INTERNATIONAL STANDARD

ISO/IEC 26557

First edition
2016-12-15

Software and systems engineering — Methods and tools for variability mechanisms in software and systems product line

Ingénierie du logiciel et des systèmes — Méthodes et outils pour les mécanismes de variabilité dans les chaînes de production de logiciels et de systèmes

Reference number
ISO/IEC 26557:2016(E)

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

The committee responsible for this document is ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and systems engineering*.

Introduction

Software and systems product line (SSPL) engineering and management creates, exploits and manages a common platform to develop a family of products (e.g. software products, systems architectures) at lower cost, reduced time to market and with better quality. As a result, it has gained increasing global attention since 1990s.

Variability, which differentiates a member product from other products within a product line, plays an important role in SSPL. Variability mechanism means ways to implement variability; it realizes variability in the product line artefacts. Variability mechanisms differ in accordance with the binding time of variability, and variability of a product line is introduced from product line scoping through product line testing and its binding can occur at any stages of product line development. Thus, variability mechanism should be systematically managed for the right operation in domain engineering and for the right binding in application engineering. Furthermore, variability mechanisms should support easy variability management and traceability management. Accordingly, this document provides processes with their supporting methods and tools capabilities for variability mechanism operationalization and for managerial supports for the right use of variability mechanisms at domain engineering stages and the right bindings at application engineering stages.

This document can be used in the following modes:

— by the users of this document: to benefit people who want to adopt SSPL for producing their products by guiding variability mechanism operationalization, variability mechanism management and variability mechanism supports;

— by a product line organization: to provide guidance in the evaluation and selection for methods and tools for the tasks of providing variability mechanism operationalization, variability mechanism management and variability mechanism supports;

— by providers of tools and methods: to provide guidance in implementing or developing tools and methods by providing a comprehensive set of the capabilities of methods and tools for supporting variability mechanism operationalization, variability mechanism management and variability mechanism supports.

The ISO/IEC 26550 family of standards addresses both engineering and management processes and capabilities of methods and tools in terms of the key characteristics of product line development. This document provides processes and capabilities of methods and tools for variability mechanisms in product lines. Other ISO/IEC 26550 family of standards are as follows.

ISO/IEC 26550, ISO/IEC 26551 and ISO/IEC 26555 are published. ISO/IEC 26558 and ISO/IEC 26559 are under preparation. ISO/IEC 26552, ISO/IEC 26553, ISO/IEC 26554, ISO/IEC 26556, ISO/IEC 26560, ISO/IEC 26561, ISO/IEC 26562 and ISO/IEC 26563 are planned.

— Processes and capabilities of methods and tools for domain requirements engineering and application requirements engineering are provided in ISO/IEC 26551.

— Processes and capabilities of methods and tools for domain design and application design are provided in ISO/IEC 26552 (planned).

— Processes and capabilities of methods and tools for domain realization and application realization are provided in ISO/IEC 26553 (planned).

— Processes and capabilities of methods and tools for domain testing and application testing are provided in ISO/IEC 26554 (planned).

— Processes and capabilities of methods and tools for technical management are provided in ISO/IEC 26555.

— Processes and capabilities of methods and tools for organizational management are provided in ISO/IEC 26556 (planned).

- Processes and capabilities of methods and tools for variability modelling are provided in ISO/IEC 26558.
- Processes and capabilities of methods and tools for variability traceability are provided in ISO/IEC 26559.
- Processes and capabilities of methods and tools for product management are provided in ISO/IEC 26560 (planned).
- Processes and capabilities of methods and tools for technical probe are provided in ISO/IEC 26561 (planned).
- Processes and capabilities of methods and tools for transition management are provided in ISO/IEC 26562 (planned).
- Processes and capabilities of methods and tools for configuration management of asset are provided in ISO/IEC 26563 (planned).
- Others (ISO/IEC 26564 to ISO/IEC 26599): planned.

Software and systems engineering — Methods and tools for variability mechanisms in software and systems product line

1 Scope

This document, within the context of tools and methods of variability mechanisms for software and system product lines**:**

— provides the terms and definitions related to variability mechanisms for software and systems product lines;

— defines processes and their subprocesses for operating variability mechanisms at each product line life cycle stages and those for providing managerial supports. Those processes are described in terms of purpose, inputs, tasks and outcomes;

— defines method capabilities to support the defined tasks of each process;

— defines tool capabilities to automate/semi-automate tasks or defined method capabilities.

