Extending and Utilizing the Software and Systems Process Engineering Metamodel with Ontology

Miroslav Líška*
Institute of Informatics and Software Engineering
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava
Ilkovičova 3, 842 16 Bratislava, Slovakia
liska@semantickyweb.sk

Abstract
SPEM is MDA's standard used to define software and systems development processes and their components. Unfortunately, its specification is semiformal, thus it is not possible to make and to verify created language statements with formal techniques such as the consistency or satisfiability verification. In order to solve this insufficiency, we propose an approach to SPEM transformation to the Semantic Web technical space that results into the SPEM ontology creation. Additionally, we present three approach's utilizations that are: a SPEM model validation with ontology, an ontology based approach to project planning and an approach to software project enactment with a supplier.

Categories and Subject Descriptors
D.2.0 [**General**]: Standards; D.2.9 [**Management**]: Software process models; I.6.4 [**Simulation and Modeling**]: Model validation; K.6.1 [**Project and People Management**]: Life cycle

Keywords
MDA, SPEM, OWL, Semantic Web, model validation, project plan verification, software project enactment

1. Introduction
The software engineering is application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software [17]. Despite the fact that at present there exist many software developments process frameworks, the fact that relatively few projects are completely successful is an indicator of the difficulty of the task [19]. 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 [36]. Moreover, these process frameworks are used in the technical spaces [20] that have model based architecture, such as MDA or EMF [34]. These kinds of technical spaces also limit knowledge based processing, owing to their weakly defined semantics [9]. Moreover the difficulty of software development is greatly enhanced when it is inevitable to cooperate with a supplier. The general issue is to manage a lot of differences such as different tasks, software work products, guidelines, roles etc [17].

1.1 Motivation
However, at present the emerging field of Semantic Web technologies promises new stimulus for Software Engineering research [12]. 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 [21]. The today's key Semantic Web technology is OWL. It 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 [2]. Thus if we transform a definition of a software method to the Semantic Web technical space, we can use many knowledge oriented techniques to maintain them. In this work we address such opportunity. We use SPEM, the MDA standard used to define software and systems development processes and their components [26]. We transform SPEM from the MDA technical space to the Semantic Web technical space; so we can work with SPEM as with an ontology. Based on this transformation, we propose three approaches that utilize created SPEM ontology in the context of selected SWEBOK knowledge areas that are: model validation, project verification and software project enactment with a supplier [17].

2. State of the art
The thesis presents an approach to extending and utilizing the Software and Systems Process Engineering Metamodel 2.0 with OWL-DL ontology. The subject is a SPEM

*Recommended by thesis supervisor:
Prof. Pavol Návrat
Defended at Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava on September 9, 2010.

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 (2010) 8-15

Ontology development from the SPEM metamodel, thus we present the definitions of the terms metamodel, model and ontology first [23, 4, 7, 29, 10, 11, 14]. Further we present SET domain ontologies that refer to the ontologies whose main goal is to represent (at least partially) knowledge of a certain subdomain within SET matter [22, 1, 15] and research works that concern with MDA combination with the Semantic Web [5, 13, 27, 9]. Finally, we discuss three works that are closet to our work, since their also intends to use SPEM in the Semantic Web technical space.

The *first* work proposes to represent SPEM in DL [35]. The work creates mapping from MOF to DL and mapping from [24] 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 proposes SPEM process constraint definition with the semantic rules with SWRL [31]. Note that SWRL is W3C Semantic Web Rule Language that combines OWL and RuleML [16]. The work introduces that the most SPEM terms can be directly translated into OWL. Additionally it states that the translation does not aim at substituting the original model, because it serves as a complement for adding reasoning and inference support to SPEM based models. A prototype design of an engine for process constraint verification is presented. The main idea is to verify consistency between SWRL process constraints that can be obtained from the EPF Composer and SWRL process constraints from a project plan.

Finally, the *third* work also intends to use SPEM in the Semantic Web technical space. It presents a competency framework for software process understanding [36]. 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 [32] software process with EPF Composer again. Consequently paper presents generation of assessments that were generated with the IMS QTI, and XML based e-Learning standard with several examples.

3. Thesis objectives

As we have already stated in the motivation, we focus on SPEM transformation to the Semantic Web technical space and on concrete utilizations of such approach. Since we have evaluated that the above mentioned proposals have addresses to this area only partially, our objectives are:

- to present the more comprehensive analysis of the SPEM 2.0 Metamodel and the SPEM UML Profile in order to create a SPEM OWL-DL ontology
- to create the more accurate ontology architecture for SPEM 2.0 with respect of the SPEM metamodel 2.0
- to present and implement a utilization of created SPEM ontology for a SPEM model validation
- to present and implement a utilization of created SPEM ontology for a project plan verification
- to present and implement a utilization of created SPEM ontology for a software project enactment with a supplier

