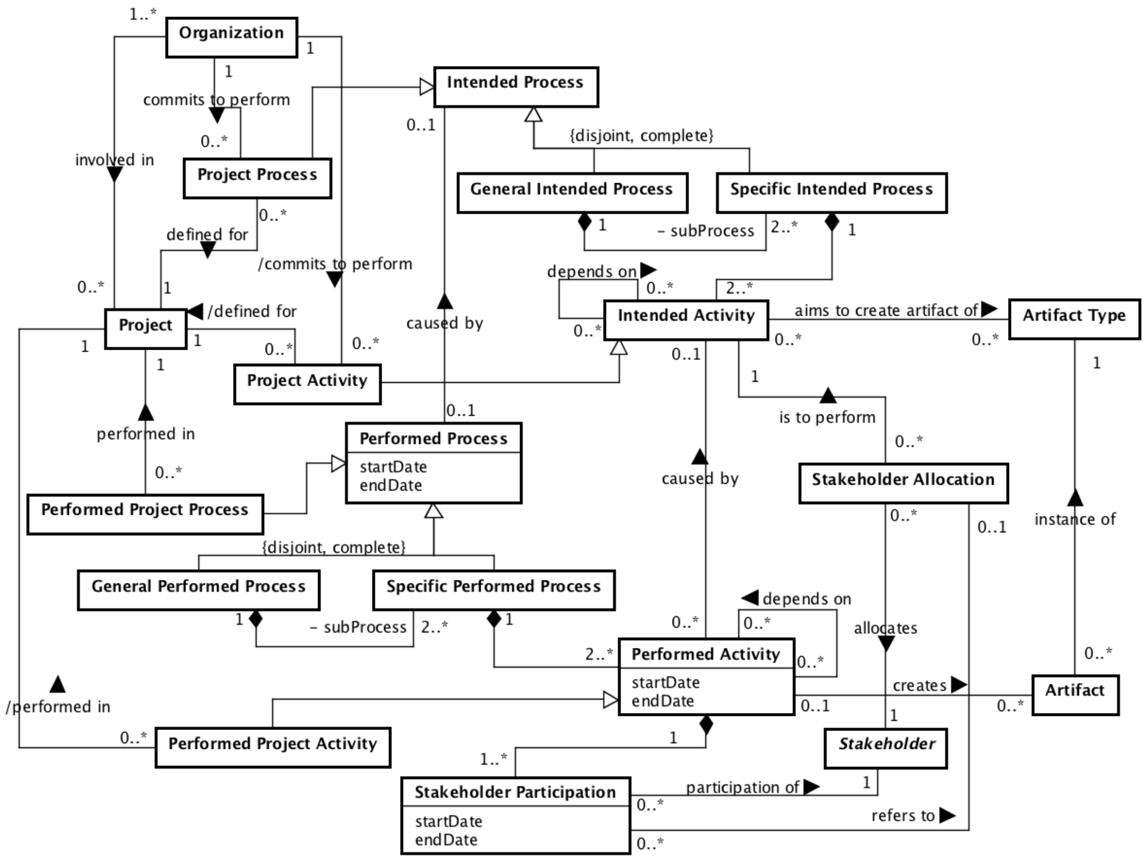


Figure 4 – A View of SPO.

An ***Organization*** can be involved in ***Projects***, and it may commit to performing actions in the context of these projects. Before performing actions effectively, an organization establishes commitments to perform them. These commitments are established during project planning, more precisely when the project manager defines a ***Project Process*** for the project. At this moment, an internal commitment (an Intention in UFO) of the organization is created: the commitment of performing the activities defined in the project process. Thus, ***Project Processes/Activities*** are ***Intended Processes/Activities***. Both intended processes and activities are intentions to perform actions; what changes is the granularity level of the actions to be performed.

There are two types of ***Intended Processes***: ***General Intended Process*** and ***Specific Intended Process***. The constraint {disjoint, complete} shown in the model represents that there are only these two types of intended processes and that an intended process is either a general intended process or a specific intended process (never both at the same time). The first one is the global process defined for a project (for instance, the general process defined for the project *P*). It is composed of specific processes, allowing defining ***sub-processes***. For instance, the general process defined for the project *P* could be composed of the following specific project processes: *Software Development*, *Project Management* and *Quality Assurance*. Specific Project Processes, in turn, are composed of ***Project Activities***, which can be ***Simple Project Activities*** or ***Composite Project Activities*** (not shown in Figure 4 for sake of simplicity). For instance, the *Project Management* process could be composed of the composite project activities *Project*

Planning and *Project Monitoring and Control*. *Project Planning*, in turn, could be composed of other activities, such as the simple project activity *Project Plan Elaboration*. Finally, ***Stakeholders*** may be allocated to Intended Activities, giving rise to ***Stakeholder Allocations***.

According to UFO, intentions cause actions, i.e., actions are intentional events. Thus, Intended Processes can cause the occurrence of actions named here ***Performed Processes***. Similarly, an Intended Activity can cause the occurrence of ***Performed Activities***. As events, Performed Processes and Activities have a time frame (start/end dates). Analogously to the Intended Processes that cause them, Performed Processes can be ***General Performed Processes*** or ***Specific Performed Process***. Specific Performed Processes are composed of ***Performed Activities***, which can be ***Simple Performed Activities*** or ***Composite Performed Activities*** (again not shown in Figure 4 for sake of simplicity). Moreover, processes and activities performed in the context of a Project are said to be ***Performed Project Processes*** and ***Activities***, respectively.

Several stakeholders can participate in a Performed Activity. A ***Stakeholder Participation*** refers to the participation of a Stakeholder in a Performed Activity. A Stakeholder Participation may refer to a Stakeholder Allocation. For instance, the stakeholder participation of *John* in the *Project Plan Elaboration* performed activity would refer to the stakeholder allocation of *John* to the *Project Plan Elaboration* project activity.

When an Intended Activity is planned, the project manager may indicate types of artifacts (***Artifact Type***) that this activity aims to produce. When a Performed Activity is accomplished, if it is caused by an Intended Activity that aims at producing an artifact of such type, then it is expected that the Performed Activity creates an ***Artifact*** that is an instance of the corresponding Artifact Type.

There are constraints involving concepts and relations of SPO that cannot be captured by the conceptual model. Thus, to make them explicit we defined axioms and formalized them in First Order Logic.

Table 2 presents three of these axioms. It is important to say that some of the axioms defined in SPO give rise to derived relations in the conceptual model (relations shown in the figure preceded by "/"). For instance, (A1) gives rise to the relation "/defined for" between Project Activity and Project in Figure 4.

Table 1 - Some SPO Axioms.

Id	Axiom
(A1)	If an Intended Process *ip* is both a General Intended Process and a Project Process, then *ip* is a General Project Process.
	$\forall ip: \mathit{Intended\ Process}\ \ \mathit{GeneralIntendedProcess}(ip) \land \mathit{ProjectProcess}(ip) \rightarrow \mathit{GeneralProjectProcess}(ip)$
(A2)	If an Intended Process *ip* is both a Specific Intended Process and a Project Process, then *ip* is a Specific Project Process.
	$\forall ip: \mathit{IntendedProcess}\ \ \mathit{SpecificIntendedProcess}(ip) \land \mathit{ProjectProcess}(ip) \rightarrow \mathit{SpecificProjectProcess}(ip)$
(A3)	If a Project Activity *pa* is part of a Specific Project Process *spp* defined for a Project *p*, then *pa* is also defined for *p*.
	$\forall pa: \mathit{Project\ Activity}, spp: \mathit{Specific\ Project\ Process}, p: \mathit{Project}\ \ \mathit{partOf}(pa, spp) \land \mathit{definedFor}(spp, p) \rightarrow \mathit{definedFor}(pa, p)$
(A4)	If a Project Activity *pa* is part of a Specific Project Process *spp* which is part of a General Project Process *gpp*, then *pa* is also part of *gpp*.
	$\forall pa: \mathit{Project\ Activity}, spp: \mathit{Specific\ Project\ Process}, gpp: \mathit{General\ Project\ Process}\ \ \mathit{partOf}(pa, spp) \land \mathit{partOf}(spp, gpp) \rightarrow \mathit{partOf}(pa, gpp)$
(A5)	If a Performed Project Process *ppp* is caused by a Project Process *pp* defined for a Project *p*, then *ppp* should be performed in the context of *p*.
	$\forall ppp: \mathit{PeformedProjectProcess}, pp: \mathit{ProjectProcess}, p: \mathit{Project}\ \ \mathit{causedBy}(ppp, pp) \land \mathit{definedFor}(pp, p) \rightarrow \mathit{performedIn}(ppp, p)$

3.2 Estimated Process Sub-Ontology

The focus of the Estimated Process Sub-ontology is on scope, time and costs estimation. This sub-ontology addresses competency questions CQ6 to CQ10, which refer to**Erro! Fonte de referência não encontrada.** estimates. Figure 5 shows the conceptual model of this sub-ontology. Concepts reused from SPO are shown in white, while concepts introduced in SPMO are shown in grey.

Once a ***General Intended Process*** is defined for a ***Project***, it is possible to plan duration, start and end dates, and cost of the process, their sub-processes and activities. The definition of dates to an ***Intended Process*** gives rise to a ***Scheduled Process***. Similarly, the planning of dates of an ***Intended Activity*** gives rise to a ***Scheduled Activity***. A Scheduled Process, when estimated in terms of duration and costs, becomes an ***Estimated Process***. Analogously, a Scheduled Activity, when estimated in terms of duration and costs, but also in terms of the ***Planned Work Packages*** that it should deliver (scope planning), becomes an ***Estimated Activity***. It is important to say that the estimated duration and cost of an Estimated Process is given by the estimates for duration and cost of the Estimated Activities that composes the Estimated Process, as captured by axiom (A7) in Table 2. For this reason, the properties ***estimatedDuration*** and ***estimatedCost*** are derived properties, as shown in Figure 5 (properties shown preceded by "/").

