UDC 004.7, DOI:10.2298/CSIS091110030L

An Approach to Project Planning Employing Software and Systems Engineering Meta-Model Represented by an Ontology

Miroslav Líška[1] and Pavol Navrat[1]

[1]Faculty of Informatics and Information Technologies,
Slovak University of Technology,
Ilkovičova 3, 842 16 Bratislava, Slovakia
liska@semantickyweb.sk, navrat@fiit.stuba.sk

Abstract. Currently, one can witness a growing mutual influence between the Model Driven Architecture (MDA) and the Semantic Web. MDA is an approach that uses models for system development, but its architecture limits usability of these models for knowledge empowered solutions. A lot of research tackles applicability of MDA standards in the technical space of the Semantic Web. In this paper, we present an approach aimed at facilitating the use of Software and Systems Engineering Meta-Model (SPEM) for improvements that are rooted in knowledge engineering approaches. We show how SPEM can be used in the Semantic Web technical space. We describe how following our approach a project plan can be generated and verified. Finally, we present an example of project planning that uses ontology of a software requirements activity.

Keywords: Software and Systems Process Engineering Meta-Model, Web Ontology Language, Model Driven Architecture, Semantic Web.

1. Introduction

Software project management is the art of balancing competing objectives, managing risk, and overcoming constraints to deliver a product that meets the needs of the customers and the end users [1]. Project management is accomplished through the use of processes such as: initiating, planning, executing, controlling and closing [2]. Despite the fact that at present there exist many software developments process frameworks (e.g. Rational Unified Process, Eclipse Process Framework), the fact that relatively few projects are completely successful is an indicator of the difficulty of the task. One of the problems is that standard software development process frameworks are usually used as a navigable websites that contain only human-readable descriptions with supporting materials as documents templates etc. Thus, these kinds of frameworks cannot be used to represent machine interpretable content [3]. Moreover, these process frameworks are used in the technical spaces [4] that have model based architecture, such as MDA or Eclipse

Modeling Framework (EMF) [5]. These kinds of technical spaces also limit knowledge based processing, owing to their weakly defined semantics [6]. However, at present the emerging field of Semantic Web technologies promises new stimulus for Software Engineering research [7].

The Semantic Web is a vision for the future of the Web, in which information is given explicit meaning, making it easier for machines to automatically process and integrate information available on the Web [8]. The today's key Semantic Web technology is Web Ontology Language (OWL). OWL is intended to be used when the information contained in documents needs to be processed by applications, as opposed to situations where the content only needs to be presented to humans [9]. Aforementioned problems in software engineering and facts about the Semantic Web implies an opportunity to support project management with OWL, and thus to support project management with knowledge based techniques. In this work we address such opportunity and propose a method for project plan generation and verification that makes use of an ontology. To achieve it we need to move project planning in to the Semantic Web technical space. According to these requirements we attempt to make use of the potential of combining OWL and Software and Systems Engineering Meta-Model (SPEM).

1.1. Related works

SPEM is MDA standard used to define software and systems development processes and their components [10]. A SPEM process can be systematically mapped to a project plan by instantiating the different process' breakdown structure views. Therefore a SPEM model can represent a knowledge base that can be used for verification, whether a project plan conforms to this knowledge. However, the SPEM metamodel has the semiformal architecture, thus it is not possible to make and to verify created SPEM language statements with formal techniques such as the consistency or satisfiability verification [11]. But if we transform SPEM to the Semantic Web technical space, we can use the mentioned formal techniques due to facilities of OWL. Because SPEM is based on MDA, we can utilize the research results of transforming other MDA's standards to the Semantic Web technical space.

SPEM is specified in the Meta Object Facility (MOF) language that is the key language of MDA. MOF is a language for metamodel specification and it is used for specification of all model-based MDA standards [12]. It provides metadata management framework, and a set of metadata services to enable the development and interoperability of model and metadata driven systems [13]. On the Semantic Web side, OWL is intended to provide a language that can be used to describe the classes and relations between them that are inherent in Web documents and applications. OWL is based on Resource Description Framework Schema (RDFS) [14]. Both MOF and RDFS provide language elements, which can be used for metamodeling. Although they have similar language concepts such as mof:ModelElement with rdf:Resource, or mof:Class with rdf:Class, the languages are not equivalent. RDFS, as a schema layer language, has a non-standard and non-fixed-layer metamodeling architecture, which makes some elements in model to have dual roles in the RDFS specification [15]. MOF is also used for specification of the Unified Modeling Language (UML) that is a language for specification, realization and documentation of software systems [16]. Even if UML and RDFS are similar in a domain of system specification, they are also substantially different. One issue that has been addressed was the problem that RDF properties are first class entities and they are not defined relative to a class. Therefore a given property cannot be defined to have a particular range when applied to objects of one class and another range when applied to objects of a different class [17]. Note this difference have also been propagated between OWL and UML [18]. It should be noted that efforts to transfer explicit knowledge into machine processable form encompass a much wider spectrum of works, e.g. [19, 20]. Still others attempt to develop domain specific languages, incorporating knowledge on the domain, that would be adaptable [21] improving in such a way the process of software evolution [22]. At present the main bridge that connects the Semantic Web with MDA is stated in the Ontology Definition Meta-Model (ODM) [23]. ODM defines the OWL Meta-Model specified in MOF (MOF – OWL mapping) and also the UML Profile for Ontology modeling (UML – OWL mapping). This architecture can be extended with additional mappings between the UML Profile for OWL and other UML Profiles for custom domains [24, 25]. We have already utilized this principle in our previous works where we created an approach to SPEM model validation with ontology [26] and ontology driven approach to software project enactment with a supplier [27]. However, our works are not the only one that concern with using of SPEM in the Semantic Web technical space. In short, the following subsection references to the three other related works.