This document does not concern processes and capabilities of tools and methods for a single system, but rather deals with those for a family of products.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— IEC Electropedia: available at http://www.electropedia.org/

— ISO Online browsing platform: available at https://www.iso.org/obp/

3.1
application configuration
derivation for a member product specific executables from domain assets in *realization* (3.10)

Note 1 to entry: The specific configuration of an application is the *binding* (3.3) results for the *variation points* (3.19) with the selected *variants* (3.17).

3.2
aspect
special consideration within product line engineering process groups and tasks to which we can associate specialized methods and tools

3.3
binding
task for making a *decision* (3.7) on relevant *variants* (3.17) using domain *variability model* (3.16) and *decision tables* (3.8)

3.4
binding time
moment of variability resolution

Note 1 to entry: The choice of binding time is independent from variability modelling. It is the consequence of *decisions* (3.7) made from requirements through *run time* (3.11). Demands for flexibility and the support of tools allow late binding times or even the use of variable binding times.

3.5
binding time decision
selection for *variability* (3.13) defined in platforms in accordance with the functional distinction between variability in time and variability in space

3.6
configurability
degree of how well a *variability mechanism* (3.14) supports the configuration of a member product

3.7
decision
types of statements in which a choice between two or more possible outcomes controls which set of actions will result

3.8
decision table
table that specifies decision variables

Note 1 to entry: It also includes rules, constraints and relevancy among variables.

3.9
post-compile time
collective name for link time and load time that are right after the compilation of components

3.10
realization
stage for detailed design and construction

3.11
run time
stage that a member product is executed

Note 1 to entry: Components can be developed, compiled, linked and loaded separately. Only at run time are they combined into a working system.

3.12
texture
architectural texture
collection of common development rules and constraints for realizing the applications of a product line

3.13
variability
characteristics that may differ among members of a product line

Note 1 to entry: The differences between members may be captured from multiple viewpoints such as functionality, quality attributes, environments in which the members are used, users, constraints and internal mechanisms that realize functionality and quality attributes.

Note 2 to entry: It is important to distinguish between the concepts of system and software variability and product line variability. Any system partially or fully composed of software can be considered to possess software variability because software systems are inherently malleable, extendable or configurable for specific use contexts. Product line variability is concerned with the variability that is explicitly defined by product management. This document is primarily concerned with product line variability.

EXAMPLE 1 In the case of a home automation system, in accordance with business strategy, the use of a LAN as an alternative to the European Installation Bus (EIB) in a home automation system might be a competitive advantage for the company since it allows the use of low-cost components.

EXAMPLE 2 Annex B provides variability examples in accordance with variability types.

3.14
variability mechanism
variability representation/implementation technique for the product line variability

Note 1 to entry: It deals with variabilities based on the *binding time* (3.4) at the specific life cycle stage.

3.15
variability mechanism operationalization
VMO
adequate provision or *binding* (3.3) of *variability mechanisms* (3.14) at each specific domain or application engineering life cycle stage

3.16
variability model
explicit definition for product line variability

Note 1 to entry: It introduces *variation points* (3.19), types of variation for the variation points, *variants* (3.17) offered by the variation points, variability dependencies and variability constraints. Variability models may be orthogonal to or integrated in other models such as requirements or design models. There are two types of variability models: application variability models and domain variability models.

3.17
variant
option or an alternative that may be used to realize particular *variation points* (3.19)

Note 1 to entry: One or more variants should correspond to each variation point to represent V_VP relationship. Selection and *binding* (3.3) of variants for a specific product determine the characteristics of the particular *variability* (3.13) for the product.

3.18
variant selection
decision making for a choice of a *variant* (3.17) in a *variation point* (3.19)

Note 1 to entry: *binding* (3.3), variability resolution.

3.19
variation point
representation corresponding to particular variable characteristics of products, domain assets and application assets in the context of a product line

Note 1 to entry: Variation points show which of the product line element varies. Each variation point can have multiple V_VP relationships.