4. Transforming SPEM to the Semantic Web technical space

The SPEM metamodel is MOF-based and reuses UML 2 Infrastructure library [32]. Its own extended elements are structured into seven meta-model packages. Above these packages SPEM defines three compliance points (CP) which are: the SPEM Complete CP, SPEM Process with Behavior CP and Content and SPEM Method Content CP. The scope of our solution is covered with Compliance Point "SPEM Complete". The reason of this compliance point is because we need to use all SPEM elements, where the most important are defined within the Method Content package, the Process with Method package and the Method Plugin package. The Method Content metamodel package provides the concepts for SPEM users and organizations to build up a development knowledge base that is independent of any specific processes and development projects. The Method Content elements are the core elements of every method such as Roles, Tasks, and Work Product Definitions. The second necessary metamodel package that we need is the Process with Method metamodel package. Process with Methods 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. Finally, the third necessary package is the Method Plugin metamodel package. The Method Plugin allows extensibility and variability mechanisms for Method Content and Process specification. It provides more flexibility in defining different variants of method content and processes by allowing content and process fragments to be plugged-in on demand, thus creating tailored or specialized content only when it is required and which can be maintained as separate units worked on by distributed teams.

4.1 The transformation problem

Since we want to work with SPEM likewise with an ontology, we have to transform it from the MDA technical space to the Semantic Web technical space. We can use several approaches to do so [28]. Certainly we preferred fully automatic approaches, but at last we have concluded that this method is unrealizable. The first such approach is a XSL transformation from the SPEM metamodel serialized in XMI [25] directly to a SPEM ontology. But even we had generated SPEM ontology classes with their taxonomy taken from the SPEM metamodel, the fact is that this approach was not sufficient, because the generated ontology does not contain all necessary information. The biggest problem was to identify relationships used for SPEM models specification, hence for object properties necessary for a SPEM ontology either. To be more concrete, Figure 1 shows an excerpt of the SPEM metamodel that defines several Method Content elements and their relationships.

The figure depicts that the Default Responsibility Assignment element is a metaclass associated with the Work Product Definition metaclass and the Role Definition metaclass. But from this metamodel it is impossible to find out, that the Default Responsibility Assignment is

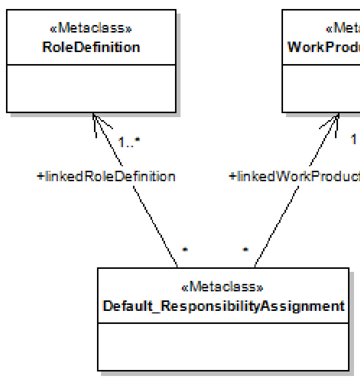
«Metaclass» RoleDefinition
«Metaclass» WorkProductDefinition
1..*
1
+linkedRoleDefinition
+linkedWorkProductDefinition
*
*
«Metaclass» Default_ResponsibilityAssignment

Figure 1: Excerpt of Method Content elements and their relationships

an association between the Role Definition and the Work Product Definition in a SPEM model. Thus it was impossible to transform this element to an object property in the transformed SPEM ontology. Moreover, it was also impossible to find out, what element should be the domain and the range of this relation. What we need is to have SPEM in such form that defines its syntax either. To do so, we can use the XMI form of the SPEM UML Profile that fulfills exactly this requirement. This time, it was possible to identify a relationship, since such element extends the Association or Dependency metaclasses. But again, it was impossible to identify the domain and the range of a relation. Moreover, even the stereotypes are structured in the taxonomical hierarchy likewise in the SPEM metamodel, some of them are bypassed. For example, the Method Content Use stereotype is omitted, thus the Task Use stereotype is generalized from the Breakdown stereotype directly. The reason is because the SPEM UML Profile does not need the full taxonomy, because in general, the purpose of a UML profile is to provide concrete language elements to create a model rather than represent semantics [3].

However, it was finally certain, that the full automatic approach of the SPEM ontology was not possible. As we have already mentioned, some elements are missing in the SPEM UML Profile. Nevertheless, they are essential from the ontology point of view. For example, a reason can be to assert a common axiom to a parent element only once, rather than multiple times to its all child elements. The motive is to have as simplest definition of the ontology as possible, because the more simple ontology the better is its maintenance. Since the missing Method Content Use stereotype is the key concept for realizing the separation of processes from method content, it is mentioned in the SPEM specifications and in our work very often. Therefore it is very essential to have it in a SPEM ontology also, because one of the main ontology purposes is to define all essential concepts from a domain that an ontology describes. Other problem we have addressed in the SPEM UML Profile is that its XMI serialization does not included the keywords of the stereotypes that are used for a stereotype presentation. Since our objective is to use the commonly accepted concrete notation for SPEM elements, we found additional reason to abandon just automatic generation of a SPEM ontology.