The first work proposes to represent SPEM in DL [28]. The work creates mapping from MOF to DL and mapping from OCL (OMG, 2006b) constraints of SPEM to DL. The reason for the former mapping is to represent the SPEM MOF based metamodel with DL and the latter is to represent additional OCL constraints that supplement the SPEM metamodel with additional semantics. The second work presents a competency framework for software process understanding [29]. The motive is to create assessments for a correct understanding of a process that can be used in a software development company. The paper introduces creation of SPEM software process ontology for the SCRUM [30] software process with EPF Composer. However, only the third work is the most closest to our approach, since it proposes a project plan verification with ontology. The work intends to use SPEM process constraint definitions with the semantic rules with SWRL [31]. Note that SWRL is W3C Semantic Web Rule Language that combines OWL and RuleML [32].

1.2. Aims and objectives

We aimed in our research to devise a method that allows generation and verification of a project plan with the SPEM Ontology that use OWL-DL reasoning. Besides that, we present an example, where the subject of project planning is a requirements specification activity.

The rest of the paper is structured as follows. Section 2 presents our solution to the problem. First we define a conformance level with the SPEM compliance point and in short we discuss the SPEM transformation to the Semantic Web that we have already created and published. Then we describe the main principles of our approach to project plan generation and verification with ontology. Subsequently, Section 3 presents an example of ontology driven project plan verification with a requirements specification knowledge. Finally, Section 4 provides conclusion and future research direction.

2. An Approach to Ontology based SPEM

The key objective of project planning is to allocate tasks and responsibilities to a team of people over time and to monitor and manage progress relative to project plan [1]. To support this objective with knowledge based techniques, we need to have knowledge about software methods that could be used for project planning. We need to store this knowledge in form that is suitable for knowledge based processing. It ought to be possible to use this knowledge for a project plan generation. Following these requirements, we selected MDA language SPEM since is aimed for software process specification and is capable for process enactment with planning tools such as Microsoft Project by providing the means to map Processes to project plan [10]. Forasmuch as SPEM is not supportive to knowledge based techniques, we propose a way of transforming it into technical space of the Semantic Web.

2.1. Setting SPEM compliance point

SPEM metamodel is MOF-based and reuses UML 2 Infrastructure Library [33]. Its own extended elements are structured into seven main meta-model packages. Above these packages SPEM defines three compliance points (CP) which are: the SPEM Complete CP, SPEM Process with Behavior and Content CP and the SPEM Method Content CP. The scope of our solution is covered with Compliance Point "SPEM Process with Behavior and Content". The reason of this compliance point is because we need to work with separated reusable core method content from its application in processes, because a software method content can be used with arbitrary software process, such as iterative, agile etc. This separation is represented with two SPEM metamodel packages that are the Method Content and the Process with Method metamodel packages. The former provides concepts for SPEM users and organizations to build up a development knowledge base that is independent of any specific processes and development projects. These concepts are the core elements of every method such as Roles, Tasks, and Work Product Definitions etc. The latter necessary metamodel package defines the structured work definitions that need to be performed to develop a system, e.g., by performing a project that follows the process. Such structured work definitions delineate the work to be performed along a timeline or lifecycle and organize it in so called breakdown structures. The most important elements of the Process with Method metamodel package are the Method Content Use elements. These elements are the key concept for realizing the separation of processes from method content. A Method Content Use can be characterized as a reference object for one particular Method Content Element, which has its own relationships and properties. When a Method Content Use is created, it shall be provided with congruent copies of the relationships defined for the referenced content element..

2.2. Moving SPEM into the Semantic Web

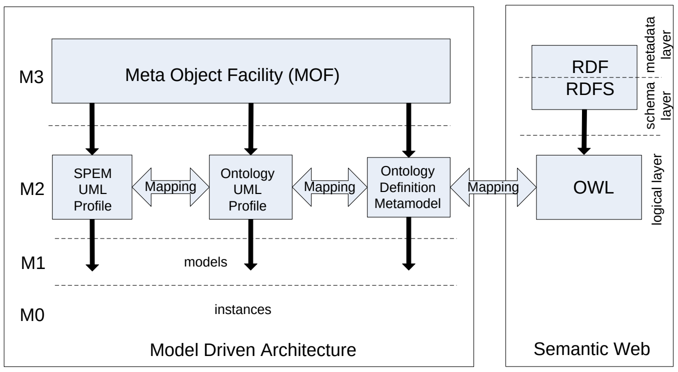


Fig. 1. Mappings between SPEM and OWL

Thus only a mapping between SPEM and OWL has to be created. Since the hallmark work [6] proposes the transformation of a MDA standard to the Semantic Web technical space with a mapping between UML Ontology Profile and an arbitrary UML Profile, we have either used this principle, thus we have created a mapping between the Ontology UML Profile and the SPEM UML Profile as it is shown in Figure 1.