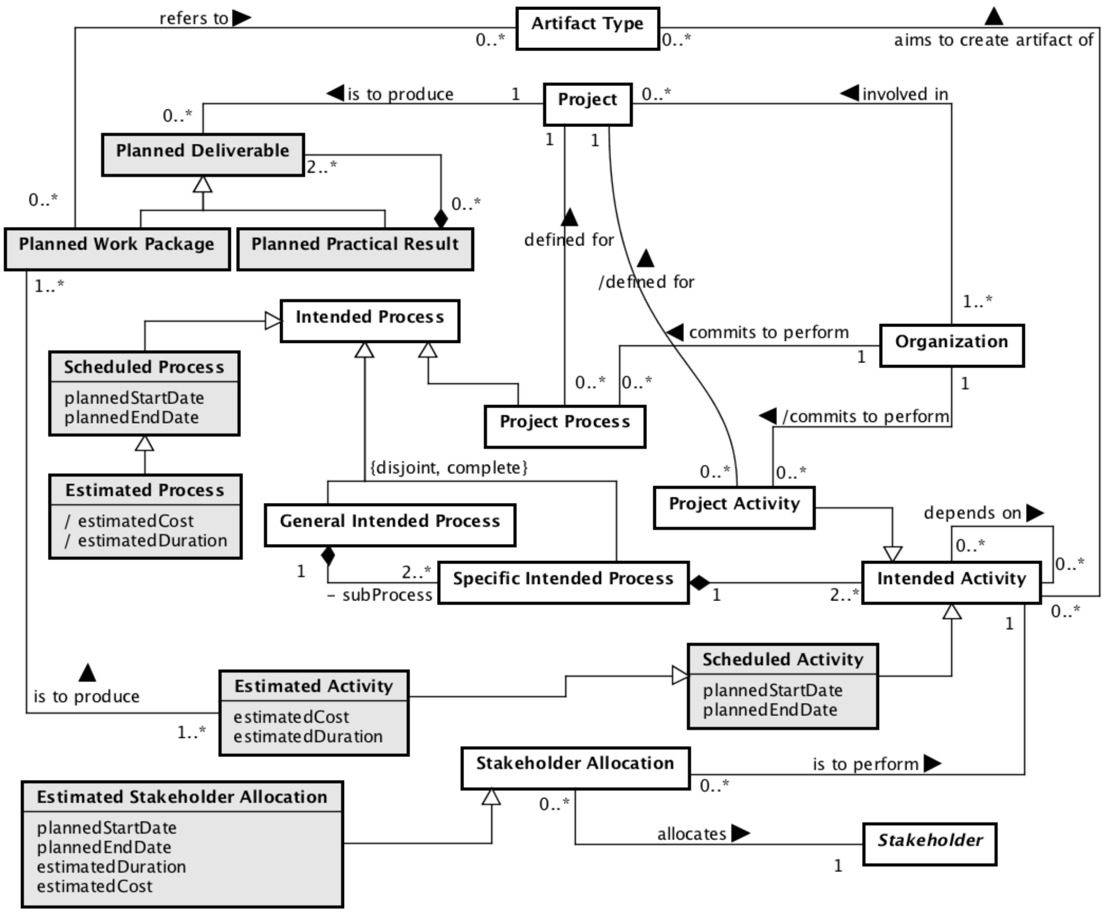


Figure 5 - Estimated Process sub-ontology.

A Project is performed to produce ***Planned Deliverables***. These deliverables can be of two types: ***Planned Practical Results*** or ***Planned Work Packages***. Planned Practical Results are composed of other Planned Deliverables. Planned Work Packages, in turn, are the smallest planned deliverables to which the work to be done in the project is associated. Thus, a planned work package must have at least one Estimated Activity related to it. Moreover, if a Planned Work Package refers to the production of an artifact of a certain type, then the Estimated Activity related to it aims to create an artifact of the same Artifact Type. For instance, the *information system IS* could be a planned practical result to be produced in a project and the *User's Manual* (which refers to the production of an artifact of the *User's Manual* type) could be one of the planned work packages that compose that practical result. The estimated activity *Elaborate User's Manual* could be the project activity defined to produce the *User's Manual*. Thus, the estimated activity

Elaborate User's Manual must be an Intended Activity that aims to create an artifact of the *User's Manual* type.

Stakeholders are allocated to perform Intended Activities. As discussed before, ***Stakeholder Allocation*** is the assignment of an Intended Activity to a Stakeholder. A Stakeholder Allocation can also be estimated, giving rise to an ***Estimated Stakeholder Allocation***.

Several constraints involving SPMO concepts and relations cannot be captured by the conceptual model. Thus, to make them explicit we defined axioms and formalized them in First Order Logic.

Table 2 presents two axioms defined for SPMO.

Table 2 - Some SPMO Axioms of the Estimated Process Sub-Ontology.

Id	Axiom
(A6)	If an Estimated Activity *ea* is to produce a Planned Work Package *pwp*, and *ea* is defined for a Project *p*, then *p* should also produce *wp*.
	$\forall$*ea: Estimated Activity, pwp: Planned Work Package, p: Project* *isToProduce(ea, pwp)* $\wedge$ *definedFor(ea, p)* $\rightarrow$ *isToProduce(p, pwp)*
(A7)	The estimated duration of an Estimated Process *ep* is the sum of the estimated duration of the Estimated Activities ea_i that compose *ep*.
	$\forall$ *ep: Estimated Process;* $ea_1, ea_2, \ldots, ea_n$*: Estimated Activity* $(ea_1.estimatedDuration = d_1) \wedge (ea_2.estimatedDuration = d_2) \wedge \ldots \wedge (ea_n.\ estimatedDuration = d_n) \wedge partOf(ea_1,ep) \wedge partOf(ea_2,ep) \wedge \ldots \wedge partOf(ea_n,ep) \rightarrow (ep.estimatedDuration = d_1+d_2+\ldots+d_n)$

3.2 Tracked Process Sub-Ontology

The Tracked Process Sub-ontology is about already (at least partially) performed actions, and aims to answer competency questions CQ20 to CQ23, addressing aspects related to tracking estimated and actual values. Figure 6 shows its conceptual model. Concepts reused/extended from SPO are shown in white, while concepts introduced in SPMO are shown in grey. Those introduced in this sub-ontology are shown in dark grey.

Processes and activities performed in the context of a Project are said to be ***Performed Project Processes*** and ***Activities***. ***Performed Processes /Activities*** in general (and not only those performed in the context of a project) have a time frame (start/end dates). When actual data regarding their cost and duration are collected, we call them ***Tracked Processes*** and ***Activities***. A Tracked Process is caused by an Estimated Process, while a Tracked Activity is caused by an Estimated Activity. Thus, the actual cost and duration of a Tracked Process/Activity can be compared with the corresponding estimated values of the Estimated Process/Activity that caused it. In Tracked Process/Activity, the progress index indicates the process/activity progress (e.g., 80%, 100%) in relation to the Estimated Process/Activity that caused it. Again, it is worthwhile to point out that the actual duration and cost of a Tracked Process is given by the actual duration and cost of the Tracked Activities that composes the Tracked Process, as captured by axiom (A8) in Table 3. For this reason, the properties ***duration*** and ***cost*** (as well as ***progressIndex***) are derived properties, as shown in Figure 6 (properties shown preceded by "/").

For Tracked Activities, we need to take information regarding the Planned Work Packages that they actually contribute to produce, in order to contrast them with the planned scope. When a Tracked Activity produces an artifact that meets the Planned Work Package to which the Tracked Activity contributes for, we say that this artifact is a ***Deliverable Artifact***. We should highlight, however, that some work packages may refer to services to be done, instead of artifacts to be produced. As a consequence, it is not always the case that a Tracked Activity produces an artifact as a deliverable. In these cases, it is enough to link the Tracked Activity to the Planned Work Package it contributes for.

Several stakeholders can participate in a Performed Activity. A ***Stakeholder Participation*** refers to the participation of a Stakeholder in a Performed Activity. A Stakeholder Participation may refer to a Stakeholder Allocation. For instance, the stakeholder participation of *John* in the *Project Plan Elaboration* performed activity would refer to the stakeholder allocation of *John* to the *Project Plan Elaboration* project activity. Again, when data regarding the cost and duration of a Stakeholder Participation are collected, we call it a ***Tracked Stakeholder Participation***.

As in the case of the Estimated Process Sub-ontology, in the Tracked Process Sub-ontology, there are constraints involving its concepts and relations that cannot be captured by the conceptual model. Thus, to make them explicit we defined axioms and formalized them in First Order Logic. Table 3 presents three axioms defined for SPMO.

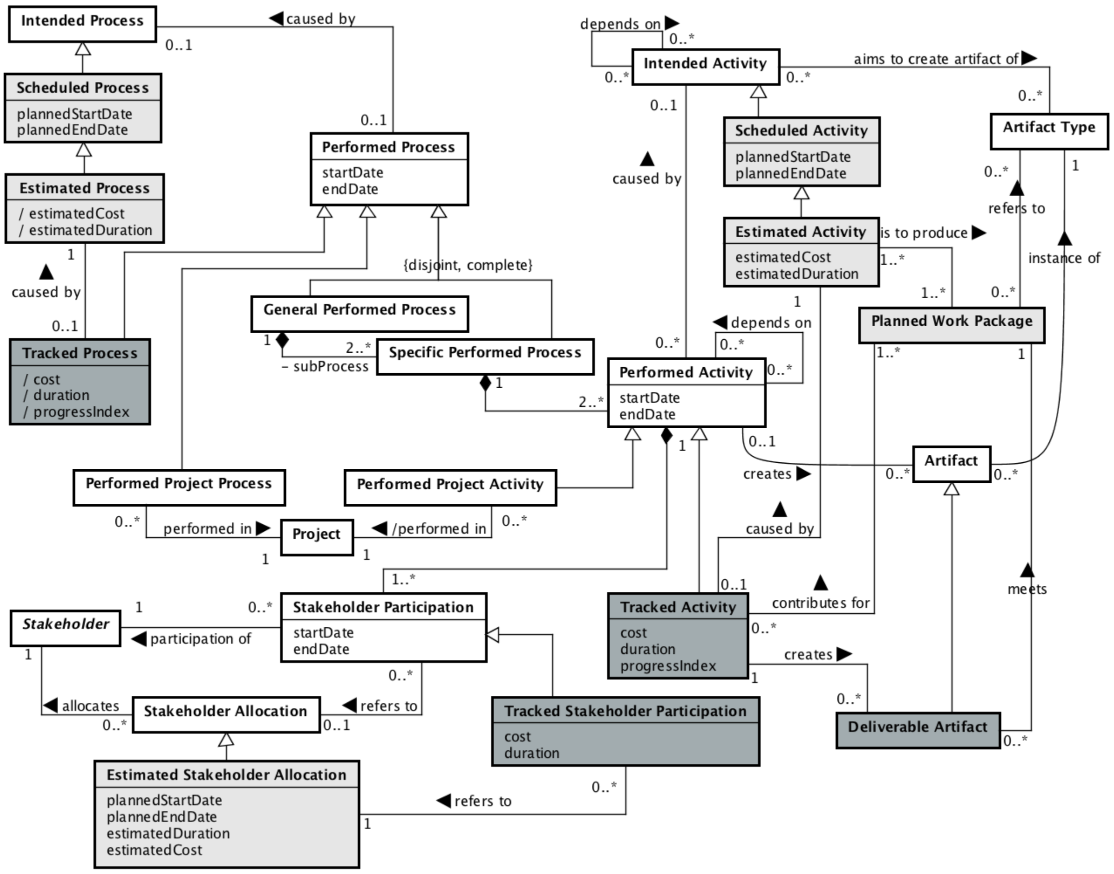


Figure 6 - Tracked Process sub-ontology.

Table 3 - Some SPMO Axioms of the Tracked Process Sub-Ontology.