4.2 Solution

Based on previous facts we have decided that our approach of a SPEM ontology creation will be semiautomatic. The automatic part was used to generate ontology classes with their taxonomy. If the SPEM metaclass was neither a relationship nor package, we have transformed it to ontology as class. Such information was acquired from the SPEM UML Profile, since every stereotype is defined with a metaclass that it extends. Note that a package metaclass element is transformed to an ontology since the purpose of both concepts is to group elements into logical units [6]. So, we have generated full taxonomy of the SPEM metamodel classes that constitute the SPEM ontology. Many of the OWL classes were initially abstract metaclasses, thus not aimed for a SPEM model specification, but they are essential for purposes of ontology, as we have already mentioned. Several SPEM ontology classes are depicted in Figure 2.

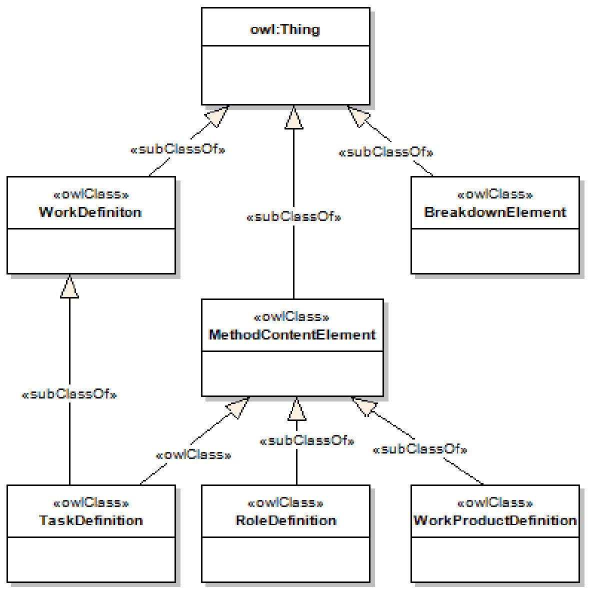
owl:Thing
«subClassOf»
«subClassOf»
«subClassOf»
«owlClass» WorkDefiniton
«owlClass» BreakdownElement
«owlClass» MethodContentElement
«subClassOf»
«owlClass»
«subClassOf»
«subClassOf»
«owlClass» TaskDefinition
«owlClass» RoleDefinition
«owlClass» WorkProductDefinition

Figure 2: Excerpt of the generated SPEM ontology classes represented with the Ontology UML Profile

The second step that had to be done was to create the relationships, i.e. object properties, between the ontology classes. Seeing that we have presented in previous that it is impossible to obtain them automatically, we had to establish them manually. Keywords of stereotypes instead of stereotypes names were used to create object properties. Additionally it was necessary to set their domains and ranges, where we used the SPEM metamodel specification. Figure 3 presents an excerpt of created object properties.

So, in the previous, we have presented the transformation of the SPEM metaclasses and their relationships to the SPEM Ontology. The created ontology provides the concepts for a SPEM method ontology and also for a SPEM process ontology. Thus we have transformed the Method Content metamodel package together with Process with Method metamodel package to the Semantic Web technical space.

The last metamodel package that had to be transformed was the Method Plugin metamodel package. The SPEM

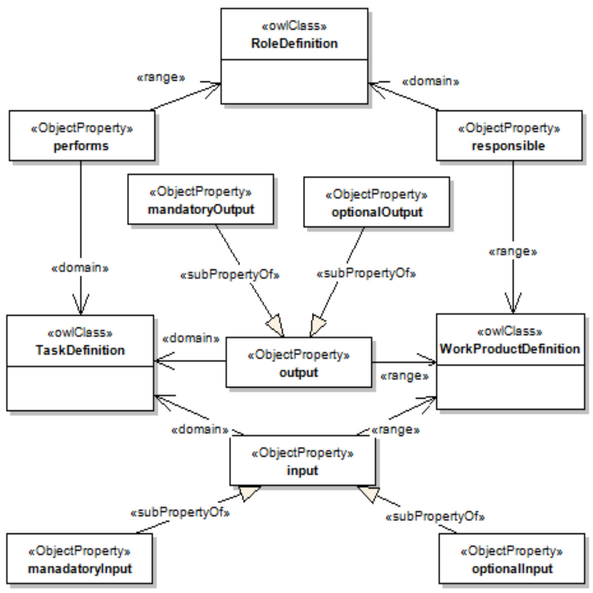
«owlClass» RoleDefinition
«range»
«domain»
«ObjectProperty» performs
«ObjectProperty» responsible
«ObjectProperty» mandatoryOutput
«ObjectProperty» optionalOutput
«domain»
«subPropertyOf»
«subPropertyOf»
«range»
«owlClass» TaskDefinition
«domain»
«ObjectProperty» output
«range»
«owlClass» WorkProductDefinition
«domain»
«range»
«ObjectProperty» input
«subPropertyOf»
«subPropertyOf»
«ObjectProperty» manadatoryInput
«ObjectProperty» optionalInput

Figure 3: Excerpt of the SPEM ontology object properties represented with the Ontology UML Profile

metamodel states, that a Method Plugin package contains a Method Content package and the Process package. By other words, a Method Plugin is a Package that represents a physical container for Content and Process Packages. It defines a granularity level for the modularization and organization of method content and processes. A Method Plugin can extend many other Method Plugins and it can be extended by many Method Plugins. Likewise as a package from MDA is transformed to an ontology, the Containment or Import relation in the MDA technical space can transformed to the Import relation in the Semantic Web technical space.