However, the mapping between the Ontology UML Profile and the SPEM UML Profile were not sufficient to create the SPEM Ontology. The main problem was that the SPEM UML Profile does not contain SPEM semantics, and in addition for example, it was not possible to derive a domain and range of a relationship, etc. Therefore we had decided to create semiautomatic transformation that is based on merged SPEM metamodel to the SPEM UML Profile, where the result is the SPEM OWL DL Ontology. For more detailed and comprehensive description about the SPEM transformation to the Semantic Web technical space and its utilizations, a reader may refer to [36, 37]. We have used OWL-DL, because this dialect of OWL retains computational completeness (all conclusions are guaranteed to be computable) and decidability (all computations will finish in finite time) [8]. To be conformed to this dialect, we have to adhere that an individual cannot be either a class, what is not violated in MDA technical space because of its 4 meta-layer architecture. For example, an analyst “Slávko Líška” is an instance of a Software Analyst SPEM class that is an instance of the Role Definition SPEM metaclass at the same time, thus the Software Analyst class is an individual and also a class. To avoid this problem in the Semantic Web technical space we have stated that a method content owl class is subclass of a SPEM owl class, and concrete individual is its instance. For example, the individual “Slávko Líška” is the instance of the Software Analyst owl class that is subclass of the Role Definition owl class from the SPEM Ontology. For the sake of clarity, follow Figure 2 illustrates the mapping in more detail.

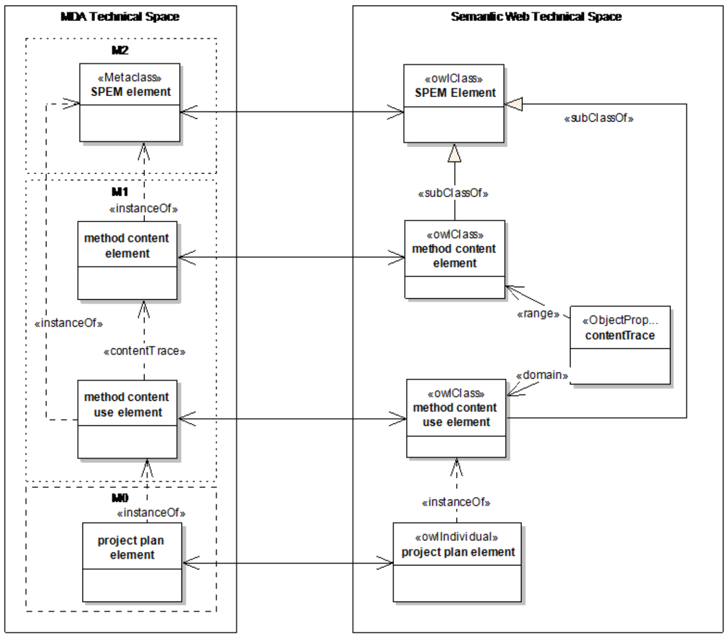


Fig. 2. SPEM in the Semantic Web Technical space

2.3. Project plan generation and verification

As we have already mentioned the SPEM Method Content is intended for a software method specification from the static point of view, whereas the SPEM Process provides concepts for representing method content elements in a process. Therefore once we have created the SPEM Ontology conformed to the SPEM Process with Behavior and Content Compliance Point, we can create ontology for a software method and process. To do so, we have created XSL transformations *SPEMMethodContent2OWL* and *SPEMProcess2OWL*. The former transforms a SPEM method content model to a SPEM method ontology and the latter transforms a SPEM process model to a SPEM process ontology.

At this point we can use OWL DL consistency verification between the SPEM Ontology, a SPEM method ontology and a SPEM process ontology. The first reason is to verify, whether a SPEM method ontology and a SPEM process ontology are correctly specified with the SPEM Ontology, for example whether a Task Definition element is performed only with a Role Definition element or whether a Role Use element is responsible just for a Work Product Use element. The second reason of the OWL DL verification at this point can be to ensure whether a SPEM process ontology is correctly traced to a SPEM method ontology, for example whether a Role Use element traces just a Role Definition element.

Since the scope of SPEM is purposely limited to the minimal elements necessary to define any software and systems development process, the SPEM metamodel does not include elements such as the Iteration, Phase etc. The reason is because not every software development process needs to have iterations for example. Therefore we had to extend the engine for a project plan validation either with the SPEM Base Plugin that is included in the SPEM specification. It provides commonly used concepts for the domain of software engineering such as the Phase, Iteration, Checklist etc.

However, none of mentioned ontologies does represent concrete project plan. To include it to the OWL DL verification we simply use the enactment between SPEM and a project planning system as it SPEM defines. The enactment is based on instantiation relationships between SPEM process elements and project plan elements. Thus we have created additional XSL transformation *MPP2OWL* that transforms a project plan to the individuals of a SPEM process ontology. The complete engine for project planning support with the SPEM ontology we propose is depicted in Figure 3.