Id	**Axiom**
(A8)	The actual duration of a Tracked Process *tp* is the sum of the actual duration of the Tracked Activities ta_i that compose *tp*.
	$\forall tp$: *Tracked Process*; $ta_1, ta_2, \ldots, ta_n$: *Tracked Activity* $(ta_1.duration = d_1) \wedge (ta_2.duration = d_2) \wedge \ldots \wedge (ta_n.duration = d_n) \wedge partOf(ta_1,tp) \wedge partOf(ta_2,tp) \wedge \ldots \wedge partOf(ta_n,tp) \rightarrow (tp.duration = d_1+d_2+\ldots+d_n)$
(A9)	If a Tracked Process *tp* was caused by an Estimated Process *ep* and the duration of *tp* is equal to the estimated duration of *ep*, then, regarding duration, *tp* was performed in conformance to *ep*.
	$\forall tp$: *Tracked_Process*, *ep*: *Estimated_Process* $causedBy(tp, ep) \wedge (tp.duration = ep.estimatedDuration(pp, p) \rightarrow durationConformance(ep, tp)$
(A10)	If a Tracked Process *tp* was caused by an Estimated Process *ep* and the duration of *tp* is shorter than the estimated duration of *ep*, then, regarding duration, *ep* was overestimated.
	$\forall tp$: *TrackedProcess*, *ep*: *EstimatedProcess* $causedBy(tp, ep) \wedge (tp.duration < ep.estimatedDuration(pp, p) \rightarrow overestimatedDuration(ep, tp)$
(A11)	If a Tracked Process *tp* was caused by an Estimated Process *ep* and the duration of *tp* is longer than the estimated duration of *ep*, then, regarding duration, *ep* was underestimated.

	$\forall tp$: *TrackedProcess, ep: EstimatedProcess* *causedBy(tp, ep)* $\wedge$ *(tp.duration>ep.estimatedDuration(pp, p)* $\rightarrow$ *underestimatedDuration(ep, tp)*

3.3 SPMO Evaluation

Once SPMO was developed, we needed to evaluate it. Following SABiO, which suggests that ontologies should be evaluated by verifying and validating them, we verified SPMO by evaluating if it contains the concepts, relations and axioms necessary (and only the necessary ones) to answer the competency questions, and we validated SPMO by instantiating its concepts to evaluate if it is able to represent elements extracted from the real world.

For verifying SPMO, for each competency question, we created an entry in a table showing the concepts, relations and axioms used to answer it. During verification, some competency questions were decomposed, and the vocabulary was adjusted to be aligned to the SPMO terminology. Table 4 presents, as example, a fragment of the table created to verify SPMO.

Table 4 - SPMO Verification.

CQ	Concepts, Relations and Properties	Axioms
CQ6.1 - What is the estimated duration of an estimated process?	Estimated Process is *subtype of* Intended Process Intended Activity is *part of* Intended Process Estimated Activity is *subtype of* Intended Activity Estimated Activity has *estimatedDuration*	A4, A7
CQ22.1 - Regarding duration, was a tracked activity executed according to the estimated activity that caused it?	Tracked Activity is *caused by* Estimated Activity Estimated Activity has *estimatedDuration* Tracked Activity has *duration*	A9, A10, A11

For validating SPMO, for each concept, we created an entry in a table showing elements extracted from the real world and that instantiate the concept. Table 5 presents, as examples, instances of some SPMO concepts.

Table 5 - SPMO Validation.

Concept	Instance
Project	Sys-Game Project
Project Process	Development Process defined for the Sys-Game Project.
Project Activity	Develop System Documentation activity, which is part of the Development Process defined for the Project Sys-Game.
Scheduled Activity	Develop System Documentation, with planned start date = 05/22/2017 and planned end date = 05/25/2017
Estimated Activity	Develop System Documentation, with estimated duration = 32 hours; estimated cost = U$ 320; and related to the work package System Documentation
Estimated Stakeholder Allocation	The allocation of the stakeholder John to perform the Develop System Documentation activity, with planned start date = 05/22/2017; planned end date = 05/25/2017; estimated duration = 32 hours; estimated cost = U$ 320.
Deliverable	Sys-Game
Practical Result	Sys-Game
Work Package	System Documentation
Tracked Activity	Develop System Documentation activity, performed in the Sys-Game Project, with start date = 05/22/2017; end date 05/26/2017; duration = 36 hours; cost = U$360; progress index = 100%; and related to the work package System Documentation.
Tracked Stakeholder Participation	The participation of the stakeholder John in the Develop System Documentation performed activity, with start date = 05/22/2017; end date 05/26/2017; duration = 36 hours; cost = U$360.
Deliverable Artifact	System Documentation of the Sys-Game Project that meets the System Documentation work package.

The ontology produced by accomplishing the two first SABiO phases is a reference ontology, i.e., it is a solution-independent conceptual model and it is not implemented in a machine-readable ontology language. In the context of this work, we also need an operational version of the ontology to be used to annotate documents and spreadsheets content. Thus, by following the three last phases of SABiO, an operational version of SPMO was produced. In the next subsection, some aspects of the design, implementation and test of the resulting operational ontology are discussed.

3.3 SPMO Operational Version: SPMO-OWL

SPMO operational version (SPMO-OWL) was implemented in OWL as a lightweight ontology, i.e., some axioms of the reference version were not implemented in the operational version. In general, since the reference ontology conceptual model is modeled in UML, we performed transformations from UML constructs to OWL constructs. Table 6 shows the main types of mappings we performed.

Table 6 – Designing the operational version of SPMO: Mapping UML to OWL.

UML	OWL

Class	**<owl:Class>**, with id got from the name of the class, without spaces;
	Add **<rdfs:subClassOf>** in the case of subclasses.
	E.g.:
	`<owl:Class rdf:about="http://localhost/ontologies/SPMO/spmo.owl#EstimatedActivity">`
	`    <rdfs:label rdf:datatype="&xsd;string">Estimated Activity</rdfs:label>`
	`    <rdfs:subClassOf`
	`rdf:resource="http://localhost/ontologies/SEON/spmo.owl#ScheduledActivity"/>`
	`</owl:Class>`

Property	**<owl:DatatypeProperty>**, with id got from the name of the property plus the name of the class, without spaces.
	E.g.:
	`<owl:DatatypeProperty rdf:about="&xsd;estimatedDurationEstimatedActivity">`
	`    <rdfs:domain rdf:resource="#EstimatedActivity"/>`
	`    <rdfs:range rdf:resource="&xsd;nonNegativeInteger"/>`
	`</owl:DatatypeProperty>`

Association	**<owl:ObjectProperty>**, with id got from the name of the association plus the name of the classes involved, without spaces.
	E.g.:
	`<owl:ObjectProperty`
	`rdf:about="http://localhost/ontologies/SPMO/spmo.owl#definedForProjectProcessProject">`
	`  <rdfs:label rdf:datatype="&xsd;string">defined for</rdfs:label>`
	`  <rdfs:domain rdf:resource="http://localhost/ontologies/SPMO/spmo.owl#ProjectProcess"/>`
	`  <rdfs:range rdf:resource="http://localhost/ontologies/SPMO/spmo.owl#Project"/>`
	`</owl:ObjectProperty>`

For implementing SPMO-OWL, we used Protégé, and the resulting OWL file (spmo.owl) was used to annotate the templates of documents and spreadsheets. For testing SPMO-OWL, we adopted an application-based approach (Brank et al., 2005), i.e., we tested SPMO-OWL in the context of the application for which it was developed. This application is presented in the next section.

4. IMSD-SPM: Infrastructure for Managing Semantic Documents in Software Project Management

For applying semantic documentation in the project management domain, both general and domain-specific features are needed. Semantic documentation general features support activities such as document annotation and version control and can be applied to any domain. Domain-specific features, in turn, support project management activities. As previously said, for implementing our approach for semantic documentation in the project management domain, we extended IMSD (see Section 2). In order to identify the extensions to be made, we analyzed IMSD considering the general and domain-specific perspectives.

Regarding the general perspective, we noticed that IMSD supported semantic documentation only in documents in text format. Since spreadsheets are very useful for recording data regarding projects (e.g., schedules and budgets), we identified as an improvement opportunity to extend IMSD to work with spreadsheets, expanding the types of files that can be used as sources for it. Concerning the domain-specific perspective, we decided to support project time, scope and cost management activities, and then we defined six requirements for IMSD-SPM:

(**R1**) Data regarding project planning may be recorded in different documents. Thus, IMSD-SPM shall provide a consolidated view of these data, helping managers to have a global view of the project planning.

(**R2**) It is important to know the dependencies between project activities, and between them and project deliverables. Thus, IMSD-SPM shall provide dependency matrices to represent these dependencies, allowing visualizing dependencies (in particular, indirect and transitive dependencies) and analyzing the impact of changes in a project.

(**R3**) Project monitoring and control depend on information about progress, allowing comparing planned and actual data. Performance indicators can be useful in this context because they quantitatively represent the project performance, and also can be used to calculate estimates for the project conclusion. Thus, IMSD-SPM shall provide performance indicators, helping managers to understand the project performance and identify situations that need corrective actions. Besides, IMSD-SPM shall derive estimations for project conclusion, providing a view on the implications of the current project performance, and allowing the manager to make decisions about project continuation or interruption.

(**R4**) As a project progresses, the amount of information to be managed increases. IMSD-SPM shall provide graphical views containing consolidated information about the project, including the project performance over time.

(**R5**) Several projects are developed in a software organization. It is important to analyze them together to verify discrepancies and investigate their causes. Thus, IMSD-SPM shall present performance indicators of several projects, providing a global view of the projects and allowing for comparisons among them.

(**R6**) Templates are useful mechanisms to improve project management documentation, and they should be consistently applied. In this sense, IMSD-SPM shall support identifying non-conformities in project management documents. Checklists shall be provided to automatically check the quality of those documents.

The identified improvement opportunity and requirements were implemented by extending and specializing IMSD, as discussed in the next two subsections.

4.1 Supporting Software Project Management with Semantic Annotations in Spreadsheets

To treat the identified improvement opportunity, we extended IMSD to work with spreadsheets and then we used SPMO as a basis to annotate spreadsheet and document templates related to the project management domain. In IMSD, annotations are added to templates resulting in *semantic templates* that when instantiated (i.e., when used to create documents and spreadsheets) give rise to *semantic documents*. Once the templates are annotated, the documents produced using them are also annotated and can be used as data sources to IMSD. Spreadsheet templates were developed using the Open Document Format (OASIS 2017), since it is an open format, with great span. Annotations for cells were produced using Open Document Spreadsheet (ODS) in LibreOffice Calc[2].

For spreadsheets annotation, the instructions for annotating text originally defined in IMSD are used to capture the cell content. These instructions allow creating instances of concepts and establishing relations between them, as defined in an ontology implemented in OWL and available in a URL.

A semantic annotation expression used to annotate cells has two parts: a tag, and the instruction itself. The expression must begin with the tag *[[completeText]]*, when the content of the annotated cell is to be considered as a whole, or with the tag *[[break with '<separator>']]*, when fragments of the content, separated by <*separator*> (e.g., comma), are to be considered.

The syntax of the instruction for creating an instance of a concept is:

instance (source, concept, variable).

