The Journal of Systems and Software 121 (2016) 49–71

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Flexible software process lines in practice: A metamodel-based approach to effectively construct and manage families of software process models

Marco Kuhrmann[a,*], Thomas Ternité[b], Jan Friedrich[c], Andreas Rausch[b], Manfred Broy[c]

[a] The Mærsk Mc-Kinney Møller Institute, Section Software Engineering, University of Southern Denmark, Odense, Denmark
[b] Department of Informatics, Technische Universität Clausthal, Clausthal-Zellerfeld, Germany
[c] Faculty of Informatics, Technische Universität München, Garching, Germany

ARTICLE INFO

Article history:
Received 19 September 2015
Revised 19 July 2016
Accepted 20 July 2016
Available online 3 August 2016

Keywords:
Software process metamodel
Software process
Software process lines
Process design
Process realisation
V-Modell XT metamodel

ABSTRACT

Process flexibility and adaptability is a frequently discussed topic in literature, and several approaches propose techniques to improve and optimize software processes for a given organization- or project context. A software process line (SPrL) is an instrument to systematically construct and manage variable software processes, by combining pre-defined and standardized process assets that can be reused, modified, and extended using a well-defined customization approach. Hence, process engineers can ground context-specific process variants in a standardized or domain-specific reference model that can be adapted to the respective context. In this article, we present an approach to construct flexible software process lines and show its practical application in the German V-Modell XT. The presented approach emerges from a 10-year research endeavor and was used to enhance the metamodel of the V-Modell XT and to allow for improved process variability and lifecycle management. Practical dissemination and complementing empirical research show the suitability of the concept. We therefore contribute a proven approach that is presented as metamodel fragment for reuse and implementation in further process modeling approaches.

1. Introduction

Software development is characterized by its diversity and, therefore, defining the optimal approach to develop software is for years subject to debate. As there is no “silver bullet” and software processes must reflect the needs of particular software projects, software processes need to be flexible and adaptable. For example, Armbrust and Rombach (2011) mention the selection of the software process in respect to the actual project context crucial and several studies (e.g., Cusumano et al., 2003; Jones, 2003; Kuhrmann and Linssen, 2014; Vijayasarathy and Butler, 2015) show companies usually using more than one development approach to address the manifold challenges of modern software & systems development. Notably, Kuhrmann and Linssen (2014) and Vijayasarathy and Butler (2015) show that companies use *hybrid approaches* in which traditional and agile software development approaches are used in combination, and Diebold et al. (2015) show that customization is relevant even for agile methods. Due to lacking empirical data (Theocharis et al., 2015), information regarding combination patterns, rationale regarding the process requirements leading to particular combinations, and descriptions of actual approaches to integrate traditional and agile development approaches is scarcely to find. For instance, Vijayasarathy and Butler (2015) found a frequent use of the classic Waterfall model and Scrum (West’s *Water-Scrum-Fall*; West, 2011)—why and how is such an integration done?

Such information is especially precious for companies that develop software in regulated domains, such as automotive software, avionics, space, or medical devices (Cawley et al., 2010; Houston, 2014). Those domains require companies to adhere to certain standards and to show their software development maturity, e.g., by presenting proper CMMI (CMMI Product Team, 2010) or ISO/IEC 15504 (ISO/IEC JTC 1/SC, 2004) certification. Standards have some inherent characteristics: they must be *universal*, i.e., applicable in many different setups, and they must be *adaptable* in order to tailor a standard to a particular organization- or project context. These characteristics are also true for the software pro-

* Corresponding author at:
E-mail addresses: kuhrmann@acm.org (M. Kuhrmann), thomas.ternite@tu-clausthal.de (T. Ternité), friedrij@in.tum.de (J. Friedrich), andreas.rausch@tu-clausthal.de (A. Rausch), broy@in.tum.de (M. Broy).

