Process-Driven Feature Modeling for Variability Management of Project Environment Configurations

Thomas Moser, Stefan Biffl, and Dietmar Winkler
Christian Doppler Laboratory for Software Engineering Integration for Flexible Automation Systems
Institute of Software Technology and Interactive Systems, Vienna University of Technology
Favoritenstrasse 9/188, 1040 Vienna, Austria
{thomas.moser, stefan.biffl, dietmar.winkler}@tuwien.ac.at

ABSTRACT
Technical projects environments, i.e., sets of methods and tools that support an engineering project, are software-intensive systems that need to be configured according to software process and project characteristics. Tailored software processes, e.g., based on the V Modell XT framework, specify project process steps and drive method and tool selection with a focus on individual feature requirements. Therefore, feature models can support the automated selection and configuration of methods and tools. For designing an effective and efficient engineering project environment, project managers and engineering domain experts can semantically integrate a given set of engineering tools and project data models in a flexible way. In this paper, we analyze challenges of managing engineering tool variability in context of engineering project environment configurations and present a conceptual approach using semantic modeling of project requirements and tool capabilities.

Categories and Subject Descriptors
D.2.9 [**Management**]: Software Configuration Management.

General Terms
Management, Design, Theory.

Keywords
Feature Modeling, Project Environment Variability Management.

1. INTRODUCTION
Technical project environments aim at making software project planning and execution more efficient and effective [3, 12]. Software environments are software-intensive systems as multiple methods and tools have to collaborate efficiently on technical and semantic levels to support project participants. As Software Product Lines (SPL) support a strategic reuse of software artifacts in a specific domain to enable a faster and cheaper delivery of solutions on a higher level of product quality [5], project environment configuration can be considered as a "product line approach" to enable efficient and effective project execution support on process level. A technical project environment configuration consists of four parts (see Figure 1):

1. *Tailored Process Approach.* The process approach defines process-related project execution strategies and required process units [4], i.e., process components, encapsulating process deliverables (products), activities to support product construction, and product responsible roles. Process units are core components of the V-Modell XT (a standard process model for software and systems projects) and enable flexible arrangement of components according to project types.
2. *Best-Practice Method Support:* Best-practice methods, aligned with tailored processes and project characteristics, enable effective and efficient construction of deliverables [2].
3. *Tool Support:* Tools – often heterogeneous derived from different engineering disciplines – support method application, collaboration, and project execution [3].
4. A *Feature Model [8]* aims at providing a link between appropriate methods and candidate tools to support project engineering. For instance, methods from requirements management need features to specify individual requirements and can enable tracing between these requirements. Candidate tools must provide these features to some extent. A feature model aims at supporting mapping of requested (method) features and provided (tool) features to identify a best-practice method/tool setting in the project context.

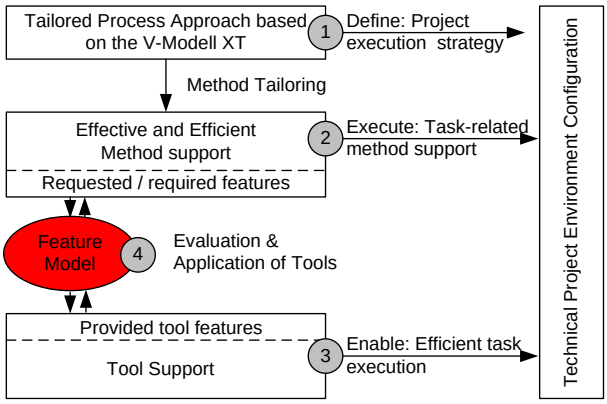

Figure 1: Project Process – Method – Tool-Support.

The considerable variability of candidate methods within an engineering project scope and an even higher number of candidate tools require appropriate approaches for linking methods, tools, project data models, and the engineering process to provide an efficient and effective project environment configuration. Based on lessons learned with SPL research, we assume a high potential of feature models to provide this missing link [8]. In this paper, we analyze the challenges of managing engineering tool variability in the context of engineering project environment configuration. Additionally, we conceptually apply the proposed approach to this context and present our findings regarding strengths and limitations.

The remainder of this paper is structured as follows: Section 2 summarizes related work on SPL in the context of project environment configuration, a process-driven approach based on the V-Modell XT, and the application of variability management with semantic techniques. We illustrate the basic research challenges from real-world use cases in Section 3 and present a solution approach in Section 4. Finally, Section 5 discusses the findings, concludes and identifies further research work.

2. RELATED WORK

This section summarizes related work on (a) variability modeling in software product lines, (b) a process-related approach based on the V-Modell XT, and (c) feature models with semantic modeling aspects.