This instruction creates, from the *source*, an instance of a *concept* and sets in *variable* a reference to the created instance for later use. The first argument of the instruction (*source*) can be of two types: *{content},* when the instruction begins with the tag *[[completeText]]*, or *{slice},* when the instruction begins with the tag *[[break with '<separator>']]. {content}* and *{slice}* are keywords of IMSD. The former means that the instruction must be executed once for the whole content of the annotated cell, while the last means that the instruction must be executed as many times as there are content fragments (slices) separated by the separator. As the second argument of the instruction (*concept*), the path of the domain ontology used for annotation must be included followed by '#' and the name of the concept to be used to set semantics to the cell content. Last, the variable that will refer to the instance created in IMSD must be added preceded by '$'. For example, the semantic annotation expression

[[completeText]]; instance ({content}, http://localhost/ontologies/SEON/spmo.owl#Project, $p)

creates an instance of *Project*, whose value is the data stored in the *{content}* of the cell, and sets a reference to this instance on *$p* for later use by IMSD.

[2]*www.libreoffice.org*

The syntax of the instruction for setting a value to a property of an instance is:

property (arg1, prop, arg2).

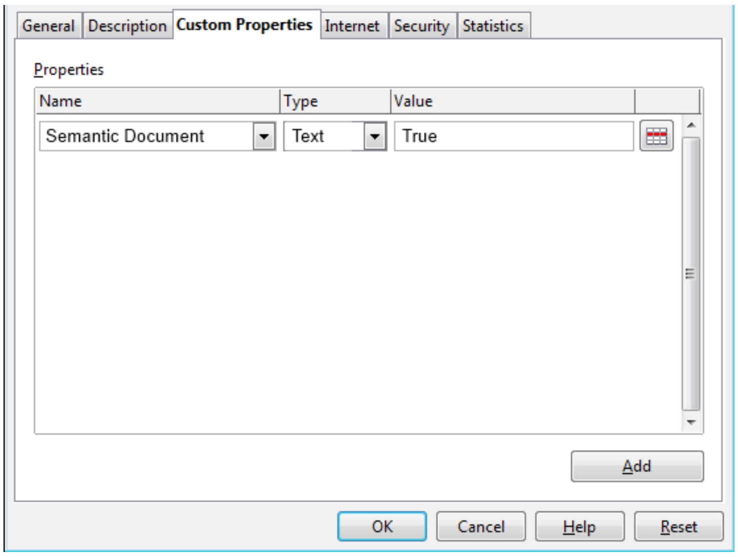
This instruction can also be used to establish a relation *prop* between two instances *arg1* and *arg2*.

For setting a value to a property, *arg2* must be *{content}* or *{slice},* indicating the source from which the value of the property *prop* will be obtained. *arg1* must be the variable (*$var*) that refers to the instance to which the property value will be set. *prop* refers to the property, and it must include the ontology path. For example, the semantic annotation expression

[[completeText]]; property ($ea, http://localhost/ontologies/SEON/spmo.owl#estimatedCost, {content})

sets the *{content}* of the cell as the value of the *estimatedCost* property of the instance of *Estimated Activity* referenced by the variable *$ea*.

For annotating spreadsheet templates in the LibreOffice Calc, we used its custom properties (they can be defined by accessing the *Custom Properties* option) to record annotations and styles (they can be defined through the *Styles and Formatting* option) to apply annotations to cells. When creating a semantic template, a custom property named *Semantic Document* must be created and its value must be set to *True*, as Figure 7 shows. In this way, IMSD identifies that a spreadsheet is a semantic document and searches for semantic annotations.



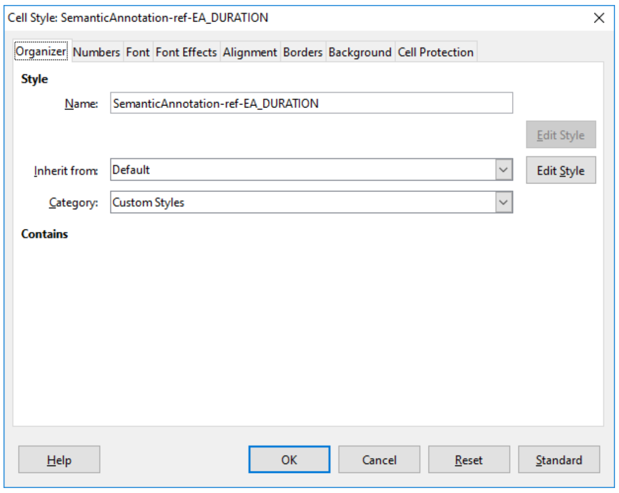
Name	Type	Value
Semantic Document	Text	True

Figure 7 - Setting a semantic spreadsheet.

Each annotation must be recorded in a new custom property whose value is the annotation expression. For each annotation, a style must be created and related to the custom property in which the annotation is recorded. Thus, when a style is applied to a cell, the cell is annotated according to the annotation expression recorded in the corresponding custom property. For IMSD to be able to identify the relationship between a style and a custom property, it is necessary to create a custom property whose value is the semantic annotation expression and a style, whose name follows the following pattern:

SemanticAnnotation-ref-<CustomPropertyName>

where <*CustomPropertyName*> is the name of the custom property to be associated with the style. Figure 8 illustrates how a style is created, and Figure 9 shows examples of custom properties and styles created to annotate spreadsheets. As highlighted in Figure 9, by following the pattern presented above, it is possible to identify the style a custom property is related to.


Cell Style: SemanticAnnotation-ref-EA_DURATION

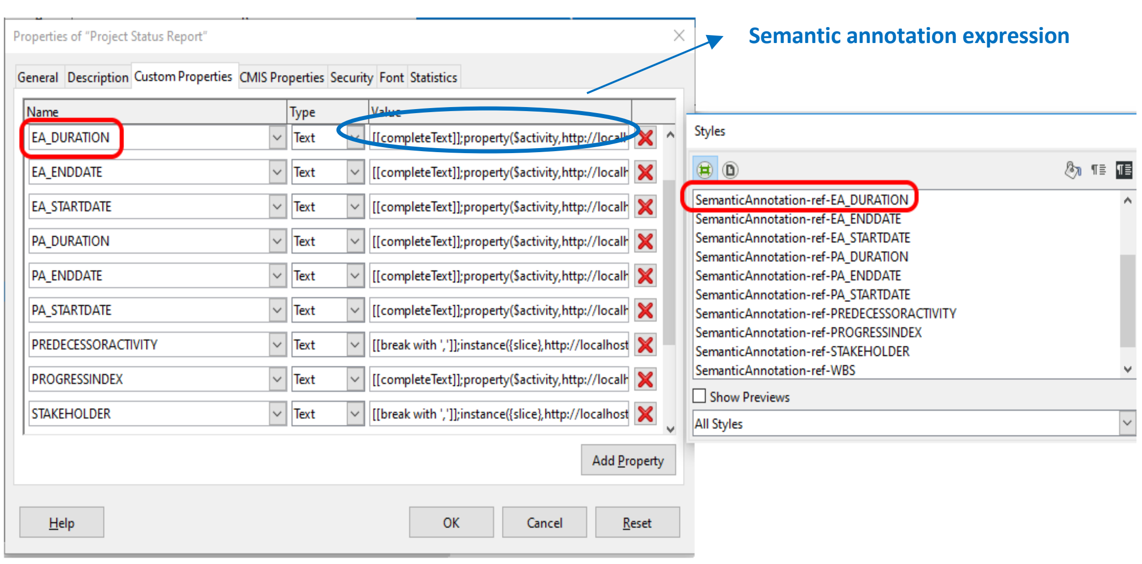
Organizer | Numbers | Font | Font Effects | Alignment | Borders | Background | Cell Protection

Style

Name:	SemanticAnnotation-ref-EA_DURATION
Inherit from:	Default
Category:	Custom Styles

Contains

Figure 8 – Creating a Style.



Properties of "Project Status Report"

General | Description | Custom Properties | CMIS Properties | Security | Font | Statistics

Name	Type	Value
EA_DURATION	Text	[[completeText]];property($activity,http://locall
EA_ENDDATE	Text	[[completeText]];property($activity,http://localh
EA_STARTDATE	Text	[[completeText]];property($activity,http://localh
PA_DURATION	Text	[[completeText]];property($activity,http://localh
PA_ENDDATE	Text	[[completeText]];property($activity,http://localh
PA_STARTDATE	Text	[[completeText]];property($activity,http://localh
PREDECESSORACTIVITY	Text	[[break with ',']];instance({slice},http://localhost
PROGRESSINDEX	Text	[[completeText]];property($activity,http://localh
STAKEHOLDER	Text	[[break with ',']];instance({slice},http://localhost

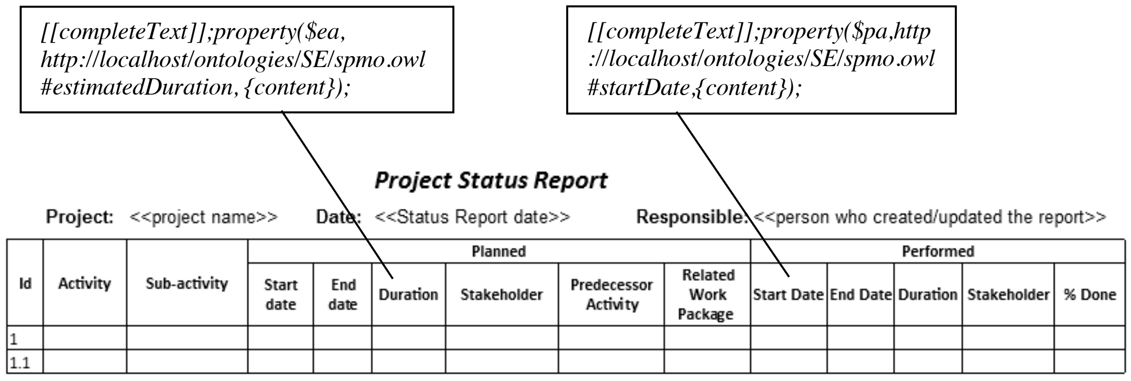
Styles

SemanticAnnotation-ref-EA_DURATION
SemanticAnnotation-ref-EA_ENDDATE
SemanticAnnotation-ref-EA_STARTDATE
SemanticAnnotation-ref-PA_DURATION
SemanticAnnotation-ref-PA_ENDDATE
SemanticAnnotation-ref-PA_STARTDATE
SemanticAnnotation-ref-PREDECESSORACTIVITY
SemanticAnnotation-ref-PROGRESSINDEX
SemanticAnnotation-ref-STAKEHOLDER
SemanticAnnotation-ref-WBS

Figure 9 - Examples of Custom Properties and Styles to annotate spreadsheets.

Semantic documents managed by IMSD are created from semantic templates. This allows for partial automation of annotation and reduces the amount of work that users need to do to create semantic documents. For instance, for the specific features provided by IMSD-SPM, we created three semantic templates related to project management, namely: *Work Breakdown Structure* (WBS), a text document that presents the project WBS, containing the project deliverables; *Project Status Report* (PSR), a spreadsheet that contains information regarding planned and accomplished schedules; and *Human Resources Costs* (HRC), a spreadsheet that contains information about costs of human resources.