When the result of the OWL DL verification is the inconsistency its source should be removed. In our case the inconsistency should be removed from the project plan. Finally, since a project plan can be mapped to a SPEM process we have created an additional XSL transformation *SPEMProcess2MPP*. The transformation generates a project plan from a SPEM process ontology. So, as it is shown in Figure 3 the OWL DL reasoning is based on the consistency verification between the SPEM Ontology, SPEM Base Plugin Ontology, a SPEM method ontology, a SPEM process ontology and SPEM process individuals obtained from a project plan. Such solution provides two major utilization scenarios:

- Scenario 1. **Project plan generation with ontology**. When a project manager want to create a project plan, he can create a SPEM method and process models first and then use OWL DL consistency reasoning to ensure that they are consistent. Then he can just simply transform his SPEM process model to the SPEM process ontology.
- Scenario 2. **Project plan verification with ontology**. This scenario is essential when a project manager wants to ensure that his already created project plan is consistent with desired method content and process. However, this second scenario usually follows upon the first. A project manager obviously makes many changes to his project plan; therefore it is necessary to ensure that these changes do not break the required consistency.

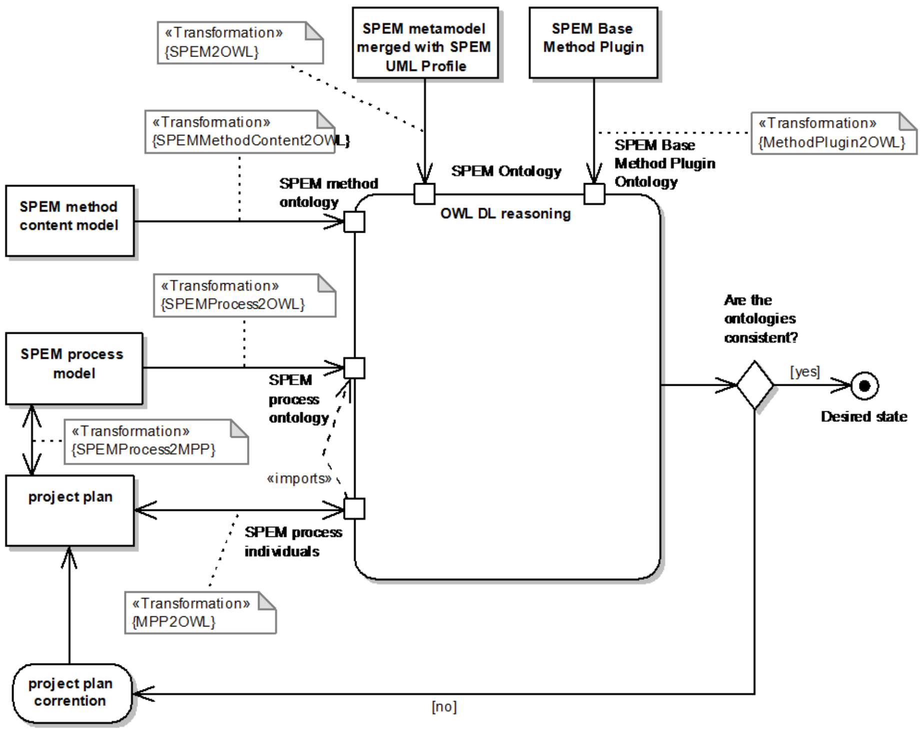

«Transformation» {SPEM2OWL}
SPEM metamodel merged with SPEM UML Profile
SPEM Base Method Plugin
«Transformation» {SPEMMethodContent2OWL}
«Transformation» {MethodPlugin2OWL}
SPEM Ontology
SPEM Base Method Plugin Ontology
SPEM method ontology
SPEM method content model
OWL DL reasoning
«Transformation» {SPEMProcess2OWL}
Are the ontologies consistent?
SPEM process model
[yes]
SPEM process ontology
Desired state
«Transformation» {SPEMProcess2MPP}
project plan
«imports»
SPEM process individuals
«Transformation» {MPP2OWL}
project plan corrention
[no]

Fig. 3. An approach of project planning with SPEM ontology

To be more precise, we give the formally defined conditions that cover the mentioned utilization scenarios. Since the first scenario is included in the second, we focus only on the Scenario 2. First we define the Project Plan Knowledge as the union of the SPEM Ontology, SPEM Base Plugin Ontology, a SPEM method ontology and a SPEM process ontology, as it is shown in Formula 1.

$$\text{Project Planning Knowledge} = \text{SPEM Ontology} \sqcup \text{SPEM Base Plugin Ontology} \sqcup \text{SPEM method content ontology} \sqcup \text{SPEM process ontology} . \quad (\mathbf{1})$$

Then we say, that the Project Planning Knowledge is satisfied in a project plan if it is true that

$$\text{Project Plan} \models \text{Project Planning Knowledge} . \quad (\mathbf{2})$$

From the First Order Logic point of view, the Project Plan Knowledge is the theory and a project plan is its model. Since a theory can have a model only if a theory is consistent [38], it is necessary, that the Formula 3 is either true

$$\text{SPEM Ontology} \vdash \text{SPEM Base Plugin Ontology} \vdash \text{SPEM method content ontology} \vdash \text{SPEM process ontology} . \quad (\mathbf{3})$$

2.4. Implementation

Ontologies rely on well-defined and semantically powerful concepts in artificial intelligence [39], such as description logics, reasoning, and rule-based systems [40]. Since we use OWL DL form of ontology, the implementation has goal to present the proposed utilization scenarios with a Knowledge Representation System that supports description logics. Developing a knowledge base using a description logic language means setting up a terminology (the vocabulary of the application domain) in a part of the knowledge base called the TBox, and assertions about named individuals (using the vocabulary from the TBox) in a part of the knowledge base called the ABox [41]. In other words, the ABox describes a specific state of affairs in the world in terms of the concepts and roles defined in the TBox [6].