4 Variability mechanisms for software and systems product line (SSPL)

4.1 Overview

Variability mechanism means a method for implementing the variability of a product line and it incorporates variability into the product line development artefacts. Because variability is introduced at the whole product line life cycle stages and so binding does, variability mechanisms for implementing variability in accordance with its binding time should be provided. It is necessary to classify variability mechanisms by binding times so the users of this document can choose proper variability mechanisms in accordance with the binding time decisions. This subclause provides a set of variability mechanisms used at given binding times.

Bindings can occur at requirements stage. In the case of requirements, binding decisions for whether external requirements variabilities exist or not, are made. Variability mechanism operationalization for requirements differ in accordance with requirements artefacts. Variability mechanism operationalization for requirements can be summarized as follows:

— SSPL specifically defined mechanism: feature variability notation;

— language extension mechanism: stereotype in activity diagram, sequence diagram and state machine diagram, coloured notations in state machine diagram;

— language supported mechanism: extends and includes in use case model, swim lane and notes in activity diagram, notes in textual specification and sequence diagram, tags or markups in textural use case scenario.

During architecture design, bindings also occur. In the case of design time, binding elements that compose architectural structure such as components, ports and connectors are selectable. In accordance with the variant selection, architectural structure differs. Components and interfaces are also substituted with another. Variability mechanism operationalization for design can be summarized as follows:

— SSPL specifically defined mechanism: plug-ins in component framework diagram, composite structure diagram, package diagram and deployment diagram, architecture reorganization in process table;

— language extension mechanism: stereotype in component diagram, class diagram, E-R diagram and communication diagram;

— language supported mechanism: notes or tagged values in component diagram, frameworks, pre- and post-conditions in Object Constraint Language (OCL).

At realization stage, additional variation points can be introduced and bindings can occur, so variability mechanisms supporting variability in realization stage are required. Variability mechanism operationalization for realization can be summarized as follows:

— SSPL specifically defined mechanism: orthogonal feature set in model-driven approaches (e.g. KobrA);

— language extension mechanism: stereotype in entity model, stereotype in detailed design level model;

— language supported mechanism: generalization/specialization (extension) in classes, aggregation (optional) in class, abstract class designed as parameterized class (template) and concrete classes (optional), tagged values in entity model, generics in codes, pointcut and advice concepts in Aspect Oriented Programming (AOP), framework implementation methods (e.g. hotspots and hooks) in framework.

There exists variability binding during compilation and in some cases, all bindings can be delayed after compilation. Most bindings occurred when compilation are difficult to change in subsequent stage. Variability mechanism operationalization at compile time can be summarized as follows:

— SSPL specifically defined mechanism: break (variation point), adapt (include) and select (variants) in XVCL, Markup (vp, insert_before, insert_after, insert) and highlighted variant elements code in frame technology;

— language extension mechanism: none;

— language supported mechanism: macro, #ifdef and directives mark in codes for conditional compilation, parameters in makefile.

After compilation, bindings can occur for generating different linked configurations. Such bindings at compile time and at link time are also difficult to change in subsequent stage. Variability mechanism operationalization at link time can be summarized as follows:

— SSPL specifically defined mechanism: configuration space including rules and constraints for different bindings at link time;

— language extension mechanism: none;

— language supported mechanism: configuration files or linker directives for linking static libraries, parameters in makefiles.

Bindings also occur for loading different executable files. Variability mechanism operationalization at load time can be summarized as follows:

— SSPL specifically defined mechanism: rules and constraints description for bindings at load time;

— language extension mechanism: none;

— language supported mechanism: import in source code or external makefile for load time dynamic linking.

For supporting different installation in accordance with the customer preferences or system environments, bindings occur at deployment/installation time. Variability mechanisms at deployment/installation time can be summarized as follows:

— SSPL specifically defined mechanism: rules and constraints description for bindings at deployment/installation time;

— language extension mechanism: none;

— language supported mechanism: options for conditional installation.

Some variabilities are bound at run time. In the case of highly reconfigurable systems, bindings occur only at run time. In normal case, bindings occur at run time for reflecting the running conditions. Variability that is bound at run time tends to be easily rebound or unbound in accordance with the user selections or context conditions. Unlike binding at compile or link time, bindings at run time can easily be changed. Variability mechanism operationalization at run time can be summarized as follows:

— SSPL specifically defined mechanism: rules and constraints description for bindings at run time;

— language extension mechanism: none;

— language supported mechanism: dynamic libraries.