Figure 10 illustrates a fragment of the template of the *Project Status Report* spreadsheet, which contains information about project activities, dependencies between them, stakeholders allocated to them and those that actually participated in their execution, related work packages, and planned and actual dates and duration. As examples, annotations related to the cells of the Planned Duration and Performed Start Date columns are shown. In both annotations, the *[[completeText]]* tag indicates that the instruction refers to the complete text stored in the cell. The first expression means that the cell content will be set as the value of the *estimatedDuration* property of an instance of *Estimated Activity*. The second means that the cell content will be set as the value of the *startDate* property of an instance of *Performed Activity*.


[[completeText]];property($ea, http://localhost/ontologies/SE/spmo.owl #estimatedDuration, {content});

[[completeText]];property($pa,http ://localhost/ontologies/SE/spmo.owl #startDate,{content});

Project Status Report

Project: <<project name>> **Date:** <<Status Report date>> **Responsible:** <<person who created/updated the report>>

Id	Activity	Sub-activity	Planned						Performed				
			Start date	End date	Duration	Stakeholder	Predecessor Activity	Related Work Package	Start Date	End Date	Duration	Stakeholder	% Done
1													
1.1													

Figure 10 - Project Status Report template (a semantic template).

When the project manager uses the semantic templates to create a WBS, PSR or HRC, the created text documents or spreadsheets become semantic documents. Then, they can be submitted to IMSD, which extracts data from them and stores in OWL files, allowing for search and retrieval, and also performs version control. It is important to notice that, although we have created semantic templates related to the project management domain, spreadsheets annotation is a general feature, since by using it one can create semantic templates related to any domain.

4.2 Supporting Software Project Scope, Time and Costs Management

To address R1 to R6, we explored SPMO conceptualization for developing domain-specific features that were implemented to support scope, time and costs management activities. These features were implemented in a specialization of IMSD called *Infrastructure for Managing Semantic Documents in Software Project Management* (IMSD-SPM). Table 7 presents the specific features of IMSD-SPM and the requirements addressed by them. In each feature, some SPMO concepts, relations and properties were explored.

Table 7 – IMSD-SPM Features and Addressed Requirements.

Feature	Short Description	Addressed Requirement
Simplified Project Plan	Presents consolidated information about project scope, time and costs planning.	R1
Dependency Matrices	Presents dependency matrices containing dependencies between project activities and between project activities and project deliverables.	R2
Project Monitoring Panel	Presents consolidated information about project execution, pointing out the differences between planned and actual values.	R3
Performance Indicators	Presents the current performance indicators of a project.	R3
Estimates for Project Conclusion	Presents the optimistic, realistic and pessimistic estimates for project conclusion, considering the current performance indicators.	R3
Project Performance History	Presents in a chart the project performance indicators and estimations over time.	R4
Projects Comparison	Presents in charts performance indicators and estimations for conclusion related to several projects, allowing comparing the values of different projects.	R5
Quality Assurance	Automatically evaluates the quality of semantic documents and spreadsheets considering a set of criteria.	R6

The first step to use the specific features provided by IMSD-SPM is to create a Semantic Document Repository to store semantic documents. This repository is, in fact, a repository in Subversion, a version control system. The repository will store the semantic documents produced from the semantic templates. When starting a project, the project manager creates the WBS, PSR and HRC semantic documents, containing information regarding the project planning and checks them in the repository. Data contained in the semantic documents are extracted by the Data Extraction and Versioning Module and stored in the IMSD-SPM data repository. From this moment, the *Simplified Project Plan* and *Dependency Matrices* features, which are related to project planning, can be used by the project manager.

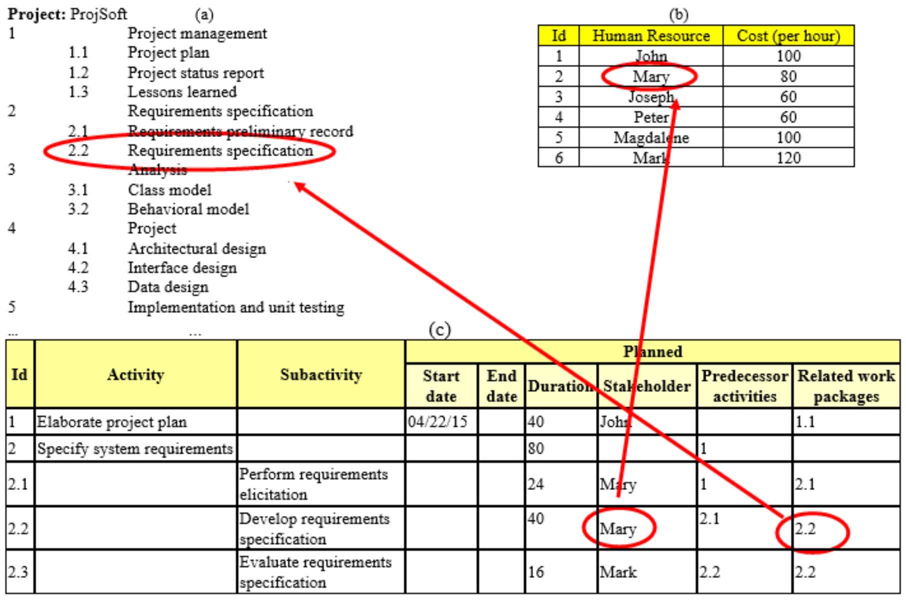
Along the project, the manager may update the semantic documents by adding data about the project execution and checks the new versions of the semantic documents in the document repository. For example, periodically, the manager should check in new versions of the Project Status Report spreadsheet, containing data regarding the project execution until that moment. Data recorded in the new versions of the semantic documents are extracted and stored in the data repository. Once data about the project execution is stored, the manager can use the IMSD-SPM features related to project monitoring and control (*Project Monitoring Panel*, *Performance Indicators*, *Estimates for Project Conclusion* and *Project Performance History*).

When data regarding execution of more than one project are stored in the data repository, managers can use the *Projects Comparison* feature to compare performance indicators of different projects. The *Quality Assurance* feature can be used at any time to automatically verify the semantic documents stored in the document repository.

Next, we present some of the IMSD-SPM features and make some discussions about the use of SPMO in their context.

4.2.1 The Simplified Project Plan and the Project Monitoring Panel

A *Simplified Project Plan* integrates data recorded in different semantic documents. Its first version is generated when the first versions of the WBS, PSR and HRC are submitted to IMSD-SPM. New versions can be generated when data regarding planning are updated and new versions of the documents are submitted. The Simplified Project Plan presents information extracted from semantic documents and also data generated by IMSD-SPM. For instance, planned start and end dates of some activities can be determined by IMSD-SPM considering the planned duration of other activities, activities dependencies, and the start date of the first activity. Besides, the composition relations between project activities and processes, and axioms that address planned costs in these relations (the planned cost of an activity is the sum of the planned costs of the activities that compose it, and the planned cost of a process is the sum of the planned costs of its activities) allow inferring that the total planned cost of the general process defined to the project is the sum of planned costs of all activities that compose the process. Figure 11 shows fragments of WBS, PSR and HRC semantic documents submitted to IMSD-SPM. Ellipses and arrows are used to highlight examples of related information.


Project: ProjSoft (a)

1		Project management
	1.1	Project plan
	1.2	Project status report
	1.3	Lessons learned
2		Requirements specification
	2.1	Requirements preliminary record
	2.2	Requirements specification
3		Analysis
	3.1	Class model
	3.2	Behavioral model
4		Project
	4.1	Architectural design
	4.2	Interface design
	4.3	Data design
5		Implementation and unit testing
...		...

(b)

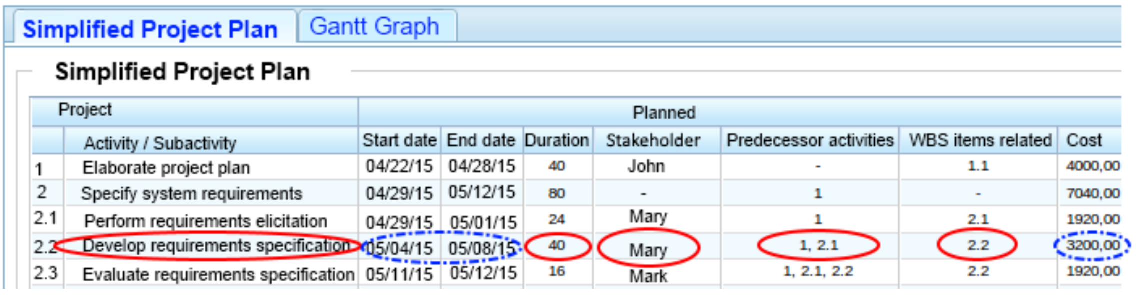
Id	Human Resource	Cost (per hour)
1	John	100
2	Mary	80
3	Joseph	60
4	Peter	60
5	Magdalene	100
6	Mark	120

(c)

Id	Activity	Subactivity	Planned					
			Start date	End date	Duration	Stakeholder	Predecessor activities	Related work packages
1	Elaborate project plan		04/22/15		40	John		1.1
2	Specify system requirements				80		1	
2.1		Perform requirements elicitation			24	Mary	1	2.1
2.2		Develop requirements specification			40	Mary	2.1	2.2
2.3		Evaluate requirements specification			16	Mark	2.2	2.2

Figure 11- Semantic documents fragments: (a) WBS, (b) HRC, (c) PSR.

A fragment of the Simplified Project Plan generated based on the documents shown in Figure 11 is depicted in Figure 12. Data inside solid red ellipses are examples of data extracted from semantic documents. Data inside dotted blue ellipses are examples of data calculated by IMSD-SPM by exploring the SPMO conceptualization. To ease the plan visualization, a Gantt Graph is also provided (not shown in Figure 12).

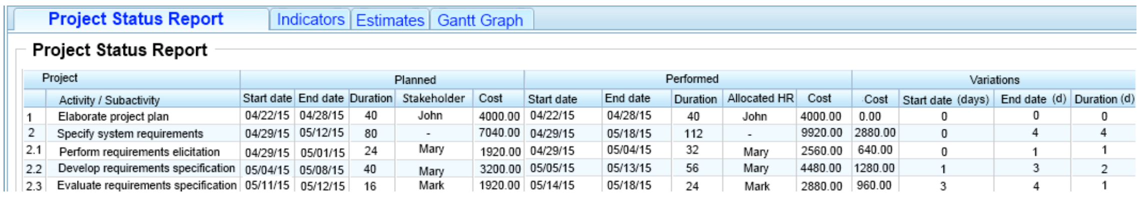

Simplified Project Plan | Gantt Graph

Simplified Project Plan

Project		Planned						
	Activity / Subactivity	Start date	End date	Duration	Stakeholder	Predecessor activities	WBS items related	Cost
1	Elaborate project plan	04/22/15	04/28/15	40	John	-	1.1	4000,00
2	Specify system requirements	04/29/15	05/12/15	80	-	1	-	7040,00
2.1	Perform requirements elicitation	04/29/15	05/01/15	24	Mary	1	2.1	1920,00
2.2	Develop requirements specification	05/04/15	05/08/15	40	Mary	1, 2.1	2.2	3200,00
2.3	Evaluate requirements specification	05/11/15	05/12/15	16	Mark	1, 2.1, 2.2	2.2	1920,00

Figure 12 - Simplified Project Plan fragment.

