Integration Mechanism for Software Product Line Development and Management Workbench

Jeong Ah Kim¹ and Jin Seok Yang²

¹Computer Education Depart., 522 NaeKok-Dong, GangNung-City, Kangwon-Do, Korea
²93st. 27, 4th floor, SunNeungRo, KangNam, Seoul, Korea
clara@cku.ac.kr, edeward@gmail.com

Abstract

To Software product line (SPL) engineering process be efficient, integrated environment supporting SPL is necessary. So far, domain engineering process supporting tools or specific purpose tools such as variability management, feature modeling have been suggested. In this paper, we suggested SW Visualization tools as application engineering supporting tools and VULCAN workbench as domain engineering supporting tools. And we suggested integration mechanism between well-defined open source software tools.

Keywords: We would like to encourage you to list your keywords in this section

1. Introduction

The approach of software product line (SPL) is ideal for software domains where the objective is to develop a set of products that have common characteristics and variable parts. Using this approach it is possible to develop systems with much less time and effort compared to traditional development approaches (focusing on a single system) [1]. SPL is a SW development paradigm that began as an attempt to maximize quality properties such as reusability and maintainability of SW [2].

For managers and working-level staffs who work for software developers, a survey on the necessity for engineering in software product line (SPL) was conducted. The purpose of the survey is to analyze the recognition and current status of application of SPL which is possessed by domestic software companies, and to find necessary supports required for introducing and applying SPL to as many firms as possible [3]. The survey was conducted for managers and working-level staffs in domestic software-developing companies. Most companies were applying their software products to the field of automobile, and the average career of respondents was 14.95 years. Most of the activities of the respondents were focused on quality assurance, software process improvement, and software test. Among the respondents very few were well aware of SPL engineering, while vast majority knew the general outline or have heard of it. But no company was applying the SPL engineering in their software development. Many companies are applying object-oriented methodology and structural methodology as the methodology for software development, while the SPL engineering is not applied. The followings are the top-4 ranking reasons why the SPL engineering is not applied. 1) Insufficient technical knowledge and technical data and environment for SPL engineering, 2) Insufficient learning, 3) Needs of organization deactivated, 4) Only few successful cases.

To make the application of the SPL engineering easy, a step-by-step implementation method and procedure guideline of SPL engineering methodology should be distributed among the software development companies, and an integrated environment supporting the SPL engineering methodology should be provided to help companies easily apply the same
SW Visualization is an integrated development environment designed to enable effective product development management by encouraging the application of SW engineering and development, management, product quality, and configuration management process to the entire product development cycle. In this integrated environment, an organization with insufficient SW engineering and theoretical foundations can effectively develop and manage its products through the systems-based product development management. [4].

Using only the SPL-based development methodology has a limit to the application and expansion of SPL to the development process in practice [5, 6, 7]. Thus, there is a need to integrate the SPL-based analysis/design modeling tools into the SW Visualization tools in order to improve the efficiency of development process and to visualize the process based on the execution of each stage.

In Section 2, we describe the reason why we select software visualization framework to integrate with SPL environment. In Section 3, we briefly explain the existing SPL tools and their limitations. Our integration strategy and architecture is suggested in Section 3 and workflow based on our integrated development environment is explained in Section 4 and Section 5 concludes our research.

2. Software Visualization Framework and Software Process

2.1. Software Visualization Framework

SW Visualization framework provides a successful SPL-based application development environment that enables a continuous monitoring and control via effective development activities using system tools and visualization method.

![Figure 1. BOM of SW Visualization Framework](image)

By using SW Visualization tools, SW development companies can overcome SW invisibility and ensure the transparency in SW development process. Creating a transparent SW development process can help the companies to secure their competitive advantages by reducing SW development costs through SW quality assurance and early
detection of issues that may arise in the development process [4]. Figure 1 shows the main building blocks of SW Visualization framework.

Ensuring visibility throughout the entire product development process can also help the companies to decrease the development cost as they can promptly identify and solve the problems in the product.

2.2. Software Process Assessment Model

In Korea, for software process improvement, Software Process (SP) quality assurance is being conducted by NIPA SW engineering center [8]. The SP quality assurance has been developed based on CMM and SPICE and is being used for examining the SP quality competence level of software companies and development organizations. It has been introduced to simplify the examination procedure compared to CMM and SPICE, so that small businesses can easily improve their process and reduce costs for the quality assurance, based on the simplified procedure [9]. SP model defines 3 stage for quality maturity. Level 2 means that project defines the well-defined process for performing that project and control the whole lifecycle of project based on the project standards. Level 3 means that organization defines standard process and each project tailors to establish the project process.

3. Related Tools

To develop a SPL according to the product line technology, a development environment that supports the process of product line development is required. This study analyzed the functions and advantages and disadvantages of the tools that provide prerequisite technologies for the application of the existing product lines prior to the introduction of a new tool. Most tools support feature modeling, requirements, architecture, and connection of artifacts such as source code with variable messages, product derivation or product instantiation technologies, and extraction of variable messages from legacy SW. They import a directory tree of source code into a family model and support both C/C++ and Java project.

