Chapter II

A Generic Framework for Defining Domain-Specific Models

Arnor Solberg
SINTEF Telecom and Informatics, Norway

Jon Oldevik
SINTEF Telecom and Informatics, Norway

Audun Jensvoll
EDBTTelesciences, Norway

ABSTRACT

How do you tailor a general-purpose system development methodology to appropriately fit the specific needs of your company and the actual domain or product-family you are working with? Moreover, how do you alter a general-purpose methodology to utilize the domain knowledge possessed by your company? This chapter describes a generic framework for tailoring general-purpose, model-based methodologies in order to deliver domain-specific models.

INTRODUCTION

As a result of the widespread popularity of UML, 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, 1999), RUP (Rational, 1996), Catalysis (D’Souza, 1998), Select perspective (Allen, 1998), and KOBRA (Atkinson, 2000). Typically, these general-purpose software system development methodologies do not immediately fulfil a company’s need. That is why many consultants, researchers, and others are in the business of helping companies to
introduce these methodologies as well as to customize general-purpose methodologies as appropriate for the actual company and its purpose. A common way of customizing a general-purpose methodology is by removing, adding, and/or merging defined tasks, phases, roles, and models/artifacts of the methodology. The customization is typically tuned based on different criteria such as domain, kind of customers, market (e.g., in-house or off-the-shelf), quality demands, size of the company, and size of the software development teams. Although introduction of a general-purpose methodology typically requires a customization effort, there does not seem to be any standard and formalized way of customizing. The aim of the customization is typically to get a methodology that fits the purpose, fits the company, and fits the people who are going to use it and make the software development efficient. A key to achieving this is to get the methodology to support a process that delivers the wanted artifacts/models (including specifications, documentation, executable increments, and the final products) comprising the right quality. As a result, how should you, in a standard and formalized way, customize a general-purpose methodology to produce domain-specific artifacts/models?

In our research group at SINTEF, we have for some time worked with customizing methodologies to satisfy specific needs. Customization has been accomplished by utilizing experience attained through the usage of a set of different OO-based methodologies, such as RUP (Rational, 1996), UP (Jacobson, 1999), OOram (Reenskaug, 1996), Select Perspective (Allen, 1998), Catalysis (D’Souza, 1998) and Open Process (Graham, 1997), as well as self-developed methodologies. This has been combined with general methodology expertise and experience as input to a collaborative process with super-users (users equal developers who will use the methodology). By massaging this input using an iterative and incremental process, we have analysed the company’s need, existing methodology (or practice) used by the company, company’s culture, particularities of the domain, customers, market etc., to output a tailored methodology. Some results of this work have been the Business Object Methodology (BOM) (Solberg, 1999) and the Magma methodology handbook (Hallsteinsen, 2000). Recently we have been working with Telenor and EDB4Tel with a methodology called TeMOD (Solberg, 2001). This methodology is now in widespread use within the Telenor group. What we discovered during our work with developing TeMOD for Telenor 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 for more tailoring of TeMOD to fit the purpose of different domains and product families within the Telenor group. A main request was to obtain a methodology that was tailored to utilize existing domain knowledge. However, one of the goals of making TeMOD was to use a common methodology throughout the company to achieve a common way of developing and specifying systems. Therefore, it was clear that we didn’t want to end up with a set of proprietary, special-purpose methodologies (i.e., one for each domain and system development group). Our challenge became keeping TeMOD as the common methodology for the company, producing specifications in a standard way, and simultaneously getting TeMOD to support specific needs and utilize the existing domain knowledge.

The most popular general-purpose, UML-based software engineering methodologies have both diversities and commonalities. One frequent commonality is that they are model-driven. A model-driven methodology signifies that the methodology prescribes a set of models to be produced during the system development process. Model-driven methodologies have gained increasing popularity, particularly even more so after the MDA (OMG, 2001) initiative was launched. TeMOD is indeed a model-driven method-
The Debugging of Rule Bases
www.igi-global.com/chapter/debugging-rule-bases/35864?camid=4v1a