During the project, the manager submits new versions of the semantic documents containing data regarding project execution. Based on their content and considering SPMO conceptualization, IMSD-SPM provides a *Project Monitoring Panel* presenting planned and actual values and the differences between them. A Gantt Graph showing planned and performed values is also shown. Figure 13 presents a Project Monitoring Panel fragment. For sake of legibility, some columns were omitted. The variations between planned and performed values are presented in the last columns.


Project Status Report | Indicators | Estimates | Gantt Graph

Project Status Report

Project		Planned					Performed					Variations			
	Activity / Subactivity	Start date	End date	Duration	Stakeholder	Cost	Start date	End date	Duration	Allocated HR	Cost	Cost	Start date (days)	End date (d)	Duration (d)
1	Elaborate project plan	04/22/15	04/28/15	40	John	4000.00	04/22/15	04/28/15	40	John	4000.00	0.00	0	0	0
2	Specify system requirements	04/29/15	05/12/15	80	-	7040.00	04/29/15	05/18/15	112	-	9920.00	2880.00	0	4	4
2.1	Perform requirements elicitation	04/29/15	05/01/15	24	Mary	1920.00	04/29/15	05/04/15	32	Mary	2560.00	640.00	0	1	1
2.2	Develop requirements specification	05/04/15	05/08/15	40	Mary	3200.00	05/05/15	05/13/15	56	Mary	4480.00	1280.00	1	3	2
2.3	Evaluate requirements specification	05/11/15	05/12/15	16	Mark	1920.00	05/14/15	05/18/15	24	Mark	2880.00	960.00	3	4	1

Figure 13 – Project Monitoring Panel fragment.

For calculating deviations between planned and actual values, the traceability between estimated processes and their executions and between estimated activities and their executions is explored in SPMO. The traceability is established by the relationships *caused by* between *Intended Process* and *Performed Process,* and between *Intended Activity* and *Performed Activity*. These relationships allow identifying start and end planned dates for processes and activities (*estimatedStartDate* and *estimatedEndDate* of *Scheduled Process* and *Scheduled Activity*)*;* their planned duration and cost (*estimatedDuration* and *estimatedCost* of *Estimated Process* and *Estimated Activity*); their actual start and end date (*startDate* and *endDate* of *Performed Process* and *Performed Activity*); and their actual duration and cost (*duration* and *cost* of *Tracked Process* and *Tracked Activity*).

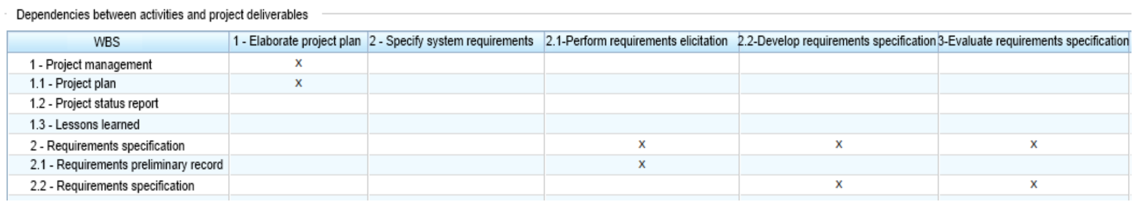
Since planned and actual costs of activities are related to, respectively, stakeholders' allocations and stakeholders' participations, the *refers to* relationship between *Tracked Stakeholder Participation* and *Estimated Stakeholder Allocation* was also explored. This relation allows relating a participation to the referred allocation and, then, comparing planned values (*estimatedDuration* and *estimatedCost properties* of *Estimated Stakeholder Allocation*) with actual values (*duration* and *cost* of *Tracked Stakeholder Participation*).

4.2.2 Dependency Matrices

Dependency relations between activities and between them and work packages are important in the project management context and can be represented by means of dependency matrices. However, a lot of human effort is usually required to create and maintain these matrices.

In the semantic documentation context, relations can be established in semantic templates by using semantic annotations. Thus, data recorded in the semantic documents are captured by IMSD-SPM and the changes made in the dependencies along the project are propagated when updated semantic documents are submitted to IMSD-SPM. Since a semantic global model of the semantic repository is generated by IMSD-SPM, the dependencies established in each semantic document, as well the ones between different documents, can be known.

Dependency relations established in semantic templates and in its instances (semantic documents) come from the relation *depends on* between *Intended Activities*, and the relation *is to produce* between *Estimated Activity* and *Work Package*. Figure 14 presents a fragment of a dependency matrix indicating dependencies between WBS items (first column) and project activities.

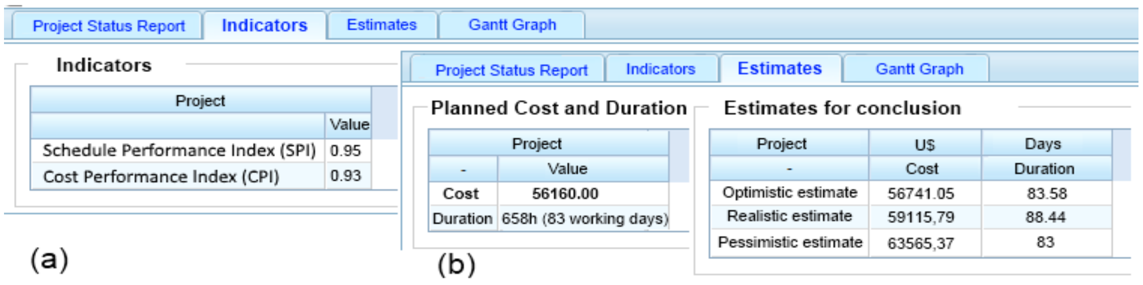

Dependencies between activities and project deliverables

WBS	1 - Elaborate project plan	2 - Specify system requirements	2.1-Perform requirements elicitation	2.2-Develop requirements specification	3-Evaluate requirements specification
1 - Project management	x				
1.1 - Project plan	x				
1.2 - Project status report					
1.3 - Lessons learned					
2 - Requirements specification			x	x	x
2.1 - Requirements preliminary record			x		
2.2 - Requirements specification				x	x

Figure 14 - Dependency Matrix fragment.

4.2.3 Performance Indicators, Estimates for Project Conclusion and Project Performance History

The Project Monitoring Panel allows managers to analyze in details differences between planned and actual values. However, it is also important to understand the project performance as a whole. Thus, IMSD-SPM calculates the Schedule Performance Index (SPI) and the Cost Performance Index (CPI), indicators that quantitatively represent the project performance. SPI and CPI are calculated from three variables: Planned Value (cost planned in the project baseline to the work to be done until a certain date), Earned Value (planned cost to the work done until that date), and Actual Cost (actual cost of the work done) (Fleming and Koppelman 1999). Considering SPMO conceptualization, Planned Value is the sum of planned costs of activities (*estimatedCost* property of *Estimated Activity*) whose *plannedEndDate* of *Scheduled Activity* is smaller than or equal to the considered date for calculating the indicators. Earned Value is the sum of planned costs of activities (*estimatedCost* property of *Estimated Activity*) that caused the *Performed Activities* whose *endDate* is smaller than or equal to the considered date. Actual Cost, in turn, is the sum of actual costs of activities (*cost* properties of *Tracked Activity*) whose *endDate* is smaller than or equal to the considered date. From SPI and CPI, it is possible to determine time and cost estimates for the project conclusion. IMSD-SPM provides optimistic, realistic and pessimistic estimates (Fleming and Koppelman 1999). Figure 15 illustrates fragments of IMSD-SPM screens showing values calculated to (a) SPI and CPI, and (b) estimates.



Indicators

Project	
	Value
Schedule Performance Index (SPI)	0.95
Cost Performance Index (CPI)	0.93

Planned Cost and Duration

Project	
-	Value
Cost	56160.00
Duration	658h (83 working days)

Estimates for conclusion

Project	US	Days
-	Cost	Duration
Optimistic estimate	56741.05	83.58
Realistic estimate	59115,79	88.44
Pessimistic estimate	63565,37	83

Figure 15 – Screens showing (a) SPI and CPI, and (b) Estimates for Conclusion.

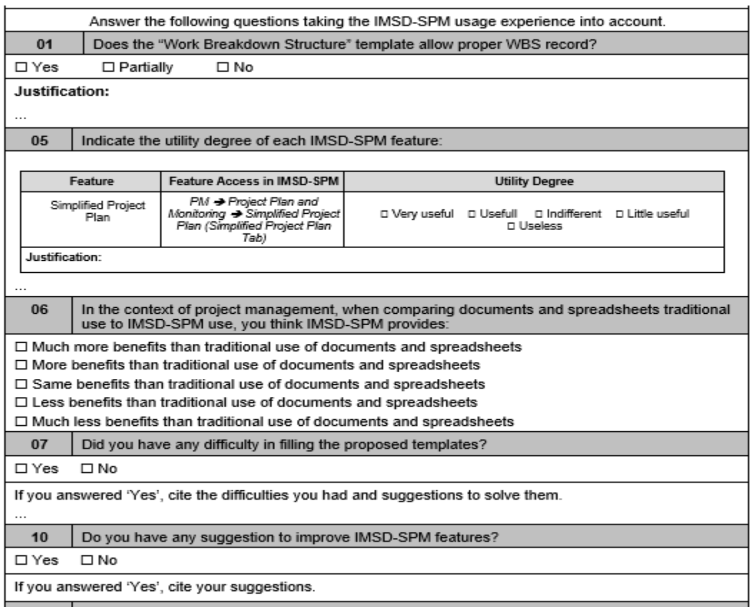
The project performance can change over time. Thus, a performance history graph showing performance indicators over the time helps the manager to understand the project performance behavior, identify performance improvements or deteriorations, investigate causes, and make decisions. The Project Performance History provides a graph of SPI and CPI values over time and also displays the history of optimistic, realistic and pessimistic estimates.

5. IMSD-SPM Evaluation

An experimental study was carried out to serve as a first evaluation of IMSD-SPM. The purpose of the study was to verify if IMSD-SPM is able to properly support scope, time and costs management activities. For this, two indicators were considered: (a) template adequacy and (b) feature utility. Eight individuals participated in the study. All of them had theoretical knowledge and practical experience in project management. Seven of them were graduate students and one was an undergraduate student.

During the study, the researcher presented information related to a project. Based on that, each participant planned the project scope, schedule and cost and created WBS, PSR and HRC semantic documents from semantic templates. The semantic documents were submitted to IMSD-SPM and the participants used the features related to project planning and quality assurance. Then, some situations concerning the project execution were introduced by the researcher (e.g., someone in the project got sick and had to be replaced; the supplier did not deliver material on time impacting on project activities dates; some activities took more than the planned time, etc.). The participants updated the semantic documents with data about project execution, submitted them to IMSD-SPM, and used features related to project monitoring and control, quality assurance and comparison between projects. Updated versions of the semantic documents were created and submitted until data regarding execution of all project activities (i.e., until project conclusion) had been provided.

After using IMSD-SPM, the participants were asked to provide feedback about the templates and IMSD-SPM features. For each semantic template, the participants were asked to evaluate its adequacy. For each feature provided by IMSD-SPM, the participants were asked to evaluate its utility. Figure 16 shows a fragment of the form used as an instrument to get the participants perceptions.