2.1 SPL and Variability Management

Product line software engineering (PLSE) is an emerging software engineering paradigm, which guides organizations towards the development of products from core assets rather than the development of products from the scratch. Two major activities of PLSE are core asset development (i.e., product line engineering) and product development (i.e., product engineering) using the core assets [5]. In order to develop reusable core assets, PLSE must provide the ability to use commonalties and manage variability. Although core assets are built for a product line, they have to be constructed with an understanding of the domain, which provides a wider engineering perspective for reusability and adaptability than a product line. Therefore, domain analysis, which identifies commonality and variability from a domain perspective, is a key requirement for reusable core asset development for product lines [8].

Kang *et al.* established feature-oriented domain analysis (FODA) [7], which identifies and classifies commonalities and differences in a domain in terms of "product features." Feature analysis results can be used to develop reusable assets for the development of multiple products in the domain.

2.2 Process Tailoring using the V-Modell XT

Project environment integration requires a well-defined baseline, e.g., a software process model for guiding the project course. The modular V-Modell XT[1] (VMXT) concept enables the individual customization of a project environment with respect to the application domain, project characteristics, project views, and organization-specific requirements [4]. Mandatory and optional process units encapsulate products (product-centric approach) and related activities, define responsible roles, and represent the basic components of the process model approach [4]. A project execution strategy defines the sequence of steps regarding the project course separated by defined decision gates. Note that passing each decision gate requires a set of deliverables at a certain state of completion (including compliance with defined quality criteria).

The modular structure of the VMXT enables process tailoring and the adjustment of the VMXT to individual project requirements and defines a sequence of project steps including required deliverables and activities. Method and tool support is sketched by the VMXT framework, but there is still a lack of implementation of method and tool support. Because of this process's flexibility we see the VMXT framework as a valuable foundation for project environment configurations. However, a key challenge is to handle the variability in configuring and combining best-practice methods and tools for successful project application. Nevertheless, all candidate methods and tools have strength and weaknesses regarding individual method and tool characteristics. Thus a well-defined evaluation and tool selection approach is required to focus on individual needs of the technical project configuration and the project.

[1] See http://www.v-modell-xt.de for a complete description of the process approach.

2.3 Feature Models and Semantic Modeling

There are several reasons why feature-oriented domain analysis has been used extensively compared to other domain analysis techniques. First, features are essential abstractions that both customers and developers understand, and therefore should be first class objects in software development. Secondly, feature-oriented domain analysis is an effective way to identify variability (and commonality) among products in a domain. Finally, the feature model can provide a basis for developing, parameterizing, and configuring various reusable assets (e.g., domain requirement models, architectural models, and reusable code components) [8].

In order to increase the quality of product line variability models and to improve the product derivation process, researchers have started investigating the use of ontologies in SPLE. For example, Czarnecki *et al.* [6] have explored the relationship between feature models and ontologies. They analyzed the notational spectrum of feature models and ontologies and derived the idea that feature models are views on ontologies. Czarnecki *et al.* suggest using ontologies as views on feature models to provide semantics for potentially overlapping feature models and support querying and constraint mechanisms for these overlapping feature model parts. Peng *et al.* [11] enriched feature models with ontologies to increase feature models' expression capacity. By converting feature models into ontology models, the authors provide a foundation for different mechanisms to validate feature models through ontology inference. However, one of the core problems is to agree on a common description language to describe features of potentially similar assets, using heterogeneous terminologies.

3. RESEARCH ISSUES

Our observations in industry projects have shown that organizations often use tailored software processes following industry sector and/or company standards, apply individual methods and tools but without comprehensive view on the technical project environment configuration. Process tailoring, method selection, and tool application is typically based on the individual experience of different roles, e.g., project managers, method specialists, and tool engineers, without considering variation points of methods and tools. The missing overview on connected decisions and configuration activities often leads to project environment that do not work well.

Therefore, an integrated view can improve project planning including the selection and application of best-practice methods and tools sets including variability considerations of methods and tools with feature models across different engineering disciplines and roles. We expect the following benefits from an integrated

approach: (a) more efficient project tailoring from individual project characteristics and (b) better fitting selection and configuration of tool sets to support method application. Thus, project environment configuration requires (a) suitable and tailored process approach, (b) appropriate methods and tools (identified by features), and (c) a semantic integration and link of features in terms of a feature model. From these requirements we derive the following research challenges:

1. Definition of a *process-driven approach* for project environment configuration. The VMXT aims at providing a framework for individual process tailoring according to project needs (project process steps) and provides a basic framework for method and tool integration. Nevertheless, a main open issue is how to couple method and tool assignment and configuration with individual process steps.
2. Handling of the project environment configuration process as a *variant of SPL on process level*. Based on the SPL approach we see various technical project environment configurations including variants of candidate method and tool sets and variation points according to individual method and tool characteristics as a promising approach for proposing a "product line" for process- and project-related engineering requirements. This research issue focuses on a possible implementation of a process-related product line approach.
3. Realization of project environment configuration with *feature models* and *semantic techniques*. Individual characteristics of methods and tools hinder an efficient (automation-supported) selection of method and tool sets. Thus we see the need for introducing feature models and semantic techniques to enable automated mapping with respect to project characteristics.