Table 1. Mapping between components of a knowledge based representation system to our approach's ontologies

Ontology type	KBRS component
SPEM Ontology	TBox
SPEM Base Method Plugin	TBox
SPEM method content ontology	TBox
SPEM process ontology	TBox
SPEM method plugin ontology	TBox
Individuals of a SPEM process ontology	ABox

As we have it discussed in Subsection 2.2, our approach is conformed to the OWL DL dialect that disallows to an individual to be either a class. Therefore all classes of the ontologies used in our approach constitute TBOX, whereas only individuals obtained from a project plan create ABOX as it is depicted in Table 1.

3. Example of project plan verification with ontology

For the sake of clarity we present an example of a project plan verification with the SPEM ontology. The example shows the OWL DL consistency reasoning of simple Create Requirements Method Content and Create Requirements Process with the project plan that also contains simple plan for requirements specification. The reasoning fully supports the engine presented in Figure 3, therefore the SPEM Ontology, SPEM Base Plugin Ontology, Create Requirements Method Ontology, Create Requirements Process Ontology and the ontology of the project plan are the input ontologies to the reasoning process. Intentionally, the first result of the reasoning is inconsistency, thus the project planning knowledge is not satisfied in the project plan. However, when the origin of the inconsistency is removed, we get desired project plan that is consistent with the project planning knowledge.

Figure 4 presents an excerpt of the SPEM Ontology. The asserted axiom depicted in the figure represents the key concept of SPEM that is the separation of a method content from process.

SPEM Ontology

owlClasses:
  BreakDownElement, MethodContentElement, WorkDefinition…
objectProperties:
  performs, mandatoryOutput, optionalOutput, responsible …
asserted axioms:
  MethodContentElement $\sqcap$ MethodContentUse $\equiv \emptyset$ …

Fig. 4. An excerpt of the SPEM Ontology

For the purpose of this article we do not show an excerpt of the SPEM Base Plugin Ontology, because it just defines additional concepts such as the Iteration, Process etc. which are the specialized classes from the SPEM Ontology classes. Instead, we focus on the Create Requirements Method Content Model that is depicted in Figure 5 in an instant.

The method content model defines role definitions, work product definitions, task definitions and their appropriate relationships which are needed to create software requirements. As it is shown in Figure 2 we have created the XSL transformation MethodContent2OWL that transforms a XML serialization of a method content model to the XML format of the OWL DL ontology. In our case, an excerpt of the resulted Create Requirements Ontology is depicted in Figure 6.

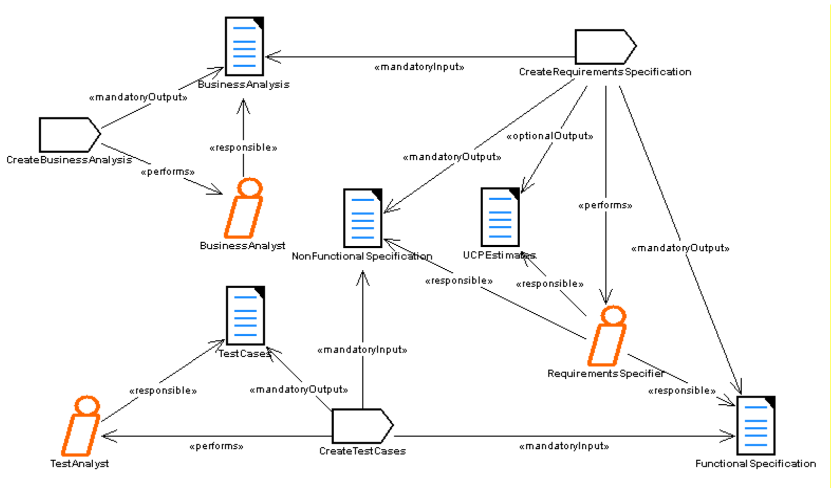


Fig. 5. Create Requirements Method Content Model

Create Requirements Method Ontology
<u>imports</u>:
SPEMOntology
<u>owlClasses</u>:
BusinessAnalyst, RequirementsSpecifier, TestAnalyst …
<u>asserted axioms</u>:
RequirementsSpecifier $\sqsubseteq$ RoleDefinition $\sqcap$
$\forall$ responsible.(FunctionalSpecification $\sqcup$ UCPEstimates $\sqcup$ …

Fig. 6. Create Requirements Method Content Model Ontology

Create Requirements Process Ontology
<u>imports</u>:
CreateRequirementsMethodOntology, SPEMBasePluginOntology
<u>owlClasses</u>:
myProcess, Iteration_I1, CreateRequirements_I1 ……
<u>asserted axioms</u>:
myProcess $\sqsubseteq$ Process,
Iteration_I1$\sqsubseteq$ Iteration $\sqcap$ nestedBreakdown.myProcess,
CreateRequirements_I1 $\sqsubseteq$ nestedBreakdown.Iteration_I1 …

