Defining SPEM 2 Process Constraints with Semantic Rules Using SWRL

Daniel Rodríguez[1], Miguel-Ángel Sicilia[1]

[1] *Information Engineering* Research Unit
Computer Science Dept., University of Alcalá
Ctra. Barcelona km. 33.6 – 28871 Alcalá de Henares (Madrid), Spain
{daniel.rodriguez, msicilia}@uah.es

Abstract. The Software & Systems Process Engineering meta-model (SPEM 2) allows the modelling of software process using OMG's MOF meta-model and UML profiles, thus being compliant with UML tools. Process definition encompasses both the static structure of activities, roles, tasks and work products and the constraints on those elements. The latter require support for constraint enforcement that is not directly available in SPEM 2. Such constraint-checking behaviour could be used to detect mismatches between process definitions and the actual processes being carried out in the course of a project. This paper approaches the modelling of such constraints using the SWRL (Semantic Web Rule Language) W3C recommendation, using an underlying OWL representation of SPEM 2 models that can be directly derived from their XMI serialization.

Keywords: SPEM, Ontologies, OWL, Semantic Rules, SRWL.

1 Introduction

Current approaches to model and evaluate processes are highly demanded in industry in order to improve the quality of their processes and in turn, the products they generate. In addition to well established frameworks such as CMMI (Capability Maturity Model Integration) 6 or SPICE (Software Process Improvement and Capability dEtermination) also know an ISO/IEC 15504 standard 12, the OMG (Object Management Group)[1] has developed a model to represent processes called SPEM (Software & Systems Process Engineering Meta-model) 14 version 2.

The SPEM is both a meta-model and profile specification. SPEM allow us to formalise the development process modelling and structuring of processes (roles, products, deliverables, guides, life-cycle, phases, milestones, etc.) which in turn allow us to manage and monitor projects. Currently SPEM is supported by the Eclipse tool[2] as part of the Eclipse Process Framework Project (EPF).

[1] `http://www.omg.org/`
[2] `http://www.eclipse.org/epf`

An ontology is an explicit representation of domain concepts and provides the basic structure of the system. More formally, an ontology "*defines the vocabulary of a problem domain and a set of constraints on how terms can be combined to model the domain*" 7. Common uses of ontologies include communication between people and organizations and interoperability between systems, i.e., translation of modelling methods, paradigms, languages and software tools 17.

In software engineering, ontologies can also be used by applications require a higher level of formality of definition. For example, cataloguing resources or mapping of vocabularies from different information sources requiring precise definitions, or at least significant characterizations that help in deciding which terms to use in practical situations. Ontologies allow us to add semantics to data so that different software components can share information in a homogeneous way. Furthermore, logic can be used in conjunction with such formal representations for reasoning about the information and facts represented as ontologies. Therefore, desirable qualities provided by ontologies include reusability providing formal representations, searchability providing meta-data as indexes to information, and reliability performing consistency checking.

Ontologies have been a recourse in the semantic Web to integrate and deal with information for quite some time. In the semantic Web, one of the current standards for representing ontologies is the OWL (Ontology Web Language) and the standard for rules is WSRL (Semantic Web Rule Language).

There are several works on representing software engineering concepts with ontologies. For example, REFSENO 16 (Representation Formalisms for Software Engineering Ontologies) that has been applied for modelling experience factories 3 using the Goal-Question-Metric paradigm 4. Althoff *et al* 2 describe an architecture oriented to reuse the experience in software engineering that use ontologies as the underlying formalism.

Other researchers are developing ontologies based on current standards such as IEEE Standard Glossary of Software Engineering Terminology [16] or the Guide to the Software Engineering Body of Knowledge (SWEBOK), which is now an ISO standard. Such standards provide an agreement on the content of what compose the software engineering discipline opening new possibilities to ontology engineering in the field of software engineering, since they represent a shared consensus on the contents of the discipline. For example, Abran *et al* 1 report on the developed of a software engineering ontology based on the SWEBOK and the process for its creation. Based on SPEM and other measurement standards García *et al* 8 developed an ontology to represent software engineering measurement concepts. Calero *et al* 5 have edited some works related to ontologies and software engineering.

In this paper, we show how the representation of projects using SPEM can be translated to OWL ontologies which in turn can be used for checking constrains with defined in the processes or in the execution of a project using SWRL rules.

The rest of this paper is structured as follows. Section 2 covers the background. Section 3 summarizes the processes of creating ontologies from SPEM, followed by how constrains can be modelled and executed in Section 4. Finally, Section 5 concludes the paper and outlines future work.