3.1. Pure::variants

Pure::variants extracts variable messages from the contents of the files created by all sets of tools interoperable to the framework and maintains data on the relations with the features [10]. Pure::variants basically uses Microsoft Excel, Word, and IBM Rational DOORS and provides Add-in Connectors that enable feature selections of each tool in order to support the variability management in architecture design. For variability modeling in the design process, if statements can be expressed by using notes that are available on UML or SysML diagrams and by using OCL grammar constraints in general. Pure::variants presents feature trees to select each feature as an Add-in for IBM Rhapsody or Enterprise Architect [11].

3.2. GEARS

GEARS creates variable information files of each project and separately manages the contents of assets from the information [12]. By using PLE Bridge API, it supports interface with IBM Rationale DOORS. GEARS provides a Bridge that supports the model-based design such as SysML and UML standards. The Bridge is a collective name for APIs that allow GEARS to interface with Third-Party tools. The source code variable information can be managed according to the management policy that separates the information on unique variables from the assets and provides a UI environment integrated with Visual Studio [13].
3.3. VULCAN Workbench

VULCAN Workbench identifies files in a certain location of a Project as particular assets and displays the relationship of a feature model by using VULCAN Project [14]. VULCAN Workbench supports Microsoft Word and text-based document formats in general and variable information can be expressed directly to the requirements by using Feature Macro Expression [15]. VULCAN Workbench supports StarUML that enables to implement the FORM Architecture Model-based design as a concept of product line. VULCAN supports C/C++ and Java languages to specify variables and provides an independent editor.

3.4. FeatureIDE

FeatureIDE is a tool that focuses on the relationship management of the source code implementing a given feature [16]. It maintains the feature-source code relationship by using variable information (Annotation, etc.) included in the source code when it is required only and does not permanently store the data. As FeatureIDE is developed as an extensible environment for FOSD [17], it supports multiple programming languages and methodologies. For instance, FeatureIDE supports C++, Java, C#, JML, Haskell, and XML.

3.5. Implications

The existing tools mostly provide an integrated environment using highly expensive frameworks. Among them, VULCAN workbench solely offers an integrated development environment based on open source software. Considering that tools are the most effective ways to improve the productivity in software development, a high price can be a factor that hinders the adoption of development tools.

Also, the existing tools are mainly used in the development stage and are rarely adopted in the areas of implementation integrity, and quality management in connection with testing tools, revealing their limits.

4. Integration Strategy SP Model and SPL Development Process

4.1. SPL Development Process based on SP Model

Software process quality is also critical as well as software product quality. To make the consistent and good quality SPL-based product, management and supporting process should be integrated with engineering process. SP quality assessment model provide the quality reference model for domestic software company. To promote the SPL project, we suggested integrated process model based on the SP. Figure 2 defines the integration scopes.
4.2. Integration Strategy

Ensuring visibility throughout the entire product development process is important as it enables to identify multiple issues occurred in the late development phase whether they occurred during the process of development or the initial requirement specification phase. To make this happen, the whole process of product development from the specification of requirements, development, and testing should be systematically connected and linked each other to ensure the traceability. In general, a product developed by using SPL-based tools is a package-based SW. Thus, the following SW Visualization system tools can be applied.

As for a new product that operates in multiple user environments based on particular markets or domains, sufficiency of features and user satisfaction should be used to maintain the product quality. As for the Customer Service Request (CSR) for the existing product and other multiple products, the quality maintenance activities should be carried out by product version and configuration management. Prevention of malfunctions or misuses caused by unspecified and multiple customers, adaptability to various environments, as well as the quality of code that extends based on the assets of the existing product are important.
Figure 3. V-Model based Integration Strategy

We suggested v-model based integration strategy to support full product line development lifecycle. Feature model should be driver to integration. Feature model should be traced into requirement, design model and code as well as test case. Labels below of phase are the candidate open-source software.

4.3. Integration Architecture

There are 2 principles to integrate SW Visualization Toolset and VULCAN Workbench: 1) loosely coupled architecture, 2) well-known and well-verified tools should be candidates for integration. SW Visualization Toolset and VULCAN Workbench are based on open-source software like eclipse but we tried to avoid license problem. Visualization Toolset can support application engineering process and VULCAN Workbench can support domain engineering process. So, it is not necessary to integrate UI. It results in reducing the integration costs.