As research approach we present a solution concept based on a real-world showcase and propose a concept for empirical evaluation of the solution concept.

4. SOLUTION APPROACH

This section introduces a solution approach including a process-driven application of project environment integration in context of SPL, applying techniques from semantic feature modeling.

4.1 Project Environment as Process Line

From SPL, we see the selection and mapping process as some kind of product line for processes, i.e., similar projects (e.g., comparable project types and application domains) which might apply similar tool sets and project environment configurations. Because of these similarities we applied the project environment configuration to SPL engineering [1]. Figure 2 presents this concept in a brief overview. The conceptual view in Figure 2 is based on a simplified version of the SPL meta-model according to Alves *et al.* [1] proposed by Muthig [10] and applies the concept of project environment integration.

The upper block illustrates a general approach on various project environment configurations and the lower block shows a detailed configuration with respect to a particular project. The project environment includes a certain tool set for a specific application domain. Infrastructure and asset lead to a feature model (variability of various tool sets and their features) and candidate components for a project environment configuration. Architecture refers to the technical implementation framework for tool integration (e.g., the Engineering Service Bus [3]). Candidate components are linked to expected tool characteristics. In this paper we focus on early binding during project execution. Alternative approaches might focus on the need for changing tool characteristics during development (late binding) and at runtime (e.g., diagnosis components).

The lower block presents a project view for individual project environment configuration engineers. The engineer derives a suitable project environment configuration based on individual project requirements, the set of available tools (derived from the feature model), proposed tool variants, and candidate components.

Note that the feature model aims at linking method and tool requirements (requested by the tailored software process) as well as tool and method features provided by the individual components. A remaining question is the structure of the feature model and how the feature models can be implemented to efficiently support product environment configurations.

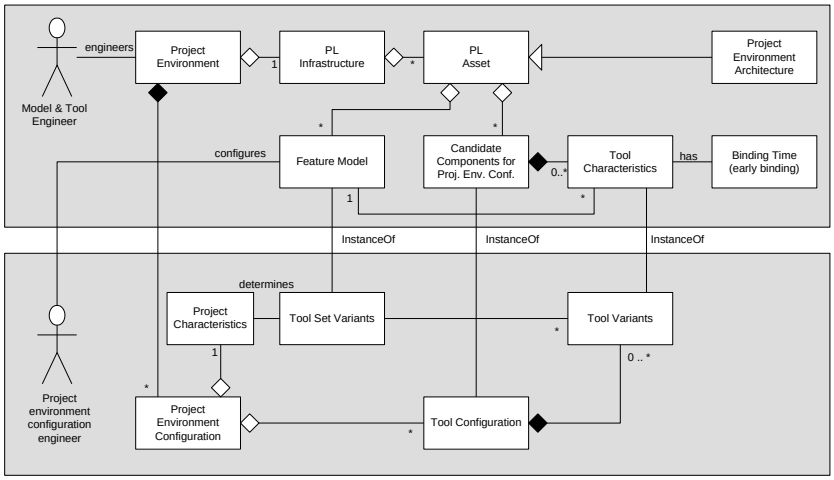

Figure 2. Project Environment Integration with SPL.

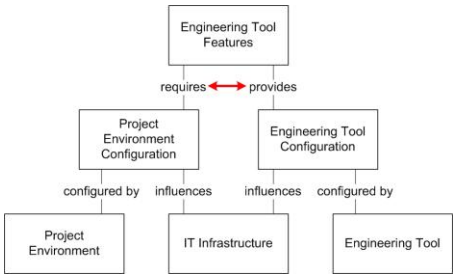

Figure 3. Engineering Tool Feature Modeling.

4.2 Semantic Feature Modeling

Figure 3 shows the interplay of required and provided engineering tool features. A typical engineering project has a defined engineering project environment and uses the available IT infrastructure. Based on these descriptions, a project environment configuration can be derived, which specifies the engineering tool features required by the particular engineering project. In contrast, typical engineering tools provide a set of engineering tool features, which again are trimmed by both the available IT infrastructure, as well as by the configuration of particular engineering tools.

The challenge is to identify from the set of candidate tools and their configurations the set which is able to provide all (or most) required engineering tool features for a given project and available IT infrastructure. In [9], we introduced an approach for semi-automatic semantic matchmaking for software services in the Air Traffic Management (ATM) domain, which can provide system designers with a set of promising matching software service candidates and therefore strongly reduces the human matching effort by focusing on a much smaller space of matchmaking candidates.