2 Background

The SPEM (Software Process Engineering Meta-model) is the OMG proposal to represent processes. As stated previously SPEM is based on the other OMG MOF and UML standards. It is composed of the following meta-model packages:

- *Core*: It contains common classes and abstractions used to build upon.
- *Process structure*: It represents the static concepts of processes with nesting activities and predecessor dependencies.
- *Process behaviour*: It extends the *Process Structure* packages with behavioural models such as activity diagrams for process behaviour or work products with state machines to represent its lifecycle.
- *Managed Content*: This package introduces concepts for managing textual description (natural language) and documentation capabilities for processes.
- *Method Content*: It provides the concepts for defining lifecycle and process-independent reusable *method content* elements that provide a base of documented knowledge of software development methods, techniques, and best practices.
- *Process With-Methods*: It defines new and redefines existing structures for integrating *Process Structure* concepts with instances of *Method Content* concepts (Tasks, Roles, and Work Products) into the context of a lifecycle model comprising, for example, of phases and milestones.
- *Method Plug-in*: It allows us to introduce the concept of variability in processes, in what is called method configuration, where the user can add or remove elements without modifying the original.

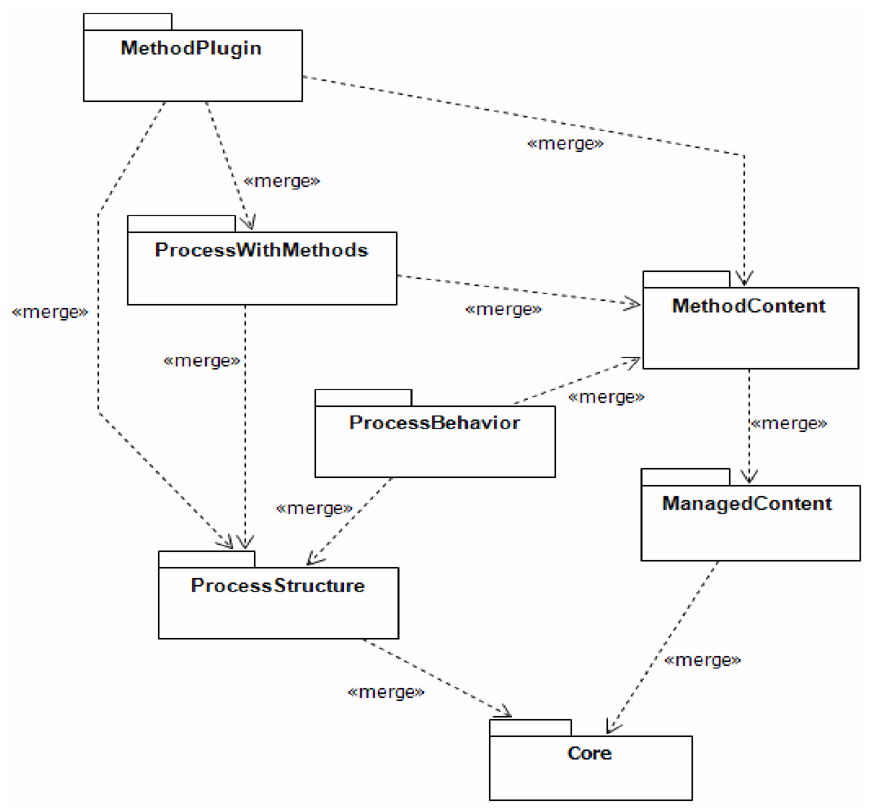

Fig 1. SPEM 2 Meta-packages 14

Ontologies can be represented in different formalisms currently one of the most popular and the one used in this work, is the Ontology Web Language (OWL) standard 15. The OWL is a family of knowledge representation languages prepared for the Web that has reached status of W3C (World Wide Web Consortium) recommendation. From technical point of view OWL extends the RDF (Resource Description Framework) and RDF-S (RDF Schema) which was originally developed for the so-called Semantic Web. It allows us to integrate a variety of applications using XML as interchange syntax. There are three OWL flavours, Lite, DL (Description Logics) and Full, being the OWL Lite or DL the ones used for the rules. The Semantic Web Rule Language (SWRL) 13 is in turn an extension to OWL and RuleML for providing logic based rules.

Protégé[3] 9 is used as a tool for ontologies and it is able to reason with the rules. It can work with several ontological languages including OWL and SWRL.