Answer the following questions taking the IMSD-SPM usage experience into account.

01	Does the "Work Breakdown Structure" template allow proper WBS record?

☐ Yes ☐ Partially ☐ No

Justification:

...

05	Indicate the utility degree of each IMSD-SPM feature:

Feature	Feature Access in IMSD-SPM	Utility Degree
Simplified Project Plan	*PM ➔ Project Plan and Monitoring ➔ Simplified Project Plan (Simplified Project Plan Tab)*	☐ Very useful ☐ Usefull ☐ Indifferent ☐ Little useful ☐ Useless
Justification:		

...

06	In the context of project management, when comparing documents and spreadsheets traditional use to IMSD-SPM use, you think IMSD-SPM provides:

☐ Much more benefits than traditional use of documents and spreadsheets
☐ More benefits than traditional use of documents and spreadsheets
☐ Same benefits than traditional use of documents and spreadsheets
☐ Less benefits than traditional use of documents and spreadsheets
☐ Much less benefits than traditional use of documents and spreadsheets

07	Did you have any difficulty in filling the proposed templates?

☐ Yes ☐ No

If you answered 'Yes', cite the difficulties you had and suggestions to solve them.

...

10	Do you have any suggestion to improve IMSD-SPM features?

☐ Yes ☐ No

If you answered 'Yes', cite your suggestions.

Figure 16 – Evaluation form used in the study.

Concerning the semantic templates, most of the participants answered *yes* to the related questions, i.e., they evaluated the templates as *adequate*. Some participants answered *partially* to these questions, meaning that they considered the templates *partially adequate*. In the cases of PSR and HRC, the comments presented by participants who considered the templates partially adequate were outside the scope of this work (e.g., record information related to human resources competencies and abilities) or referred to operational improvements (e.g., automatic data generation). As for the WBS template, the reasons given by the participants who considered the template partially adequate were related to lack of graphic representation.

Regarding IMSD-SPM features, five out the eight features listed in Table 7 were evaluated as *very useful* or *useful* by 100% of participants, namely: Dependency Matrices, Project Monitoring Panel, Project Performance Indicators, Estimates for Project Conclusion and Project Performance History. The other 3 were considered *useful* or *very useful* by at least 75% of participants.

When asked to compare the benefits of using ISDM-SPM versus the ones using traditional documents, 63% of the participants answered that ISDM-SPM provides *much more* benefits and 37% answered that it provides *more* benefits.

No participant reported difficulties in filling in the templates. Nevertheless, one of them pointed that in large projects, with many activities, the use of documents and spreadsheets can be unproductive and increase automatic data recording would be useful. As improvement suggestions, the participants cited: (i) provide automatic alerts regarding the project performance based on SPI and CPI deviations to indicate to the project manager the deviation degree (e.g., critical, regular, small); (ii) provide comparisons considering other project aspects; (iii) improve automatic data recording; and (iv) export reports in different formats.

The study results provide indications that the templates are adequate and the features are useful, what can be seen as initial evidence that the use of IMSD-SPM as a way to support software project time, scope and cost management is feasible. However, as in any study, there are some threats to validity that must be considered together with the obtained results.

In this study we considered threats to the internal, external, construct, and conclusion validities (Barros and Neto 2011). Internal validity regards the capacity of a new study to repeat the behavior of the current study with the same participants and objects. The main threat to internal validity is communication and information sharing among participants. To treat this threat, participants were instructed not to exchange information while using IMSD-SPM and filling the evaluation form. External validity refers to the capacity of repeating the same behavior considering other participants. The main threat, in this case, is the participants' homogeneous profile. It was not possible to treat this threat. Concerning construct validity, which refers to the relationship between instruments, participants, and the theory being evaluated, the main threat comes from the fact that the study was not conducted in the context of a real project. To minimize the impacts of this threat, a project as similar as possible to a real one (it was based on a real project) was used. Finally, with respect to conclusion validity, i.e., the capacity of the study results generate conclusions, the small number of participants, the sample homogeneity, and the use of a controlled environment impact on the study results. Thus, the obtained results cannot be generalized and must be understood as preliminary results that represent some evidence, but that are not conclusive. In this sense, the preliminary results must be confirmed (or not) in additional studies, involving more participants, with different profiles. In addition, other studies types (e.g., case studies) should be carried out to contribute to a better evaluation.

6. Related Works

As said in the Introduction of this paper, at the beginning of this work we performed a systematic literature review aiming at identifying and analyzing initiatives applying semantic annotation to support project management. Five works were identified (Bastos et al. 2015):

[1] *Semantic Annotation based on Software Knowledge Sharing Space* (Lu et al. 2008): It is a system that aims to improve knowledge sharing among software development team members. It allows annotating documents produced during projects, creating a network that facilitates accessing and sharing information about the project.

[2] *Content Management for Inter-Organization Projects* (Nakatsuka and Ishida 2006): It is a system to manage content of inter-organizational projects. Project content is semantically annotated, and when a project member creates, modifies or manages content in a project, automatic emails are sent to the other project members, communicating explicitly what has changed in the project.

[3] *Collaboration in Public Policy Making, Implementation and Evaluation* (Loukis 2007): It consists of a structured electronic forum in which participants opine about programs, projects, tasks and deliverables related to public policies. A Public Policy Ontology is used for semantically annotating posts, allowing for organization, indexing, integration and querying of the posts recorded in the forums.

[4] *Semex* (Talas et al. 2011)**:** It is a module of a project management system. It is responsible for semantic annotation of wiki pages. It supports creation, sharing and publication of collaborative content in projects, providing a common environment that allows project team members to access information and contribute to discussions.

[5] *Use of Semantic Wiki as a Capturing Tool for Lessons Learned* (Elkaffas and Wagih 2013)**:** It uses a semantic wiki as a tool to capture lessons learned in projects, supporting project managers in project initiation and closure. Information is get by using semantic queries.

There are some similarities between our work and those we found in the systematic review. However, there are also differences. Table 8 shows an overview of the main characteristics of our proposal and the ones cited above.

As for similarities, like IMSD-SPM, four proposals use domain ontologies as a basis for annotations and provide general features for managing semantic content (annotation, storage, indexing and retrieving). Only [5] does not use domain ontology, but what its authors call knowledge model (a conceptual reference model).

There are also similarities in the used technologies. IMSD-SPM was built using Java programming language, Postgres database and Subversion version control system. Ontology design and implementation were made using Web Ontology Language (OWL). The SPARQL language was used to perform queries on ontologies and semantic repositories. Some of these technologies were also used in other proposals found in the systematic literature review, such as Subversion to versioning control ([4]) and Resource Description Framework (RDF), which is similar to OWL, to data descriptions and modeling ([2] and [4]).

The main differences between our proposal and the ones found in the systematic review concern the types of annotated files, the annotation type, and the project management knowledge areas supported. Regarding types of files, the proposals annotate web pages, electronic forums, pdf and text documents. IMSD-SPM also annotates text documents, like [1] and [2], but it is the only one to annotate spreadsheets. As for annotation type, all initiatives adopt manual annotation. IMSD-SPM, in contrast, uses semantic templates to semi-automatically annotate documents and spreadsheets created by users. By doing this, users do not need to annotate files, because annotated instances of documents are generated from annotated templates.

Table 8 - Characteristics of semantic annotation proposals supporting Project Management.

Aspect / *Proposal*	*Involved Technologies*	*Annotated Documents Types*	*Ontology Type*	*Ontology Purpose*	*Semantic Annotation Type*	*Knowledge Areas*
[1]	Plug-in for annotations, service provider, database	Word,Eclipse,VS.Net and Adobe Reader Documents	Domain	Semantic Annotation	Manual	Communication
[2]	RDF and XML-RPC	Web pages, text and pdf documents	Domain	Semantic Annotation	Manual	Communication and Integration
[3]	Electronic forum and XML	Electronic forum pages	Domain	Semantic Annotation	Manual	Scope and Stakeholder
[4]	RDF, RDFLib, wikis, Subversion and modules to bug tracking	Wiki pages	Domain	Semantic Annotation	Manual	Communication
[5]	Semantic MediaWiki	Wiki pages	Not applicable	Not applicable	Manual	Integration
[6]	Java, *Postgres* Subversion, OWL, SPARQL	ODF Spreadsheets and text documents	Domain	Semantic annotation	Semi-automatic (based on *templates*)	Scope, Time and Cost

Regarding supported knowledge areas, the proposals found in the systematic review support aspects related to Scope, Integration, Communication and Stakeholder Management. Communication Management covers communication planning (definition of what information should be available; how, when and where it should be recorded; who is responsible for recording it; and who can access it), management (communication plan execution) and controlling (comparison between planned and executed, and corrective actions execution). Three proposals support this area, mainly in aspects related to communication management, which occurs during the project execution phase. In [1], semantic annotation helps information recording and sharing. For instance, documents produced during the project can be annotated and related one to others in a network. As a result, when a document is accessed by a project member, he/she also gets its related documents. In [4], a common knowledge base is shared among projects and supports information sharing. Semantic annotation allows browsing pages containing project content and selecting information related to the projects (e.g., projects that share a certain human resource). [2] supports project content creation, modification and management, and sends automatic emails to project members communicating the changes made. By doing this, [2] also supports aspects related to Integration Management that includes, among others, integrated change control, consisting of recording the project changes, their reasons, and performing the necessary actions in an integrated way.

Integration Management is also supported by [5]. The proposal focus is to capture learned lessons, which are recorded during project closure, in Integration Management (PMI 2013). Considering that learned lessons can support project planning and corrective actions identification, other knowledge areas can also be supported by the proposal. For instance, learned lessons can help risk identification.

[3] supports Scope and Stakeholder Management aspects. Scope Management concerns the definition of the work to be done in the project, while Stakeholder Management involves identifying and managing project stakeholders, their expectations and involvement. The forum proposed in [3] is used to define the public policies and requirements to be addressed in projects, i.e., the project scope. Moreover, the forum helps interact with stakeholders, encouraging the appropriate involvement of them in project activities.

Although all the investigated proposals support management aspects, only two of them ([4] and [5]) were conceived to focus on project management. IMSD-SPM is also devoted to project management, particularly to software project management. Differently from the other proposals, it deals with aspects related to Scope, Time and Cost Management. Thus, IMSD-SPM differs from the found proposals mainly due to the features to support project management activities, obtained by exploring the SPMO conceptualization in functionalities that help managers to plan, monitor and control projects. Although the other proposals support some project management aspects, the domain ontologies used by them do not address aspects that allow comparing project planning and execution. Also, no proposal provides indicators or estimates to help project managers to monitor projects performance. Summarizing, by exploring the

SPMO conceptualization, domain-specific features are provided by IMSD-SPM, contributing to better support project management activities.

7. Final Considerations

Text documents and spreadsheets are often used in the context of project management to record information, which, usually, is distributed in several documents. Being able to properly retrieve, integrate, manage, and analyze information recorded in documents is a critical success factor for organizations. Semantic documentation applies semantic web technologies to desktop documents aiming to improve automatic reasoning, support efficient information management and satisfy users information needs (Leifler and Eriksson 2009).