5. DISCUSSION AND CONCLUSION

A technical project environment configuration including process, method and tool support (adjusted to each other) is the backbone for an efficient and effective project execution. In industry practice we observed that each topic requires individual experts but there are limitations on a comprehensive view on the technical project environment configuration. Additionally, individual variation points of methods and tools seem not to be considered (variability aspects). Therefore, we proposed a framework for setting up a process-driven approach for variability of project environment configuration based on feature models and semantic techniques. Expected benefits are:

1. *Systematic approach for environment configuration.* Individual process tailoring based on a well-known software process (V-Modell XT) enables a well-suited sequence of process steps according to individual project needs. Additionally, the modular configuration enables an integrated view on methods and tools within the project configuration.

2. *Application of successful SPL principles.* Based on reports from SPL, we see similarities of product line and project environment configurations as individual methods and tools include variation points, which have to be adjusted to each other. Similar projects can be based on a common (organization specific) project management and engineering base; individual tailoring according to a specific project enable a project specific selection of tool-sets.

3. *Semi-automated support for tool selection and configuration.* Candidate methods and tools include individual features and characteristics. Thus, there is a need to map these individual characteristics to identify a best-practice method/tool set in a given project context. Nowadays, this mapping is done be individual roles (e.g., project manager, method experts, and tool specialists) manually. Feature models and semantic techniques can support this selection and evaluation process automatically.

In our work we see technical projects environments as software-intensive systems that need to be configured and have to adhere to software process and project characteristics. In this paper we analyzed the basic characteristics of project environment configuration based on the V-Modell XT framework and presented a feature-model based approach and illustrating showcase for the selection and configuration of methods and tools from a candidate set. The feature models in combination with semantic integration approaches were shown to principally support the automated selection of methods and tools during project configuration and planning. We analyzed the challenges of managing engineering tool variability in the context of engineering project environment configuration and presented a conceptual approach using semantic modeling of project requirements and tool capabilities.

In our further research work we will evaluate the proposed feature modeling approach with practitioners in the use case domain to find out whether the approach seems usable and useful in a real-world context.

6. REFERENCES

[1] V. Alves, D. Schneider, M. Becker, N. Bencomo, and P. Grace, "Comparitive Study of Variability Management in Software Product Lines and Runtime Adaptable Systems," *3rd International Workshop on Variability Modelling of Software-Intensive Systems*, 2009, pp. 9-17.

[2] S. Biffl, C. Denger, F. Elberzhager, and D. Winkler, "Quality Assurance Tradeoff Analysis Method (QATAM)-An Empirical Quality Assurance Planning and Evaluation Framework," *Euromicro SEAA, Work in Progress*, 2007, pp.

[3] S. Biffl, A. Schatten, and A. Zoitl, "Integration of Heterogeneous Engineering Environments for the Automation Systems Lifecycle," *IEEE Industrial Informatics Conf., 2009*, 2009, pp. 576-581.

[4] S. Biffl, D. Winkler, R. Höhn, and H. Wetzel, "Software Process Improvement in Europe: Potential of the New V-Modell XT and Research," *Software Process: Improvement and Practice*, vol. 11, no. 3, 2006, pp. 229-238.

[5] P. Clements, and L. Northrop, *Software product lines*, Addison-Wesley Reading MA, 2001.

[6] K. Czarnecki, C.H.P. Kim, and K.T. Kalleberg, "Feature models are views on ontologies," *10th International Software Product Line Conference*, IEEE CS, 2006, pp. 41-51.

[7] K.C. Kang, S.G. Cohen, J.A. Hess, W.E. Novak, and A.S. Peterson, "Feature-oriented domain analysis (FODA) feasibility study," *Technical Report CMU/SEI-90-TR-21*, 1990.

[8] K. Lee, K.C. Kang, and J. Lee, "Concepts and Guidelines of Feature Modeling for Product Line Software Engineering," *7th International Conference on Software Reuse: Methods, Techniques, and Tools*, Springer-Verlag, 2002, pp. 62-77.

[9] T. Moser, R. Mordinyi, W.D. Sunindyo, and S. Biffl, "Semantic Service Matchmaking in the ATM Domain Considering Infrastructure Capability Constraints," *21st International Conference on Software Engineering and Knowledge Engineering (SEKE 2009)*, 2009, pp. 222-227.

[10] D. Muthig, "A light-weight approach facilitating an evolutionary transition towards software product lines," *Series of PhD Theses in Experimental Software Engineering*, vol. 11, 2002.

[11] X. Peng, W. Zhao, Y. Xue, and Y. Wu, "Ontology-Based Feature Modeling and Application-Oriented Tailoring," *9th International Conference on Software Reuse (ICSR)*, Springer, 2006, pp. 87-100.

[12] I. Sommerville, *Software Engineering. 8th*, Addison-Wesley, 2006.