[3] `http://protege.stanford.edu/`

3 Representing SPEM 2 Models with OWL Ontologies

The creation of ontologies is not straight forward; there are no standard modelling methodologies but a mix of guidelines that are combined with techniques from the database modelling and object oriented modelling to iteratively achieve the desired representation 10. This section specifies how SPEM 2 models can be translated to a representation in OWL that retains all the modeling semantics specified in SPEM. Typically, a SPEM model would be translated into one or several OWL files, and other OWL files with the basic mappings would be imported by these. As stated previously the idea is that the resulting OWL files can be enriched with rules or axioms thus benefiting of the capabilities of formal semantics. These capabilities can be used for implementing process realization behaviors in diverse engineering tools.

The translation does not aim at substituting the original model, but to serve as a complement for adding reasoning and inference support to SPEM based models. In consequence, simplicity has been preferred in contrast to mirroring every SPEM element. For example, many terms in the stands are linked to other through inheritance to provide the necessary semantic meaning which can be directly defined when creating the ontology. Said this, it should be noted that other translation approaches could be devised in the future with better alignment to OMG meta-modeling specifications.

For the translation, we generated the OWL files using the Protégé tool following the nomenclature used in the standard whenever possible. For example, *WorkDefinition* from the core ontology (in core meta-package of the) can be further refined in the process structure package as *WorkBreakdownElement* which in turn contains the *Activity* and *Milestone* elements.

At this level, there we can also deal with instances. For example, *RoleUse* can have instances such as *analyst*, *programmer*, *designer*, etc. In the same way, the *WorkProductUse* could be generic documents such as User Require Documents (URD) or design documents. All these activities and work products (tangible or intangible) are related to generic processes, and not to any concrete execution of a process.

A large number of terms are generic to many software engineering processes and methodologies which are defined in numerous standards and guides. As stated previously, there are current research works aiming to create ontologies from standards and/or the guide to the SWEBOK. Such works can be used in conjunction with SPEM as starting point of the ontological process.

4 Modelling process constraints with SWRL

As stated previously, the main motivation of this work is to actually check and verify constrains that can be defined as part of a SPEM models and other information introduced with associated tools such as project management tools. On the one hand, the process and structure of a process can be obtained from SPEM, using tools such as EPF. Also, the actual execution of process can be further defined and monitored with project management tools such as MS Project which extends the SPEM models with information such as start and end dates, duration of activities, etc. On the other hand, the ontological tools such as Protégé and JESS can execute rules to verify constrains and inconsistencies in a project. Fig. 2 shows this approach, where ontologies represented using OCL and rules with SWRL are combined to better manage the project.

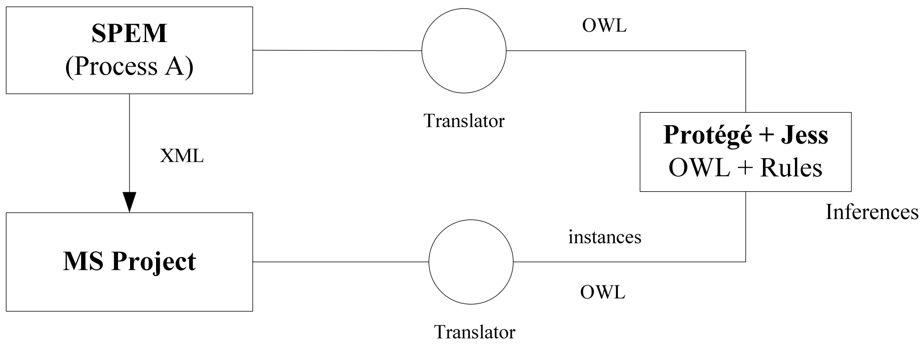

Fig. 2. Extension of SPEM with Semantic Knowledge and Rules

It is worth noting that although many of the constrains in UML can be defined using the Object Constrain Language (OCL), it lacks, however, the expressiveness and tool support that the semantic Web provides. Following the example described in the SPEM specification 14 as a precondition: "*Input Document X has been reviewed and signed by customer AND the work defined by Work Definition 'Management Review' is complete*". Such precondition is expressed in natural language and associated to the the `WorkDefinition` class compositional association. Even if expressed in OCL, we are not aware of any environment that allows their execution.

