Process Definition and Project Tracking in Model Driven Engineering

Ivan Porres[1] and María C. Valiente[2]

[1] Department of Computer Science, Åbo Akademi University
Lemminkäisenkatu 14, FIN-20520 Turku, Finland
ivan.porres@abo.fi
[2] Department of Computer Science, Carlos III University of Madrid,
Avda. Universidad 30, 28911 Leganés (Madrid), Spain
mcvalien@inf.uc3m.es

Abstract. This paper presents a software process definition language that is targeted towards the development of software and systems using Model Driven Engineering methods. The dynamics of a process model are based on Petri Nets. This allows us to use a process definition model to plan and track the execution of actual projects. This new language can be integrated with existing approaches for software process modeling such as Software Process Engineering Metamodel.

1 Introduction

Model Driven Engineering (MDE) is a software and system construction approach based on high-level abstract modeling. All the relevant information in a project is stored in models based on well-defined languages and development is then carried out as a sequence of model transformations. The MDE term was first proposed by Kent in [8] but it is derived from the OMG's Model Driven Architecture (MDA) initiative [13].

MDE is the result of recent developments on computer languages, awareness of the need of software and system development methodologies and the constant need to tackle larger and more complex system development projects. The two key elements in MDE are modeling languages and modeling tools to create and transform models. A modeling language is defined using standardized metamodeling languages such as MOF [11] or the UML 2.0 Infrastructure [16].

Automated model transformation is sometimes seen as the next silver bullet in software and system engineering. However, we consider that this is not the main advantage of MDE, since any non trivial development process will contain many development steps that cannot be automated. As a consequence, any MDE development step involves model transformations, but these transformations should be often performed by a skilled designer and not by a CASE tool.

On the other hand, we consider that the fact that we represent in MDE our software and system artifacts as models and that these models are based on a common metamodeling languages brings us many advantages to system development that have not been fully exploited. A uniform metamodeling approach enables us to build tools to manage complex relationships between the artifacts that form a complex project such as refinement [1] or retrenchment [2].

J. Münch and M. Vierimaa (Eds.): PROFES 2006, LNCS 4034, pp. 127 – 141, 2006.

In this paper, we study how recent advances in the area of software modeling languages and model transformations can be used in the context of software process modeling.

Different process modeling techniques have been used during decades. Software process modeling studies how to capture and describe a software process. A software process model is often a simplified representation of an actual process. However, this is one of its main advantages: a software process model should be easy to understand and follow by all the developers involved in a given project.

Approaches already exist to process definition that are integrated with modeling languages such as *Software Process Engineering Metamodel* (SPEM) [15] or the Rational Unified Process (RUP) [9]. However, these approaches do not address the definition of actual project tasks or development steps and tracking or execution of a project based on a process. This is a consequence of the fact that SPEM and RUP are generic process definition approaches that can be used in any development approach.

In this article, we propose a simple process definition modeling language that combines concepts from process definition languages such as SPEM with concepts from Model Driven Engineering such as model mappings and transformations. We define the execution semantics of the process models in terms of Petri nets. The result is a language that can be used to define a model-based development process in detail and, thanks to its behavioral semantics, to track the execution of projects based on a process.

We should note that our proposal is not a replacement to existing process definition approaches such as SPEM but a complement to them. Our approach focuses on the definition of process steps, but ignores other important aspects such as resource and role definitions since we consider that they are well supported in existing approaches such as SPEM.

We proceed as follows: In Section 2 we review the main concepts of Model Driven Engineering and how can be applied to software process modeling. We describe our new language for process modeling in Section 3 and we describe its dynamics in Section 4 using a practical example. Finally, we conclude in Section 5 with some final remarks and related work.

2 Definition of Software Development Processes in MDE

The execution of any complex software development process will include many different development steps that will produce internal and deliverable artifacts. A software development process contains the definition of each one of these steps and artifacts. Many organizations with complex projects have realized the need of better techniques and tools to support the management of software development process [4] and continue to demand more progress in the development process with acceptable results.

In short, software development process produces one tangible “thing” that is the software system. If we study the characteristics of software process to produce the “thing”, we can conclude that there is a close relationship between software development process and business process. A software development process may be considered, and therefore managed as a business process. Like in business processes [21], in the software process there are cases (i.e., projects) that involve a process (i.e., software process that defines the life cycle of a project), conditions, tasks, work items, activities and resources that perform specific tasks in the process.