Fig. 7. Create Requirements Process Ontology

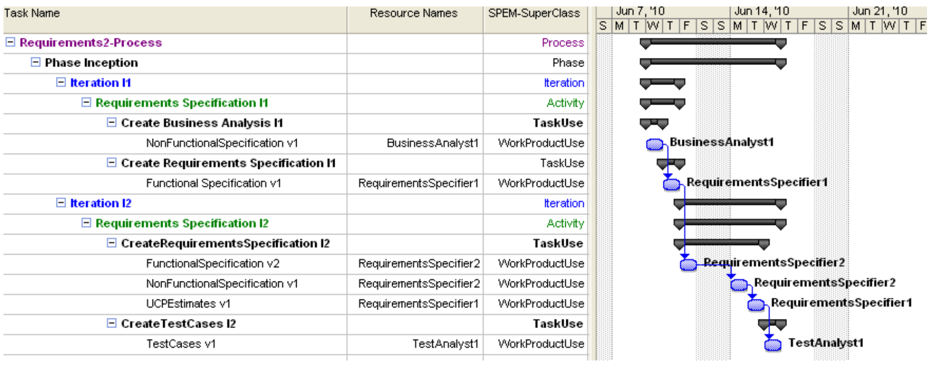
The Create Requirements Ontology imports the SPEM Ontology for the purpose of SPEM classes specialization. For example, the Requirements Specifier class is specialization from the Role Definition SPEM class. However, as it can be seen in Figure 5 or Figure 6, the method content ontology does not represent any dynamics aspect of the create requirements software method. Thus, we have to additionally define even a process. An excerpt of the process is depicted in Figure 7. The excerpt defines that the

Process “myProcess” consists of the Iteration “Iteration_I1” that consists of the Activity “Create Requirements I1”.

Since we have presented the SPEM Ontology, Create Requirements Method Ontology and Create Requirements Process Ontology, we can define the Create Requirements Knowledge, as it is depicted in Formula 4.

Create Requirements Knowledge = SPEM Ontology ⊔ SPEM Base Plugin Ontology ⊔ Create Requirements Method Ontology ⊔ Create Requirements Process Ontology . (**4**)

At this point, all we need for the verification process is a project plan that also defines how to create requirements. The plan is depicted in Figure 8.



Task Name	Resource Names	SPEM-SuperClass
⊟ **Requirements2-Process**		Process
⊟ **Phase Inception**		Phase
⊟ **Iteration I1**		Iteration
⊟ **Requirements Specification I1**		Activity
⊟ **Create Business Analysis I1**		**TaskUse**
NonFunctionalSpecification v1	BusinessAnalyst1	WorkProductUse
⊟ **Create Requirements Specification I1**		TaskUse
Functional Specification v1	RequirementsSpecifier1	WorkProductUse
⊟ **Iteration I2**		Iteration
⊟ **Requirements Specification I2**		Activity
⊟ **CreateRequirementsSpecification I2**		**TaskUse**
FunctionalSpecification v2	RequirementsSpecifier2	WorkProductUse
NonFunctionalSpecification v1	RequirementsSpecifier2	WorkProductUse
UCPEstimates v1	RequirementsSpecifier1	WorkProductUse
⊟ **CreateTestCases I2**		**TaskUse**
TestCases v1	TestAnalyst1	WorkProductUse

Fig. 8. Project Plan 1X

The project plan is created with the MS Project Plan that allows saving the plan into the XML format. Thus we can use our XML transformation MPP2OWL that transforms a project plan to the individuals of the Create Requirements Process Ontology. An excerpt of the resulted project plan ontology Project Plan 1X Ontology is depicted in Figure 9. Note that the depicted instantiation represents the enactment of SPEM with a project planning system.

Project Plan 1X Ontology

imports:
CreateRequirementsProcessOntology
individuals:
Requirements2-Process, PhaseInception …
instantiation:
Process(“myProcess”),
BusinessAnalyst_I1(“BusinessAnalyst1”),
BusinessAnalyst_I1(“RequirementsSpecifier1”) …
object property assertions:
responsible(“BusinessAnalyst1”,” NonFunctionalSpecification_I1”) …

Fig. 9. An excerpt of the Project Plan 1X Ontology

So, when we execute the OWL verification at this point, we find out that the result is inconsistency. This means that the project plan is not properly defined with regard to the Create Requirements Knowledge, or by the more precise words, that the project planning knowledge is not satisfied in the project plan. It is true that

ProjectPlan1X |/= Create Requirements Knowledge . (**5**)

The source of inconsistency lies in Figure 7, where is it intentionally stated that the Role Definition “Business Analyst 1” is responsible for the Work Product Definition “NonFunctional Specification”. The assertion is inconsistent with the Create Requirements Method Ontology, where it is stated, that the Role Definition “Requirements Specifier” is responsible for the work product definition. Thus, after this inconsistency is removed, it is true that

ProjectPlan1 |= Create Requirements Knowledge . (**6**)

4. Conclusion

We presented our approach to project plan generation and verification with the SPEM Ontology. When we compare our approach with the most similar work [29] we conclude that we created not only wider method specification, but we have also presented its implementation. Moreover, we presented the engine for project plan verification with ontology that supports key property of SPEM that is the Method Content separation from a Process and either the separation from the SPEM Base Plugin. Additionally, since a Method Plugin consists of a Method Content and a Process, our approach can be easily extended with any Method Plugin, for example, with the Rational Unified Process Plugin. However, we have to admit that our research must continue in this topic, in order to succeed in real commercial projects. It is very difficult to imagine that for a purpose of project plan verification a project manager will use a knowledge based framework directly, without appropriate user interfaces. Therefore, we have started implementation of a macro for the MS Project that will remotely access OWL API for OWL-DL reasoning purposes and it will print verification results back into MS Project Plan. Additionally, like the most similar work does, we have to include either SWRL to our approach to extend the expressiveness of description logic with the rule based expressions. All these mentioned deficiencies of our solution are the objectives of our future development.