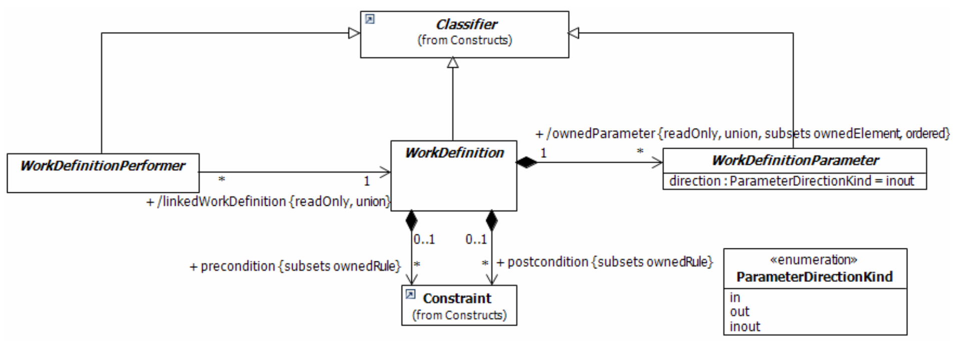
Classifier
(from Constructs)
WorkDefinitionPerformer
WorkDefinition
WorkDefinitionParameter
direction : ParameterDirectionKind = inout
+ /ownedParameter {readOnly, union, subsets ownedElement, ordered}
+ /linkedWorkDefinition {readOnly, union}
+ precondition {subsets ownedRule}
+ postcondition {subsets ownedRule}
Constraint
(from Constructs)
«enumeration»
ParameterDirectionKind
in
out
inout

Fig. 3. Defining Constrains in SPEM 14

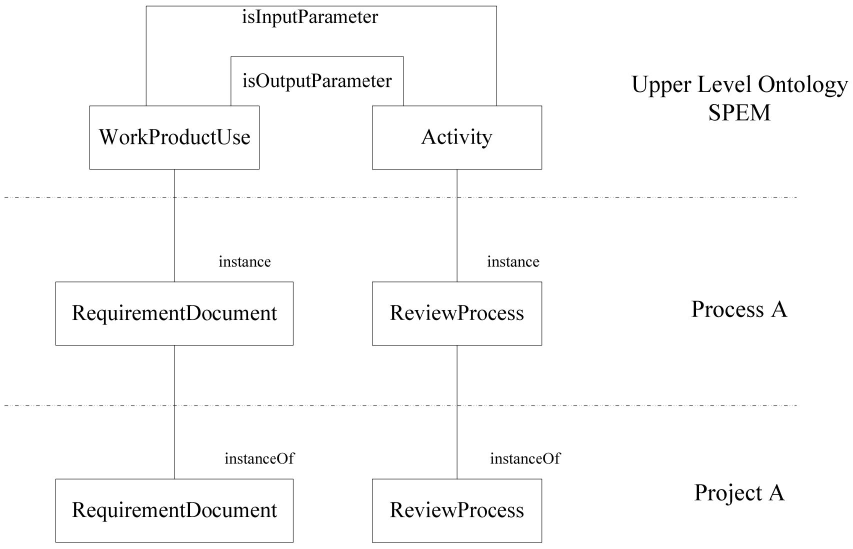

Fig. 4. Different Semantic Levels when Creating SPEM Ontologies

After defining the SPEM ontology, we now provide an overview of how SPEM and project constrains can be expressed using the SWRL as a rule language capable of checking and verifying constrains. It is possible to run rules at the same level or between different levels in the ontological hierarchy shown in Fig. 4. For example, when an activity has a work product as input (`isInputParameter`) and output (`isOutputParameter`), a rule could automatically include another property that such work product is both input and output parameter (`isInOutParameter`). In the UML SPEM profile this is defined as an enumeration (`ParamterDirec-tionKind`).

```
WorkProductUse(?x) ^
isInputParameter(?x, ?a) ^
Activity(?a)
```

```
→ isInOutParameter(?x, ?a)
```

Another example of rule could be when a work product goes through the process of review; in such a case, there could exist a property (reviewed) which automatically is updated to true.

```
WorkProductUse(?x) ^ i
sInputParameter(?x, Review)
→ reviewed(?x, true)
```

Finally, we could specify concrete examples when running a project such as if a *User Requirements Document* (URD) has been reviewed and signed, then we could assign to the Boolean property `modifiable` the value false. Note that such a rule could be part of some guideline when specifying the process.

```
WorkProductUse(URD) ^
reviewed(URD) ^
signed(URD)
→ modifiable(URD, false)
```

5 Conclusions and Future Work

In this paper we have presented how ontologies of software processes and projects can be defined using the Software & Systems Process Engineering Meta-model (SPEM) ver. 2. Also, those ontologies can be extended with rules representing project constrains with the objective of automating their execution.