http://dx.doi.org/10.1016/j.jss.2016.07.031
0164-1212/

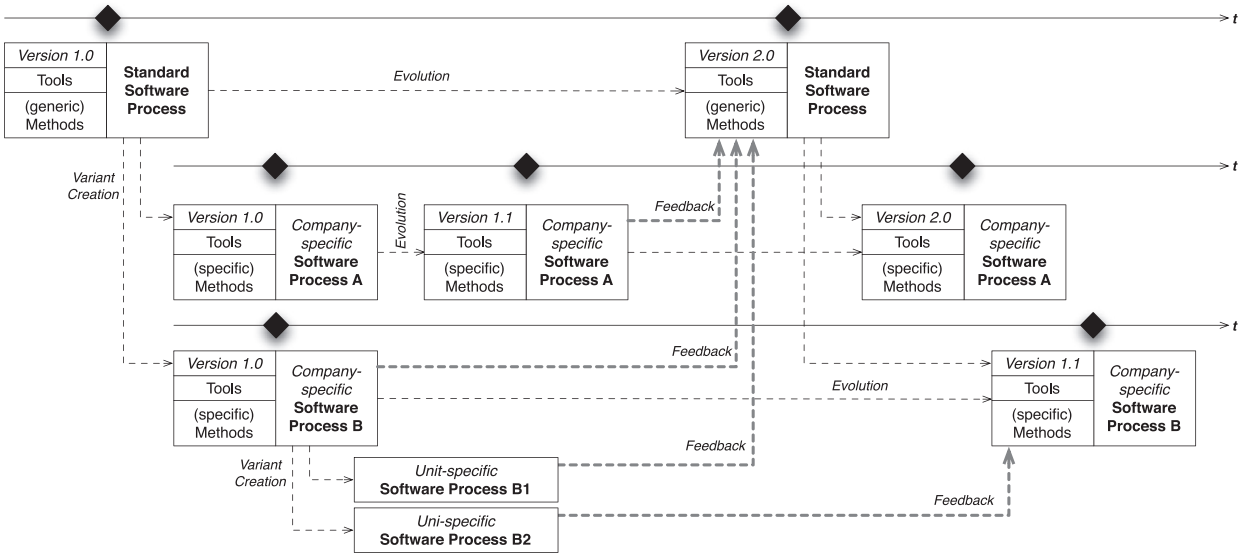

Fig. 1. Example of evolving process variants within a software process line.

cess, especially in aforementioned domains to proactively ensure software quality attributes, such as safety and security (see for instance process-related requirements of CMMI levels 3, 4, and 5).

Fig. 1 illustrates this situation from the perspective of so-called *software process lines* (SPrL; Rombach, 2005). A (generic standard) process is centrally managed and released, and several organizations create variants of this particular standard. The organization-specific variants each follow an own lifecycle and, potentially, serve as organization-specific reference model of which further variants can be created, such as one variant for "normal" software projects and another one for safety-critical systems with special compliance and certification needs. Furthermore, the figure shows that variants also provide valuable feedback to the standard model supporting its improvement. However, later in the processes' lifecycle, new versions of the variants have to respect the evolution of the standard as well as the organization-specific evolution.

Problem Statement & Objective. Standards, such as software processes, need to be adapted to the respective organization- and/or project context. While the term "process tailoring" (according to Ginsberg and Quinn, 1995) serves as umbrella for different approaches to construct organization- or project-specific software processes, still, the evolution of standards and processes and the impact on deployed process models is barely addressed. That is, companies face problems when an adapted standard process evolves (cf. Ocampo and Soto, 2007; Ocampo et al., 2009; Ocampo and Münch, 2009). In 2005, Rombach presented an approach that proposes adopting principles from software product lines (Bass et al., 1997; Software Engineering Institute, 2012) to software processes. Accordingly, a *software process line* (SPrL) is based on product-line principles, i.e., *"[...] by means of a domain engineering process, create a generic (set of) process(es), which capture the commonalities and controlled variabilities across a domain. The variabilities—and thereby the discriminators for process instances—in the case of processes are product & process goals and project characteristics [...]"* (Rombach, 2005). Software process lines are a promising route toward managing families of evolving software processes. However, Carvalho et al. (2014) conducted a systematic literature review finding SPrL-related research still immature, notably lacking in practically relevant and evaluated approaches.

Our overall goal is thus to provide process engineers with a toolbox to help design, implement, and manage software process lines. For this, we make use of existing process tailoring instruments, and propose an extension at the process metamodel level to inject SPrL-concepts to support organization-wide and standards-based software process lines.

Previously Published Material. This article presents evaluated work from long-term problem-driven research. It is grounded in Kuhrmann (2008) in which we proposed new modularity and lifecycle concepts and Ternité (2010) in which these were extended by different process variability instruments. Both concepts together built the basis for a substantial update of the metamodel of the V-Modell XT (Ternité and Kuhrmann, 2009). Over the years, the proposed concepts were disseminated and applied to practice. When a sufficient dissemination and a certain amount of experience could be expected, in Kuhrmann et al. (2011), we conducted a study on the general dissemination of the process models. In Kuhrmann et al. (2014), we conducted an initial study on the use of the variability instruments, which we continued with Schramm et al. (2015b) to observe long-term effects. However, so far, no integrated article was published that explains all concepts together. Other than in our previously published material, in this article, we present a practically applied and evaluated approach as a whole. We describe all steps from the solution development, the solution (illustrated by example), and its evaluation in practice.

Contribution. In this article, we contribute an approach to extend software process metamodels to provide software process line concepts to *process engineers* to support software process development and management. The approach presented extends existing process tailoring instruments by the two concepts *Partitioned Software Process* and *Variability Operation* providing a systematic approach to organize process variants within a software process line and to declare required modifications of a standard process model. The concepts presented allow for extensive software process lifecycle management, inter alia, including automatic updates or appraisal support due to traceable change logs. We implemented both concepts in the V-Modell XT, which is the standard software development process for IT projects in the German public sector. Using the V-Modell XT, we show by example how these concepts are applied to a process metamodel. Furthermore, we provide key results from a long-term study of an evolving V-Modell-XT-based process line to investigate the concepts' feasibility. The implementation in