Finally, we can evaluate our approach in short. Since we have merged the SPEM metamodel to the SPEM UML Profile, we have created an extended SPEM UML Profile in matter of fact. Hence the mapping between the created Extended UML Profile and UML Ontology Profile was established; therefore the SPEM Ontology can be represented with the Ontology UML Profile as it is shown in Figure 2 and Figure 3. Since the hallmark work [9] 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 conclude that we created the solution that is conformable to this approach.

5. An Approach to Ontology Oriented SPEM Models Validation

In order to enable a SPEM method content model and a SPEM process model validation, we decided to create reusable engine that will be the base engine for every our further SPEM utilizations. Since the key models of the SPEM are the method content model and the process model, the engine should work with these models inevitably. We have already presented the SPEM Ontology creation in Section 4, thus only a transformation of a SPEM method content model to a SPEM ontology and a SPEM process model to a SPEM process ontology has to be created. Consequently we have to present an OWL DL based engine that can reason with all these three ontologies. To accomplish this task, we have a several possibilities which we discuss in following text.

The first possibility is to use a SPEM method ontology and a SPEM process ontology as the instances of the SPEM Ontology. The second possibility is to state a SPEM method ontology as the specialization of the SPEM Ontology and a SPEM process ontology as the instance of a SPEM method ontology. Likewise as a method content model is separated from its use in the process, the model level is separated from its instance in this second possibility. Thus a software process is the instance of a software method. Note that the Content Trace object property is not necessary, because is replaced by the Instantiation relation. The third possibility is to use a SPEM method content as the instance of the SPEM Ontology, and a SPEM process ontology as the instance of a SPEM method ontology.

However, all of these three possibilities are technically correct, but all of them are unsuitable for purposes of this work. The first problem is that we want to use OWL DL reasoning, because this dialect of OWL retains computational completeness, i.e. all conclusions are guaranteed to be computable and decidability, i.e. all computations will finish in finite time [8]. Therefore the third possibility that we have mentioned must be excluded in instant. But why even the two others proposed possibilities are unsuitable? The reason is because both of them do not count with real individuals of a real development project. It is evident that a requirement specifier is a real person, ”Samuel Fox” for example, than the Role Use ”Requirements Specifier Use”. By other words, an additional instance level for real individuals is needed, but it will cause the unconformity with the OWL DL reasoning again. To avoid this problem, we can state that a SPEM process model represents the instance level, rather than class level. But even this solution is insufficient. The first problem is that a SPEM process model is not intended to represent concrete individuals, thus the change of the SPEM metamodel should be made to redefine the SPEM process semantics in order to use this solution. Additionally in a real project, concrete resources (e.g. ”Samuel Fox”) are assigned much latter than an appropriate SPEM process is selected. Hence even this solution is unusable.

5.1 Solution

Based on previous facts we have realized we need a solution that addresses to all mentioned problems. Solution must to enable to validate the SPEM Ontology, a SPEM method ontology and a SPEM process ontology. It also must be extendable with the individuals from real projects (e.g. obtainable from a project plan) and finally, the solution must be OWL DL conformable. An approach that fulfills to all these requirements is that a SPEM method ontology will be the specialization of the SPEM Ontology and a SPEM process ontology will be the specialization of the SPEM Ontology either. Figure 4 depicts the presented solution from the MDA and the Semantic Web point of view. It shows the mapping between a SPEM metamodel element and a SPEM ontology class, between a SPEM method content model element and a SPEM method ontology class and finally between a SPEM method content use model element and a SPEM method content use ontology class.

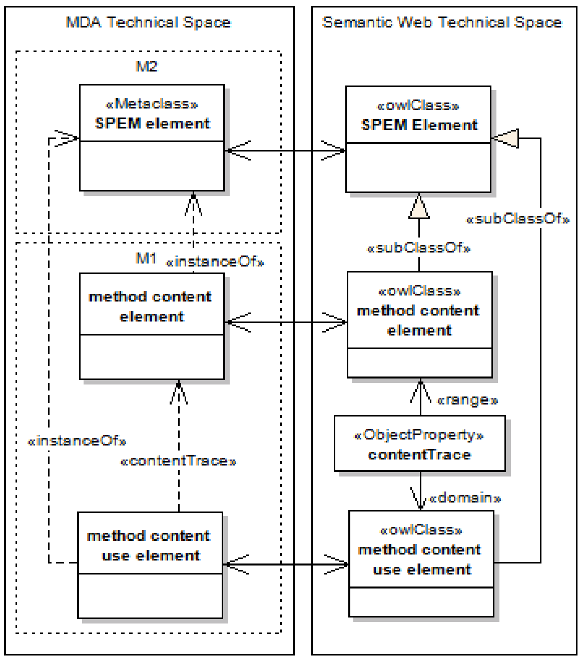

Figure 4: Mapping between elements of SPEM in MDA and SPEM in the Semantic Web in the context of the Approach to SPEM Models Validation with Ontology

