Generic Framework for Defining Domain-Specific Models

Arnor Solberg
SINTEF ICT, Norway

Jon Oldevik
SINTEF ICT, Norway

Audun Jensvoll
EDB Telesciences, Norway

INTRODUCTION

As a result of the widespread popularity of the Unified Modeling Language (UML) (OMG, 2003-1), many companies have invested in introducing a UML-based methodology. There are many general purpose UML-based methodologies on the market today; among the most popular are UP (Jacobson, Booch & Rumbaugh, 1999), RUP (Kruchten, 2000), Catalysis (D'Souza & Wills, 1998), Select Perspective (Allen & Frost, 1998), and KOBRA (Atkinson et al., 2001). Typically, these general purpose software system development methodologies do not immediately fulfill a company's need. Aiming to provide methodologies that may be applied in many domains and for many purposes, these general purpose methodologies typically become extensive and are perceived as overwhelming. At the same time they typically lack support for the more exclusive needs that the companies and domains encounter. Thereby, introducing a general purpose methodology in an organization commonly implies two particular challenges that at first sight seems to be contradictory. On one hand there is a problem that the general purpose methodology provides/prescribes far too much and encounters too many situations. On the other hand the general purpose methodology does not support specific modeling concepts, mechanisms, and techniques wanted by the particular company or development group. Thus, in that respect the general purpose methodology actually covers too little. This state of affairs is why lots of consultants, researchers, and others are in the business of helping companies to introduce these methodologies, as well as customizing general purpose methodologies to be appropriate for the actual company and purpose. The customization is typically tuned based on different criteria such as domain, kind of customers, quality demands, size of the company, and size of the software development teams. A common way of customizing a general purpose methodology is by removing, adding, and/or merging prescribed tasks, phases, roles, and models/artifacts of the methodology. However, even if introduction of a general purpose methodology almost always requires a customization effort, there does not seem to be any standard and formalized way of doing it.

BACKGROUND

Our research group has for quite some time worked with customizing methodologies to satisfy specific needs. Our customization has been accomplished by taking a set of different general purpose methodologies (e.g., RUP, UP, OOram (Reenskaug & Wold, 1996)), methodology expertise, and experience as input into a collaborative process together with architects and super-users. By massaging this input through an iterative and incremental process in which we have analyzed the company's need and existing methodology (or practice) in use within the company, company culture, particularities of the domain, customers, market, and so forth. The output has been a tailored methodology.

Some results of this work have been the COMBINE methodology (COMBINE, 2003, 2000), the DAIM methodology (Hallsteinsen, Solberg, Fægri, Oldevik & Syrstad, 2003; DAIM, 2001), TeMOD (Solberg & Oldevik, 2001) and the Configurable LIght-Weight Method (CLIMB) (Solberg, Oldevik & Jensvoll, 2002a). What we have discovered during our work was that even if we gained substantial benefits from tailoring general purpose methodologies to the needs of the company, the company itself is quite diverse. Thereby, a need was expressed of even more tailoring to fit the purpose of different domains and product families within the company. For instance when developing TeMOD for Telenor[1] and later CLIMB for EDB Telesciences[2], a main request was to deliver a methodology that was tailored to capture and utilize existing domain knowledge. However, one of the goals of making

TeMOD and CLIMB was to provide a common methodology to be used throughout the company, in order to achieve a common way of developing and specifying systems. Thus, we were not supposed to end up with a set of proprietary special purpose methodologies, one for each domain and system development group. Our challenge became to keep TeMOD and CLIMB as the common methodologies for the respective company, enforcing standardized processes and specifications, and at the same time get the methodology to support specific needs of different domains and utilize the existing domain knowledge possessed within the company.

The most popular general purpose UML-based software engineering methodologies have both diversities and commonalties. One frequent commonality is that they are model driven. A model-driven methodology signifies that the methodology prescribes a set of models as the artifacts to be produced during the system development process. Model-driven methodologies have gained increasing popularity, even more so after the Model-Driven Architecture (MDA) (OMG, 2001; Frankel, 2003) initiative was launched. Our approach to the above described challenge was to exploit this model-driven aspect to develop a generic framework that provides utilities for tailoring model-driven methodologies in a formal and standardized way. Using the framework, the tailoring will only affect the expression of the models prescribed by the general purpose methodology.

FRAMEWORK DESCRIPTION

By applying the tailoring framework, a domain-specific reference model is produced. The reference model describes the extensions of the actual general purpose methodology made for a specific domain. It consists of UML-profiles, existing (reusable) models, and patterns.

The set of UML-profiles, existing models, and patterns defined in a reference model are developed, structured, and aligned according to the chosen general purpose software engineering methodology. UML-profiles are used for defining domain concepts and reference architectures. Existing models are prepared for reuse, and patterns describe standard solutions of recurring problems within the domain. Thus, tailoring a software engineering methodology using the framework constitutes a leveraging of the methodology in an environment of domain concepts, defined reference architectures, existing models, and patterns.

Figure 1. Example of framework usage