Although some tools have been proposed to support semantic documentation, the use of this approach in the project management domain is a recent research topic and can be further explored. In this work, we explored the use of semantic documentation in the software project management domain, particularly in the context of scope, time and cost management. We extended IMSD (Arantes and Falbo 2010) making it able to extract information from spreadsheets, and specialized IMSD providing features that take the conceptualization provided by the Software Project Management Ontology (SPMO) into account to support software project management aspects. In IMSD-SPM, semantic documents are created from semantic templates. Thus, users do not need to be directly involved in annotating documents.

The two main contributions of this work are SPMO, which provides a conceptualization about the software project management domain, and IMSD-SPM, which supports a semantic documentation approach for the software project management. SPMO was used in this work as a basis for semantic annotations and domain-specific features. However, its applications are not limited to those. SPMO can serve as a reference model for semantic interoperability initiatives and also can provide a common vocabulary and conceptualization for people talk about the software project management domain. Besides, since SPMO is a networked ontology of SEON, it is integrated to other ontologies, providing a broader and integrated conceptualization regarding the software engineering domain. People interested in addressing not only the software project management domain, but also other processes it relates to, can recur to SEON and extract the fragment of interest.

IMSD-SPM is also a relevant contribution. It helps software organizations to get consolidated information from data recorded in different documents. This contributes to well-informed decision making. It is important to point out that problems to retrieve information from documents and spreadsheets could be also addressed by approaches such as corporative architectures and information systems. However, although these approaches are capable of dealing with information recording and extraction, they imply in modifying the way organizations and people perform activities, since they replace documents by information systems. For organizations that use documents and spreadsheets, using IMSD-SPM can be an advantageous approach, because it accesses information recorded in the documents, allowing organizations and people to keep the way they perform activities.

IMSD-SPM provides some features that are also present in traditional project management tools (e.g., create a simplified project plan), but it also provides features that are not provided by those tools. For example, IMSD-SPM allows managers to compare performance from different projects, provides graphical information about the project performance over time, calculates estimates for project conclusion considering the current project performance, and also automatically checks the quality of documents. Thus, even organizations using traditional project management tools could benefit from the IMSD-SPM features. Aiming to allow these organizations to use IMSD-SPM features, recently we integrated a software project management tool (DotProject) to IMSD-SPM. As a result, information recorded in this tool is captured by IMSD-SPM and managers can use both tools in an integrated way. Moreover, it is worth noticing that, even though IMSD-SPM is proposed to the software project management domain (SMPO was developed from a software process ontology), it can also be applied to other domains.

As a limitation of this work, we highlight its evaluation. As discussed in Section 5, the evaluation counted on few participants with homogeneous profile and was performed in a controlled environment. As a consequence, the results cannot be generalized and should be understood as preliminary results. Other studies, including case studies in real organizations, are necessary to better evaluate the proposal. In this sense, as future works, we plan to carry out other studies to evaluate the proposal. We also intend to enable IMSD to annotate images. As a result, it will be possible, for example, to represent WBS graphically and annotate it.

Finally, considering that SPMO is a networked ontology of SEON and that ontology networks are continuously evolving, the current SPMO version addressed in this paper can be evolved. So far, we have focused on the aspects of the software project management domain necessary to our semantic documentation approach. New concepts related to methods, indicators and other project management aspects can be added to SPMO. Additionally, SPMO can be extended to cover other project management areas and specialized to address project management aspects in more specific contexts, such as agile projects. By evolving SPMO it will be possible to explore the use of semantic documentation to support other project management areas and contexts.

References

Anantatmula, V., & Rad, P. Linkages among project management maturity, PMO, and project success. In *International Conference on Engineering, Technology and Innovation (ICE) & IEEE International Technology Management Conference, Netherlands, 2013.*

Arantes, L. O., & Falbo, R. A. An infrastructure for managing semantic documents. In *Joint 5th International Workshop on Vocabularies, Ontologies and Rules for The Enterprise (VORTE) - International Workshop on Metamodels, Ontologies and Semantic Technologies (MOST). 2010.*

Bastos, E., Barcellos, M. P., & Falbo, R. A., Exploring Ontologies for Semantic Documentation in Project Management. In *Ontobras, Brazil,* 2015.

Bastos, E., Barcellos, M. P., & Falbo, R. A., Semantic Documentation in Project Management. In *XV Brazilian Symposium on Software Quality, Brazil,* 2016.

Barros, M. D. O., & Neto, A. C. D. (2011). Threats to Validity in Search-based Software Engineering Empirical Studies. Rio de Janeiro: Universidade Federal do Estado do Rio de Janeiro.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001) The semantic web. Scientific American 284 (5), 2001, 34–43.

Brank, J., Grobelnik, & M., Mladenić, D., A Survey of Ontology Evaluation Techniques, In *Conference on Data Mining and Data Warehouses* (SiKDD 2005), 2005.

Bringuente, A.C., Falbo, R.A., Guizzardi, G. (2011). Using a Foundational Ontology for Reengineering a Software Process Ontology. Journal of Information and Data Management, v. 2, n. 3, pp. 511.

CMMI (2010). CMMI for Development, Version 1.3. Software Engineering Institute.

D'Aquin, M., & Gangemi, A. (2011). Is there beauty in ontologies? *Journal Applied Ontology, 6*(3), 165-175.

Elkaffas, S. M., & Wagih, A. S. Use of Semantic Wiki as a Capturing Tool for Lessons Learned in Project Management. In *Science and Information Conference, London, UK, 2013.*

Eriksson H., & Bang M. Towards document repositories based on semantic documents. In *I-KNOW '06*, 2006.

Eriksson, H. (2007). The semantic-document approach to combining documents and ontologies. *International Journal of Human-Computer Studies, 65*(7), 624-639.

Falbo, R. A. SABiO: Systematic Approach for Building Ontologies. In *1st Joint Workshop ONTO.COM / ODISE on Ontologies in Conceptual Modeling and Information Systems Engineering, 2014.*

Falbo, R. A., Barcellos, M. P., Nardi, J. C., & Guizzardi, G. Organizing Ontology Design Patterns as Ontology Pattern Languages. In *10th European Semantic Web Conference – ESWC fRANCE, 2013* (pp. 61 - 75).

Falbo, R. A., Braga, C. E. C., & Machado, B. N. Semantic Documentation in Requirements Engineering. In *17th Workshop on Requirements Engineering (WER 2014), Pucón, Chile, 2014.*

Fleming, Q. W., & Koppleman, J. M. (1999). *Earned value Project Management* (2ed.). Pennsylvania: Project Management Institute.

Graaf, K. A. D., Tang, A., Liang, P., & Vliet, H. V. Ontology-based software architecture documentation. In *Joint 10th Working IEEE/IFIP Conference on Software Architecture & 6th European Conference on Software Architecture (WICSA/ECSA), Helsinki, Finland, 2012* (pp. 121-130).

Guizzardi, G. (2005). *Ontological Foundations for Structural Conceptual Models*. University of Twente, Enschede.

Guizzardi, G. (2007). On Ontology, ontologies, Conceptualizations, Modeling Languages and (Meta)Models. In J. E. Olegas Vasilecas, Albertas Caplinskas (Ed.), *Frontiers in Artificial Intelligence and Applications, Databases and Information Systems IV*. Amsterdam: IOS Press.

Guizzardi, G., Falbo, R. A., & Guizzardi, R. S. S. Grounding software domain ontologies in the Unified Foundational Ontology(UFO): the case of the ODE software process ontology. In *XI Iberoamerican Workshop on Requirements Engineering and Software Environments, Recife, Brazil, 2008* (pp. 244-251).

Hillson, D. (2003). Assessing organizational project management capability. *Journal of Facility Management, 2*(3), 298 - 311.

ISO/IEC (2008). Information Technology - Software Lifecycle Processes. Geneva, Switzerland.

Jessen, S. The impact on project success of using technology in modern project planning and control. In *International Technology Management Conference, California, EUA, 2011* (pp. 944 - 948).

Leifler, O., & Eriksson, H. Domain-specific Knowledge Management in a Semantic Desktop. In *I-KNOW '09 9th International Conference on Knowledge Management and Knowledge Technologies, Graz, Austria, 2009* (pp. 360 - 365).

Lethbridge, T. C., Singer, J., & Forward, A. (2003). How Software Engineers Use Documentation: The State of the Practice. *IEEE Software, 20*(6), 35 - 39.

Loukis, E. N. (2007). An ontology for G2G collaboration in public policy making, implementation and evaluation. *Artificial Intelligence and Law, 15*(1), 19 - 48.

Lu, Q., Chen, M., & Wang, Z. A semantic annotation-based software knowledge sharing space. In *IFIP International Conference on Network and Parallel Computing (NPC), China, 2008.*

Nakatsuka, K., & Ishida, T. Content management for inter-organizational projects using e-mail metaphor. In *International Symposium on Applications and the Internet (SAINT), Phoenix, Arizona, USA, 2006* (pp. 202 - 205).

Nesic, S. (2010). *Semantic Document Architecture for Desktop Data Integration and Management*. University of Lugano, Lugano, Switzerland.

OASIS (2017). Open Document Format for Office Applications. https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=office. Accessed 18 april 2017.

PMI (2013). *A guide to the Project Management Body of Knowledge (PMBoK)* (5ed.). Pennsylvania: Project Management Institute.

Popov, B., Kiryakov, A., Kirilov, A., Manov, D., Ognyanoff, D., & Goranov, M. KIM- Semantic Annotation Platform. In *2nd International Semantic Web Conference (ISWC2003), Florida, USA, 2003* (pp. 844-849).

Ruy, F. B., Falbo, R. A., Barcellos, M. P., Costa, S. D., & Guizzardi, G. SEON: A Software Engineering Ontology Network. In *20th International Conference on Knowledge Engineering and Knowledge Management (EKAW), Bologna, Italy, 2016* (pp. 527-542).

Sicilia, M. (2006). Metadata, semantics and ontology: providing meaning to information resources. *International Journal of Metadata, Semantics and Ontologies, 1*(1), 83 - 86.

Suárez-Figueroa, M. C., Gómez-Pérez, A., Motta, E., & Gangemi, A. (2012). *Ontology Engineering in a Networked World* (Springer Science & Business Media): Springer-Verlag Berlin Heidelberg.

Talas, J., Gregar, T., & Pitner, T. Semantic wiki in environmental project management. In *IFIP Advances in Information and Communication Technology, 359 AICT, Brno, Czech Republic, 2011* (pp. 437-444).

Tallis, M. Semantic Word Processing for Content Authors. In *Knowledge Markup and Semantic Annotation Workshop, Florida, USA, 2003.*

Uren, V., Cimiano, P., Iria, J., Handschuh, S., Vargas-Vera, M., Motta, E., et al. (2006). Semantic annotation for knowledge management: requirements and a survey of the state of the art. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web, 4*, 14 - 28.

Villalobos, J., Sanabria, S., & Caceres, R. (2011). Activity scheduling through Gantt charts in an ms excel spreadsheet. *Revista Facultad de Ingenieria*(61), 132 - 145.