Since we have resolved the problems that we have stated, we can continue to define the engine for a SPEM model validation. The engine should be usable for two scenarios, a SPEM method content model validation and a SPEM process model validation. To do so, the mandatory condition is to transform both models from the MDA technical space to the Semantic Web technical space. This requirement is not hard to accomplish because an UML model transformation even extended with arbitrary UML Profile can be transformed to a XMI serialization [18].

All we need is to create a transformation from a serialized SPEM method content model and from serialized SPEM process model to the SPEM method ontology and SPEM process ontology. Due to these requirements, we have created XSL transformations SPEMMethodContent2OWL and SPEMProcess2OWL. The former transforms a method content model serialized in XMI to a SPEM method ontology, and the latter transforms a SPEM process model to a SPEM process ontology. Since we have created the SPEM Ontology already, the OWL DL reasoning can be executed with an OWL DL reasoner. If the reasoning process contains only the SPEM Ontology and a SPEM method ontology, then the first SPEM model validation scenario is accomplished, or if a SPEM process ontology is additionally included, then the second SPEM model validation scenario can be executed. If the reasoner finds the inconsistency, its source should be removed. The engine for a SPEM model validation is depicted in Figure 5.

6. Solution

Our approach of the ontology based project plan verification fully reuses the previously presented ontology based SPEM model validation. The main extension is that the individuals from a project plan are also included for the

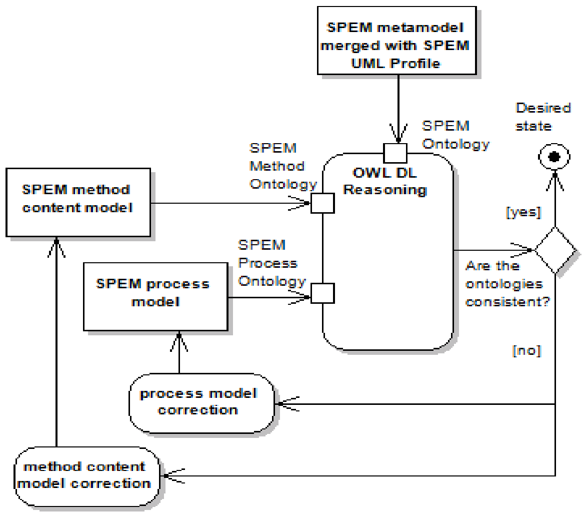

Figure 5: An approach to ontology oriented SPEM models validation

reasoning process. A project planning system usually supports creation of a project plan using the breakdown elements, such as phases, milestones, tasks, roles etc. The purpose is to decompose them to the atomic elements that can be executed by or assigned to a project member. This is the reason, why the SPEM Method Content Use metaclasses are specialized from the Breakdown metaclass. The child classes of the Method Content Use metaclass are the Work Product Use, the Role Use and the Task Use. Therefore the enactment of a project plan with a SPEM process definition has to create the instantiation relation between method content uses elements included in a project plan and method content uses elements that constitute a SPEM process ontology. Thus, our approach is fully conformable with the SPEM architecture that explicitly states a process enactment with the project planning systems through the instantiation relation. From the ontology point of view the elements obtained from a project plan are the individuals of a SPEM process ontology.

So, the previous engine for a SPEM model validation is reused and simply extended with individuals, which can be obtained from a project plan. Therefore the engine for project plan verification includes several additions than the engine for a SPEM model validation. First addition is a XSL transformation MPP2OWL that generates SPEM process instances that constitute project plan ontology from a XML format of a project plan. Second addition is the XSL transformation SPEMProcess2OWL that transforms XML format of a project plan to a SPEM process model. Note that this transformation could be bidirectional, thus it is possible to generate project plan from a SPEM process model also. 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 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 verification with the SPEM Base Plugin either. The engine that supports the ontology based project generation and verification is depicted in Figure 6.