Acknowledgments. "This work was supported by the Scientific Grant Agency of Slovakia, grant No. VG1/0508/09, and it is a partial result of the Research & Development Operational Program for the project Support of Center of Excellence for Smart Technologies, Systems and Services II, ITMS 25240120029, co-funded by ERDF."

References

1. Kruchten, P.: The Rational Unified Process: An Introduction. (3rd edition). Addison-Wesley, USA. (2003)
2. Project Management Institute: A Guide to the Project Management Body of Knowledge (PMBOK– 4th edition). Project Management Institute, USA. (2008)
3. Fujita, H., Zualkernan, I. A. (ed.): An Ontology-Driven Approach for Generating Assessments for the Scrum Software Process. In Proceedings of the seventh SoMeT_08. IOS Press, The Netherlands, 190-205. (2008)
4. Kurtev, I., Bézivin, J., Aksit, M.:Technological spaces: An initial appraisal. In Proceedings of the Confederated International Conferences, CoopIS, DOA, and ODBASE, Industrial Track, Irvine, CA, USA, (2002)
5. Steinberg, D., Budinsky, F., Paternostro, M., Merks, E.:EMF: Eclipse Modeling Framework (2nd Edition). Addison-Wesley Longman, Amsterdam, (2009)
6. Gašević, D., Djurić, D., Devedžić, V.: Model Driven Engineering and Ontology Development, 2nd ed., Springer, Berlin, (2009)
7. Happel, H.J., Seedorf, S.: Applications of ontologies in software engineering. In: International Workshop on Semantic Web Enabled Software Engineering (SWESE'06), Athens, USA. (2006)
8. Mcguinness, D. L., Harmelen, F.: OWL Web Ontology Language Overview, W3C Recommendation, (2004). [Online]. Available: http://www.w3.org/TR/owl-features/ (current November 2009)
9. Smith, M.K., Welty, Ch., McGuinness, D.L.: OWL Web Ontology Language Guide, W3C Recommendation, (2004). [Online]. Available: http://www.w3.org/TR/owl-guide/ (current November 2009)
10. Object Management Group: Software and Systems Process Engineering Meta-Model 2.0, formal/2008-04-01. Object Management Group, USA, (2008). [Online]. Available: http://www.omg.org/technology/documents/formal/spem.htm (current November 2009)
11. Krdžavac, N., Gašević, D., Devedžić, V.: Model Driven Engineering of a Tableau Algorithm for Description Logics. Computer Science and Information Systems, Vol. 6, No. 1, (2009)
12. Frankel, D.S.: Model Driven Architecture. Applying MDA to Enterprise Computing. Willey, USA. (2003)
13. Object Management Group: Meta Object Facility (MOF) 2.0 Core Specification, formal/2006-01-01. Object Management Group, USA, (2008). [Online]. Available: http://www.omg.org/spec/MOF/2.0/ (current November 2009)
14. Brickley, D., Guha, R. V., McBride, B.: RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation, (2004). [Online]. Available: http://www.w3.org/TR/rdf-schema/ (current November 2009)
15. Pan, J., Horrocks, I.: Metamodeling Architecture of Web Ontology Languages, In Proceedings of the First Semantic Web Working Symposium, Stanford, 131-149. (2001)
16. Object Management Group: UML 2.2 Superstructure Specification, formal/09-02-03. Object Management Group, USA, (2009). [Online]. Available: http://www.omg.org/technology/documents/formal/uml.htm (current November 2009)
17. Cranefield, S.: Networked Knowledge Representation and Exchange using UML and RDF. Journal of Digital Information, Volume 1 Issue 8, (2001)
18. Hart, L., Emery, P., Colomb, B., Raymond, K., Taraporewalla, S., Chang, D., Ye, Y., Kendall, E., Dutra, M.: OWL Full and UML 2.0 Compared. OMG TFC Report, (2004)