Since SW Visualization Toolset and VULCAN Workbench are based on eclipse IDE so that we can easily integrate Myln, SVN, and CVS as collaboration tools. Myln can be good asset base for Redmine task. SVN server and client are very powerful tools for asset repository of SPL. But feature model, product configuration, variability information of asset, asset version information belonged to product are the additional item to asset repository. It means additional modules should be developed. Additional module can be developed by extension point. It is easier and more efficient way not to modify the existing tools. Redmine and Jenkins have extension points for plugin and we developed bridge and VULCAN workbench provides SPLE information through this bridge. This architecture was described in Figure 4. Figure 5 shows the data integration mechanism between the SW Visualization Toolset and VULCAN Workbench. When we use the SW Visualization Toolset in domain engineering process, feature model, product line requirements, product line design model, asset version and change history are transferred to SW Visualization Toolset. If we use the SW Visualization Toolset in application engineering process, feature configuration, requirement instance, and design instance are delivered. Therefore, SPLE engineer can refer the asset information in asset repository.
5. Integration of SW Visualization tool into SPL Development Phase

Use of SW Visualization tool is required for effectively integrating the management of SP quality model and supporting process with the development of SPL, and for enhancing the visibility and quality improvement of product development projects. The SW Visualization uses the management of product development based on system tools, by inducing SW engineering as well as the application of configuration management process for development, management, and quality to overall product development, through the utilization of system tools even in situation which lacks the theoretical infrastructure for the SW engineering. Therefore, the SW Visualization contributes to efficient product development management. Since the introduction of SPL-based development method alone has limits in the application and dispersion of actual SPL development process, the efficiency of the development process should be increased by
the linkage between the SPL-based analysis/design modeling tools and SW Visualization tools, while also visualizing the development process.

Also, the visualization of the overall development process of a product enables the swift comprehension and response to the inherent problems of the product, thus reducing the costs for the product development.

![Integration Workbench of SSPL](Image)

**Figure 6. Integration Workbench of SSPL**

### 5.1. Tool Chains for Engineering Process

In the requirement specification and management phase, a process of identifying and managing functional and nonfunctional requirements of SPL-based product and its assets is included in this phase. With Redmine and other requirement Visualization tool, feature traceability mechanism should be added.

The step of design and realization is a process including: a domain engineering in which an actual developer develops the product assets based on SPL; and an application engineering process, in which an application is developed by reusing the assets in accordance with the product line. In this step, the convenience of the developer is regarded as the top priority. In this phase, VULCAN Workbench is powerful to design the reusable architecture of feature modeling, core assets of the identified features, and the product line, based on the requirements specified in the previous phase [18]. The VULCAN framework can be easily integrated with sets of SW Visualization tools by plug-in type tool. A static analysis to validate the reliability of the product assets or Application codes implemented based on the design can be conducted by using SW Visualization tools such as PMD. The adequacy and stability of the codes can be ensured by linking the result to a Visualization tool (Jenkins and others) [18].

The static analysis can be supported by SW Visualization tools such as PMD and CppUnit. In testing phase, the integrity of the product line assets and application functions can be verified by using Impasses, the Redmine plug-in for test management [18].

Product Build refers to a phase to build a product line by using or reusing the assets developed in the previous phase. To develop and manage product line-based assets and applications, a continuous integration to automate the build and deployment process is
critical in order to maintain and manage the product configuration based on the product line in a stable manner [18]. Jenkins, a SW Visualization tool, can be used for the continuous integration of automated build of the implemented product assets and application source code and testing.

5.2. Tool Chains for Supporting Process

Changing business environments and markets demand companies to change the product line and to continue to maintain their products. Prompt updates of a new product or multiple versions are required as well. Thus, configuration management of SPL must be included in the product development and maintenance process. The SPL-based configuration management refers to the activities to systemically control and manage the deliverables and results that are made during the process of developing assets and asset-based applications. Subversion and other SW Visualization tools can be used in this phase [18].

6. Conclusion

The SPL development process is basically composed of product line asset development process (domain engineering) and product line application development process (application engineering). In the development of SPL-based product, project management and supporting process can improve the entire development process. SW Visualization divides the process into requirement management, implementation, testing, and configuration management and visualizes each development phase. We suggest an integration mechanism between SW Visualization tools and product line engineering tools. This suggestion can be the reference model for integrated development of SPL.

Acknowledgments

This work was supported by research fund of Catholic Kwandong University. (CKURF- 201601150001) and SSPL Promotion and Spread Program through the National IT Industry Promotion Agency funded by the Ministry of Science, ICT & Future Planning

References


Authors

**Jeong Ah Kim**, he received Ph. D degree at ChungAng University. Since 1996, she has worked at Catholic Kwandong University as professor. She is the member of Korea Institute of Information Science and Engineering and the member of board of directors of Convergent Research Society. Her research areas are software product line engineering, software modeling, software process improvement, clinical decision support system.

**Jin-Seok Yang**, he received the MS degree from the KOREA University in 2006. Since then, he has worked as a researcher at the Pohang University of Science and Technology (POSTECH) in Korea. He is currently a consultant and developer for software engineering at the SPID. His areas of research interest include software product line engineering, engineering tool development and software design.