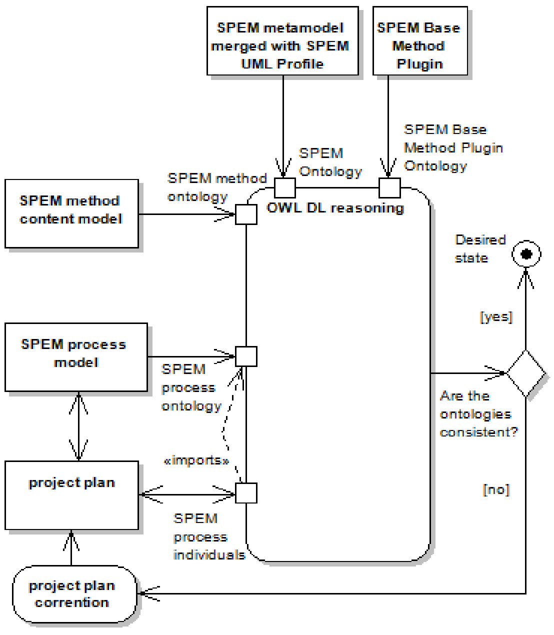

Figure 6: The engine for ontology based project planning

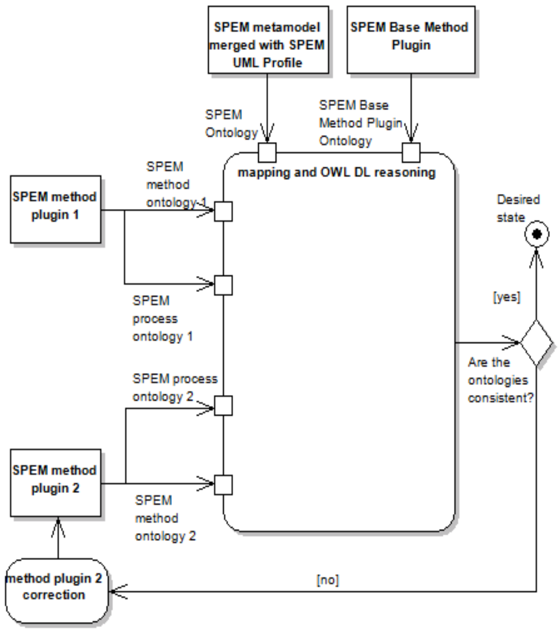

Figure 7: The engine for software project enactment with a supplier

7. An Approach to Software Project Enactment with a Supplier

Our approach to software process enactment with a supplier is based on OWL DL verification with a set of different method plugins, which represent different methods and processes of a company and its supplier. When the result of the OWL DL verification is inconsistency, it means that a project cannot be enacted with a supplier and the source of inconsistency should be removed. Therefore the necessary condition to use this method is to have company's and supplier's software process specified with SPEM models. Additionally, this approach can be used to other scenario than the software project enactment with a supplier that is the precise evaluation of the relationships between standard software processes such as RUP with Microsoft Solution Framework for example. Even this is out of scope of this work we wanted to point on it, since it is very interesting topic. The substantial part of the engine for a project enactment with a supplier is depicted in Figure 7.

The approach consists of several major steps. First it is necessary to transform the both method plugins to the ontologies. Since a Method Plugin is constituted with a Method Content and a Process we can reuse our previously defined XSL transformations SPEMMethodContent2OWL and SPEMProcess2OWL to create desired ontologies. A method plugin 1, i.e. company's method plugin is transformed to a SPEM method content ontology 1 and a SPEM process ontology 1, whereas a method plugin 2, i.e. supplier's method plugin is transformed to a SPEM method content ontology 2 and a SPEM process ontology 2. Second it is necessary to create mapping [8] between the elements of these ontologies, because some of them are usually related. To create mapping, the relationships such as sameClassAs, subClassOf, isEquivalentWith, subPropertyOf or samePropertyAs can be used. Finally, the OWL DL validation is executed to verify consistency between the both method plugins.

8. Implementation

We have used the Protégé [30], a free open source ontology editor and knowledge-base framework with the Pellet[33], an open source OWL DL reasoner that support standard reasoning capabilities such as the consistency reasoning, satisfiability verification, classification and verification. These capabilities were necessary for our approaches implementation which we have created. The consistency reasoning was used to verify whether the ontologies used in our approaches are consistent or not. The concept satisfiability was helpful to ensure that created SPEM content or process model can be really used. Seeing that the unsatisfied classes cannot have individuals, it means that a reality based on a SPEM model cannot exist. The classification was also very important because an ontology class must to adhere to all axioms stated with its superclasses. Finally either the realization was necessary, due to ensure that an individual adheres to all axioms stated with its all classifiers on the one hand, and that an individual has inferred just its desired classifiers on the other hand.

9. Conclusion

The state of art presented in this thesis was divided to the three major groups. The works that concern with ontologies in software engineering in general constitute the first group. The works that concern with the usability of ontologies in the MDA technical space constitute second group. Finally, the works that are closest to our work, which focus on the utilization of ontologies for SPEM constitute third group. Hence the evaluation of the thesis' contributions is primarily focused against this group. When we compare our work with these works, we conclude that we have created

(a) the more comprehensive analysis of the SPEM metamodel and the SPEM UML Profile in order to create a SPEM ontology than every mentioned work

(b) the more accurate ontology architecture for SPEM with respect to the actual SPEM metamodel than every mentioned works

(c) the additional ontology based approach to SPEM model validation with implementation

(d) the more accurate ontology based approach to a project plan verification with implementation than the first and third work

(e) the additional ontology based approach to software project enactment with a supplier with implementation