19. Polášek, I., Kelemen, J.: Ontologies in Knowledge Office Systems. In: KEOD 2009, 1st International Conference on Knowledge Engineering and Ontology Development, Funchal - Madeira, Portugal. INSTICC PRESS, 400-413. (2009)
20. Polášek, I., Chudá, D., Kristová, G.: Modelling System Dynamics in a Newer Version of UML (in Slovak). In: Proc. Systémová integrácia 2006. Žilina University, Žilina,311-317. (2006)
21. Hrnčič, D., Mernik, M., Forgáč, M., Kollár, J.: Evolution and Adaptation of Domain Specific Languages. In: Proc. of the Tenth International Conference on Informatics 2009, Technical University of Kosice, Kosice, 154-159. (2009)
22. Kollár, J., Porubän, J., Václavík, P., Bandáková, J., Forgáč, M.: Adaptive Language Approach to Software Systems Evolution. In: Proc. International Multiconference on Computer Science and Information Technology: 1st Workshop on Advances in Programming Languages (WAPL'07), Polish Information Processing Society, 1081-1091. (2007)
23. Object Management Group: Ontology Definition Meta-Model 1.0. formal/2009-05-01. Object Management Group, USA, (2009). [Online]. Available: http://www.omg.org/spec/ODM/1.0/ (current November 2009)
24. Gašević, D., Djurić, D., Devedžić, V.: MDA and Ontology Development. Springer, Berlin, Heidelberg. (2006)
25. Gašević, D., Djurić, D., Devedžić, V.: Bridging MDA and OWL Ontologies. Journal of Web Engineering, Vol. 4, no. 2, pp. 119–134. (2005)
26. Líška, M.: An Approach of Ontology Oriented SPEM Models Validation. In Proceedings of the First International Workshop on Future Trends of Model-Driven Development (FTMDD) in the context of the 11th International Conference on Enterprise Information Systems. Milan, Italy, 40-43. (2009)
27. Líška, M., Navrat, P.: An Ontology Based Approach to Software Project Enactment with a Supplier. In 14th East-European Conference on Advances in Databases and Information Systems (ADBIS2010), Lecture Notes in Computer Science 6295, Novi Sad, Serbia, Springer, pp. 378-391. (2010)
28. Wang, S., Jin, L. J. CH. Represent Software Process Engineering Metamodel in Description Logic. In Proceedings of World Academy of Science, Engineering and Technology, vol. 11. (2006)
29. Zualkernan, I. A. An Ontology-Driven Approach for Generating Assessments for the SCRUM Process. New Trends in Software Methodologies, Tools and Techniques. IOS Press. (2008)
30. Schwaber, K., Beedle, M. Agile Software Development with SCRUM. Prentice Hall. (2002)
31. Rodríguez, D., Sicilia, M., A.: Defining SPEM 2 Process Constraints with Semantic Rules Using SWRL. In Proceedings of the Third International Workshop on Ontology, Conceptualization and Epistemology for Information Systems, Software Engineering and Service Science held in conjunction with CAiSE'09 Conference. Amsterdam, The Netherlands, pp. 95-104. (2009)
32. Horrocks, I., Patel-Schneider, P., F., Boley, H., Tabet, T., Grosof, B., Dean, M.: SWRL: A Semantic Web Rule Language, Combining OWL and RuleML. W3C Member Submission, (2004). [Online]. Available: http://www.w3.org/ Submission /SWRL/ (current November 2009)
33. Object Management Group: UML 2.2 Infrastructure Specification, formal/2009-02-04. Object Management Group, USA, (2009). [Online]. Available: http://www.omg.org/spec/UML/2.2/ (current November 2009)
34. Object Management Group: MOF 2.0 / XMI Mapping Specification, v2.1.1, formal/2007-12-01. Object Management Group, USA, (2007). [Online]. Available: http://www.omg.org/spec/XMI/2.1.1/ (current November 2009)

35. Djurić, D.: MDA-based ontology infrastructure, Computer Science and Information Systems, Vol. 1, no. 1, pp. 91–116. (2006)
36. Líška, M.: Extending and Utilizing the Software and Systems Process Engineering Metamodel with Ontology. PhD Thesis, ID: FIIT-3094-4984. Slovak University of Technology in Bratislava. (2010)
37. Líška, M. Extending and Utilizing the Software and Systems Process Engineering Metamodel with Ontology. Information Sciences and Technologies, Bulletin of the ACM Slovakia, Vol. 2, No. 2, pp. 13-20 (2010)
38. Kvasnička, V., Pospíchal, J.: Mathematical logic. Slovak University of Technology in Bratislava. STU Press. (2005)
39. Navrat, P. et al.: Artificial Intelligence, 2002, Slovak University of Technology in Bratislava. STU Press. (2002)
40. Vianu, V. Rule-based languages. Annals of Mathematics and Artificial Intelligence, vol. 19, no. 1–2, pp. 215–259. (1997)
41. Baader, F., Horrocks, I., Saatler, U.: Description Logics. In Steffen Staab and Rudi Studer, editors, Handbook on Ontologies, International Handbooks on Information Systems, Springer. 3-28. (2004)

Miroslav Líška received the M.S. degree in informatics from the Technical University in Košice, Slovakia in 2002, and the PhD. degree in software and information systems from the Institute of Informatics and Software Engineering, Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava in 2010. His interests include semantic web, ontologies, software process engineering and semantic enterprise oriented architectures. He currently works as a requirements analyst and software methodologist in Datalan, in Bratislava, Slovakia.

Pavol Navrat received his Ing. (Master) cum laude in 1975, and his PhD. degree in computing machinery in 1984 both from Slovak University of Technology in Bratislava. He is currently a professor of Informatics at the Slovak University of Technology and serves as the director of the Institute of Informatics and Software Engineering. During his career, he was also with other universities abroad. His research interests include related areas from software engineering, artificial intelligence, and information systems. He published numerous research articles, several books and co-edited and co-authored several monographs. Prof. Navrat is a Fellow of the IET and a Senior Member of the IEEE and its Computer Society. He is a Senior Member of the ACM and chair of the ACM Slovakia Chapter. He is also a member of the Association for Advancement of Artificial Intelligence, Slovak Society for Computer Science and Slovak Artificial Intelligence Society. He serves on the Technical Committee 12 Artificial Intelligence of IFIP as the representative of Slovakia.

Received: November 10, 2009; Accepted: October 19, 2010.