The ontological representation of process definition encompasses both the static structure of activities, roles, tasks and work products and all the possible constraints in the definition and execution of processes. Furthermore, ontologies as formal representation and associated rules allow us to execute and enforce such constrains once the project is being carried out in order to detect possible mismatches. Such capability that is not directly available using SPEM 2. We presented our approach using OWL as ontological language and the ontological modelling of such constraints using the SWRL (Semantic Web Rule Language).

Future work includes the further development of the ontologies started here as a proof of concept and additional development of rules for other methodologies already specified using the SPEM standard. For example, SCRUM has numerous rules informally defined in relation to team size which are typically between 5 and 9, development times (a sprint should be between 2 to 4 weeks), meetings and their duration, etc. The build-ins rules already provided with SWRL can be further explored for defining such constrains.

Acknowledgments. This research was supported by the Autonomous Community of Madrid Research Agency (CCG07-UAH/TIC-1588) and the University of Alcalá. Authors thank the anonymous reviewers for their comments.

References

1. Abran, A., Cuadrado, J.J., Garcia-Barriocanal, E., Mendes, O., Sanchez-Alonso, S., Sicilia, M. A.: Engineering the ontology for the software engineering body of knowledge: issues and techniques. Ontologies for Software Engineering, Springer Verlag, New York, NY, (2006)
2. Althoff, K.D., Birk, A., Hartkopf, S., Muller, W., Nick, M., Surmann, D., Tautz, C.: Systematic Population, Utilization, and Maintenance of a Repository for Comprehensive Reuse. In Learning Software Organizations - Methodology and Applications, Springer Verlag, LNCS 1756, pp. 25-50. (2000)
3. Basili, V. R., G. Caldiera and D. Rombach, "Experience Factory", *Encyclopedia of Software Engineering*, vol. 1, J. Marciniak, Ed.: John Wiley & Sons, pp. 469-476, (1994).
4. Basili, V., Caldiera, G. and Rombach, H.D.: Goal Question Metric Approach, *Encyclopedia of Software Engineering*, John Wiley & Sons, Inc., pp. 528-532, (1994)
5. Calero, C., Ruiz, F.; Piattini, M. (Eds.): Ontologies for Software Engineering and Software Technology, Springer, (2006)
6. CMMI: Capability Maturity Model Integration, Version 1.2. Web site: `http://www.sei.cmu.edu/cmmi/`
7. Devedžic, V.: Understanding Ontological Engineering, *Communications. of the ACM*, **45** (4), pp. 136-144 (2002).
8. García, F., Piattini, M., Ruiz, F., Canfora, G., and Visaggio, C. A.: FMESP: Framework for the Modelling and Evaluation of Software Processes. In *Proceedings of the 2004 Workshop on Quantitative Techniques For Software Agile Process* (2004).
9. Gennari, J. H., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubézy, M., Eriksson, H., Noy, N. F., Tu, S. W.: The evolution of Protege: an Environment for Knowledge-based Systems Development. *International Journal Human-Computer Studies*, **58**(1), pp. 89–123, (2003)
10. Gruber, T. "Toward Principles for the Design of Ontologies Used for Knowledge Sharing, *International Journal of Human-Computer Studies*, **43**(5-6) pp. 907-928, (1995)
11. IEEE: IEEE Standard Glossary of Software Engineering Terminology, IEEE Std (1992)
12. ISO: ISO/IEC 15504 Information technology Process assessment. Parts 1 to 8. Web site: `http://www.iso.org/` (2009)
13. SWRL: A Semantic Web Rule Language, W3C World Wide Web Consortium, Web Site: `http://www.w3.org/Submission/SWRL/` (2009)
14. OMG: Software Process Engineering Meta-model (SPEM) Specification. Version 2. Technical Report ptc/2008-04-01, Object Management Group (2008)

15. OWL: OWL Web Ontology Language Overview, W3C World Wide Web Consortium, Web site: `http://www.w3.org/TR/owl-features/` (2009)

16. Tautz C., von Wangenheim, C.G.: REFSENO: A Representation Formalism for Software Engineering Ontologies, Fraunhofer Institute IESE IESE-015.98/E, (1998)

17. Uschold, M., Grüninger, M.: Ontologies: Principles, Methods, and Applications, *Knowledge Engineering Review*, **11**(2), pp. 93-113, (1996)