An Ontology Driven Approach to Software Process Engineering

Miroslav Liška¹, Pavol Návrat²

¹ DATALAN, a. s., Galvaniho 17A, 821 04 Bratislava, Slovakia
² Faculty of Informatics and Information Technologies, Slovak University of Technology, Ilkovičova 3, Bratislava, Slovakia
liska@semantickyweb.sk, navrat@fiit.stuba.sk

Abstract. Model Driven Architecture (MDA) provides a set of metamodel standards as MOF, UML, BPMN, SPEM and others. A metamodel makes statements about what can be expressed in the valid models of a certain modeling language. Unfortunately, the mentioned metamodels have semiformal architecture, thus it is not possible to make and to verify created language statements with formal techniques such as the consistency or satisfiability verification. However, recently the combination of MDA and the Semantic Web becomes the leading subject in this direction. Nevertheless, the works mostly focus only on the UML-OWL relationships, since the possibilities of these languages are very similar. In this paper, we aimed to present our published work in this domain that is: a method of SPEM transformation into the Semantic Web technical space; a method that allows generation and verification of a project plan; and method of ontology oriented software methods integration with the SPEM Ontology.

Keywords: SPEM, OWL, Software and Systems Process Engineering Metamodel, Web Ontology Language, 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 [1]. Despite the fact that at present there exist many software developments process frameworks and the fact that relatively few projects are completely successful is an indicator of the difficulty of the task [2]. 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 EMF [5]. These kinds of technical spaces also limit knowledge based processing, owing to their weakly defined semantics [6]. 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 [1].

2 An Approach

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 [7]. 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. However, this idea is not new; it can be found in many other research works such as [8, 9, 10].

For the purpose of our method we had chosen SPEM, the MDA standard used to define software and systems development processes and their components [11]. The SPEM’s key concepts are the Method Content elements and the Method Content Use elements (SPEM Process). The former elements are the core elements of every software method definition such as Roles, Tasks, and Work product Definitions. The latter elements are abstract generalizations for special Breakdown Elements that reference one concrete Method Content Element and therefore they are the key concepts for realizing the separation of processes from method content.

2.1 Moving SPEM into the Semantic Web

In order to enable use of SPEM in the Semantic Web technical space, we make use of the fact that OWL, ODM and SPEM are serialized in XML format. The mapping between OWL and ODM is expressed in ODM that contains OWL Meta-Model [12]. The OWL Metamodel is a MOF2 compliant metamodel that allows a user to specify ontologies using the terminology and underlying model theoretic semantics of OWL [13]. Thus a mapping between SPEM and OWL had to be created. Since we wanted to use SPEM UML Profile for SPEM modeling rather than Ontology UML Profile, we have created mapping between merged SPEM UML Profile to SPEM Metamodel and OWL through ODM. The result of the transformation is the SPEM Ontology [14].

2.2 Project Plan Generation and Verification

Seeing that Method Content Use elements can be mapped to a project plan, we can apply instantiation relation between Method Content Use elements and a project plan elements and verify their consistency. We had stated that the Method Content Use elements are classifiers specialized from the Method Content elements, whereas the project plan elements are the individuals of a Method Content Use elements. In order to execute OWL-DL reasoning for a project plan verification, we have created three XSL transformations SPEMMethodContent2OWL, SPEMProcess2OWL and MPP2OWL [15]. The first and second transforms a Method Content and Process model to an ontology that constitutes a knowledge base for project planning of a subjected software engineering area (e.g. Software requirements). The third
transformation transforms a project plan to the individuals of a SPEM process ontology. When the result of the OWL DL verification is the inconsistency its source should be removed. Note that when the last mentioned transformation is executed with reversed order, i.e. OWL2MPP, a project plan can be generated.

2.3 Software process enactment with a supplier

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 and their responsibilities, etc. Therefore an extensive analysis of these variations should be done before a software process will start. According to this requirement we propose an approach that supports integration of software methods [16]. The approach is based on consistency verification between a set of different Method Plugins. The Method Plugin allows extensibility and variability mechanisms for Method Content and Process specification. 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 [17] between the elements of these ontologies, because some of them are usually related.

3 Conclusion

When we compare our approach with a similar work we conclude that we created not only wider method specification, but we have also presented its implementation. However, we admit that our implementation is only partially automated, because our transformation SPEM-Method-Content2OWL does not transform OCL Constraints into SWRL rules, thus, they must be added manually. Moreover, it is fair to say that our research in this topic must continue, 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. The other important deficiency of our approach that limits its use in commercial project is that we do not support Method Content derivation from standardized software processes such as RUP. All these mentioned deficiencies of our solution are the objects of our future development.

Acknowledgement: This work was partially supported by the Science and Technology Assistance Agency under the contract No. VG1/508/09.
References

12. OMG. Ontology definition meta-model 1.0, 2009.