Explanation:

- The reason of (a) is because the second and the third work did not analyze the SPEM metamodel with concern of creating an ontology of SPEM as comprehensive as we did. The second paper depicts just an excerpt of the SPEM metamodel and concludes that "the most SPEM terms can be directly translated into OWL". Third paper does not discuss SPEM metamodel at all. First work has that intention, but unfortunately the paper is outdated and concerns with the SPEM metamodel version 1.1, that did not support the separation of method content from process; and on the contrary, the papers proposes transformation of SPEM metamodel constraints in OCL to the ontology, but the actual SPEM metamodel does not support OCL anymore
- The (b) is implied from (a)
- None of the works have presented an approach to neither a SPEM method model nor a SPEM process verification with a SPEM ontology, hence the (c) is true
- Just the second approach has concerned with the project plan verification with a SPEM ontology. Even (a, b, c, e) are true we cannot to say that our approach of project plan verification is more accurate (with respect of SPEM metamodel), rather is more comprehensive. If we focus only on the substantive part of the approach, the second work uses instantiation relation between project plan and a SPEM process either, thus it conforms to the project plan enactment with a SPEM process as it SPEM defines. Moreover, the approach proposes use of SWRL that add rules to the reasoning process. Thus we conclude that the approach proposes reasoning with advanced expressiveness.
- None of the presented works have concerned with the reasoning of more method contents or processes, thus (e) is true either

Acknowledgements. 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] M. Ángel Sicilia, J. J. Cuadrado, E. García, D. Rodríguez, and J. R. Hilera. The evaluation of ontological representations of the swebok as a revision tool. In *29th Annual International Computer Software and Application Conference)*, 2005.

[2] G. Antoniou and F. V. Harmelen. Web ontology language: Owl. In *Handbook on Ontologies in Information Systems*, pages 67–92. Springer, 2003.

[3] C. Atkinson and T. Kuhne. Profiles in a strict metamodeling framework. *Science of Computer Programming*, 44:5–22, 2002.

[4] C. Calero, F. Ruiz, and M. Piattini. *Ontologies for Software Engineering and Software Technology*. Springer-Verlag, Berlin, Heidelberg, 2006.

[5] S. Cranefield. Networked knowledge representation and exchange using uml and rdf. *Journal of Digital Information*, 1(8), 2001.

[6] D. Djurić, D. Gašević, and V. Devedžić. Ontology modeling and mda. *Journal of Object Technology*, 4(1):109–128, 2005.

[7] S. Ed. What models mean. *IEEE Software*, 20(5):26–32, 2003.

[8] J. Euzenat and P. Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2007.

[9] D. Gašević, D. Djurić, and V. Devedžić. *Model Driven Engineering and Ontology Development*. Springer, Berlin, 2. edition, 2009.

[10] T. R. Gruber. A translation approach to portable ontology specifications. *Knowl. Acquis.*, 5(2):199–220, 1993.

[11] E. N. Guarino, R. Poli, and N. Guarino. Formal ontology, conceptual analysis and knowledge representation. *International Journal of Human and Computer Studies*, 43:625–640, 1995.

[12] H.-J. Happel and S. Seedorf. Applications of ontologies in software engineering. In *International Workshop on Semantic Web Enabled Software Engineering (SWESE'06)*, Athens, USA, November 2006.

[13] L. Hart and P. Emery. OWL Full and UML 2.0 Compared. http://uk.builder.com/whitepapers/0and39026692and60093347p-39001028qand00.htm, Acessado em Outubro de 2004.

[14] J. Hendler. Agents and the semantic web. *IEEE INTELLIGENT SYSTEMS*, 16(2):30–37, 2001.

[15] J. R. Hilera, S. Sánchez-Alonso, and E. García. Issues in the development of an ontology for an emerging engineering discipline. In *First Workshop on Ontology, Conceptualizations and Epistemology for Software and Systems Engineering*, 2005.

[16] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. Swrl: A semantic web rule language combining owl and ruleml. Technical report, World Wide Web Consortium, May 2004.

[17] IEEE Computer Society. *Software Engineering Body of Knowledge (SWEBOK)*. Angela Burgess, EUA, 2004.

[18] A. G. Kleppe, J. Warmer, and W. Bast. *MDA Explained: The Model Driven Architecture: Practice and Promise*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2003.

[19] P. Kruchten. *The Rational Unified Process: An Introduction, Third Edition*. Addison-Wesley Professional, December 2003.

[20] I. Kurtev, J. Bézivin, and M. Aksit. Technological spaces: An initial appraisal. In *CoopIS, DOA'2002 Federated Conferences, Industrial track*, 2002.

[21] D. L. Mcguinness and F. van Harmelen. OWL web ontology language overview. W3C recommendation, W3C, February 2004.

[22] O. Mendes and A. Abran. Issues in the development of an ontology for an emerging engineering discipline. In *First Workshop on Ontology, Conceptualizations and Epistemology for Software and Systems Engineering*, 2005.

[23] OMG. Meta object facility 2.0, 2006.

[24] OMG. Object constraint language 2.0, 2006.

[25] OMG. Mof 2.0 / xmi mapping specification, 2007.

[26] OMG. Software process engineering metamodel 2.0, 2008.

[27] OMG. Ontology definition meta-model 1.0, 2009.

[28] OMG. Unified modeling langugae infrastructure 2.2, 2009.

[29] M. Pidd. *Tools for Thinking: Modelling in Management Science*. Wiley, 3 edition, February 2009.

[30] P. Rector, N. Drummong, and M. Horridge. Advanced reasoning with owl. In *9th International Protégé Conference*, 2006.

[31] D. Rodríguez and M. A. Sicilia. 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*, pages 95–104, 2009.

[32] K. Schwaber and M. Beedle. *Agile Software Development with Scrum*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2001.

[33] E. Sirin, B. Parsia, B. Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical owl-dl reasoner. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):51–53, June 2007.

[34] D. Steinberg, F. Budinsky, M. Paternostro, and E. Merks. *EMF: Eclipse Modeling Framework (2nd Edition) (Eclipse)*. Addison-Wesley Longman, Amsterdam, 2nd revised edition (rev). edition, January 2009.

[35] S. Wang and L. J. C. Jin. Represent software process engineering metamodel in description logic. In *Proceedings of World Academy of Science, Engineering and Technology*, 2006.

[36] I. A. Zualkernan. An ontology-driven approach for generating assessments for the scrum software process. In *SoMeT*, pages 190–205, 2008.

Selected Papers by the Author

M. Líška, P. Návrat. An ontology based approach to software project enactment with a supplier. In *14th East-European Conference on Advances in Databases and Information Systems (ADBIS2010)*, to be published in *Lecture Notes in Computer Science*, Novi Sad, Serbia, 2010, Springer.

M. Líška, P. Návrat. An Approach to Project Planning Employing Software and Systems Engineering Meta-Model Represented by an Ontology. *Computer Science and Information Systems Journal (COMSIS)*, 7(4):721-736, 2010.

M. Líška. An Approach to Ontology Oriented Employment of SPEM. In Mária Bieliková *Proceedings of IIT.SRC 2010: Student Research Conference in Informatics and Information Technologies*, pages 308–315, Bratislava, Slovakia, 2010, STU.

M. Líška. An Approach of Ontology Oriented SPEM Models Validation. In Luís Pires and Slimane Hammoudi *Proceedings of the First International Workshop on Future Trends of Model-Driven Development (FTMDD2009).* In conjuction with *11th International Conference on Enterprise Information Systems (ICEIS2010)*, pages 40–43, Milan, Italy, 2009, INSTICC Press.

M. Líška. Developing an Ontology for SPEM Method Content Models Validation. In Mária Bieliková *Proceedings of IIT.SRC 2009: Student Research Conference in Informatics and Information Technologies*, pages 179–185, Bratislava, Slovakia, 2009, STU.

M. Líška. Developing an Ontology for SPEM Method Content Models Validation. In *Proceedings of IWCIT2008-The Seventh International PhD Students' Workshop Control and Information Technology*, pages 164–169, Gliwice, Poland, 2008.

M. Líška. Developing an Ontology for SPEM Method Content Models Validation. In Peter Vojtáš *Proceedings of ITAT 2008*, pages 67–72, Hrebienok, Slovakia, 2008.

M. Líška. Constructing expert system for Model Driven Development. In Mária Bieliková *Proceedings of IIT.SRC 2007: Student Research Conference in Informatics and Information Technologies*, pages 49–50, Bratislava, Slovakia, 2007, STU.

M. Líška. Model driven development enhancement with analysis agent system. In *AIESA 2007*, Bratislava, Slovakia, 2007, FHI EUBA.

M. Líška. Constructing expert system for Model Driven Development. In Michal Laclavík, Ivana Budinská, Ladislav Hluchý *Proceedings of WIKT 2006: 1st Workshop on Intelligent and Knowledge oriented Technologies*, pages 135–138, Bratislava, Slovakia, 2006, Slovak Academy of Science.

M. Líška. Constructing multi-theories expert system for UML models validation. In Ivan Plander *Proceedings of Informatics' 2007: Proceedings of the Ninth International Conference on Informatics*, pages 163–169, Bratislava, Slovakia, 2007, Slovak Society for Applied Cybernetics and Informatics.

M. Líška. Formal Unified Process. In Mária Bieliková *Proceedings of IIT.SRC 2006: Student Research Conference in Informatics and Information Technologies*, pages 220–226, Bratislava, Slovakia, 2006, STU.

M. Líška. Extensional and intensional semantics of PUML objects. In Mária Bieliková *Proceedings of IIT.SRC 2005: Student Research Conference in Informatics and Information Technologies*, pages 167–174, Bratislava, Slovakia, 2005, STU.