Variabilities should be encoded into test artefacts for addressing test variations among member products. Bindings at testing stage are closely related to the bindings of member product's development stage. Variabilities can also be added for achieving test variations among member products. Variability mechanism operationalization for test artefacts can be summarized as follows:

— SSPL specifically defined mechanism: segmentation and fragmentation mechanisms with notes for test case scenario in sequence diagram;

— language extension mechanism: decision point and its branches for test model in extended activity diagram;

— language supported mechanism: note for test plan in natural language.

NOTE 1 Variability mechanisms for the selected software development activities are depicted in Annex A.

NOTE 2 Binding time suggestions in accordance with variability types and their exemplar variability are depicted in Annex B.

4.2 Reference model for variability mechanisms in product line

The methods and tools for variability mechanisms for software and systems product line should support systematic implementation and management of variability mechanisms in accordance with the determined binding times. They also need to adequately be used in domain engineering life cycle processes in order to enable right bindings in application engineering life cycle processes.

The reference model specifies the structure of supporting processes and subprocesses for variability mechanisms in product line. As shown in Figure 1, variability mechanism in product line can be structured into three processes: variability mechanism management, variability mechanism operationalization and variability mechanism support. In the rest of this document, tasks, methods and tools are described in terms of processes and subprocesses defined in the reference model.

Each process is divided into subprocesses and each subprocess is described in terms of the following attributes:

— the title of the subprocess;

— the purpose of the subprocess;

— the inputs to produce the outcomes;

— the tasks to achieve the outcomes;

— the outcomes of the subprocess;

— the capabilities of methods and tools required for performing the tasks effectively and efficiently.

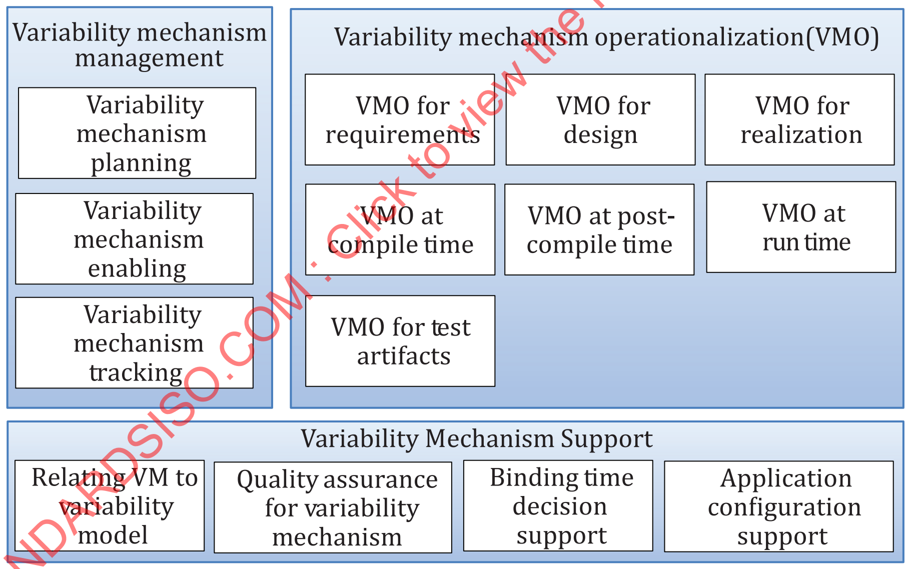
Variability mechanism management
Variability mechanism planning
Variability mechanism enabling
Variability mechanism tracking
Variability mechanism operationalization(VMO)
VMO for requirements
VMO for design
VMO for realization
VMO at compile time
VMO at post-compile time
VMO at run time
VMO for test artifacts
Variability Mechanism Support
Relating VM to variability model
Quality assurance for variability mechanism
Binding time decision support
Application configuration support

Figure 1 — Variability mechanisms for SSPL

Variability mechanism management shall serve to do the following and to define the capabilities of tools and methods for supporting them.

— *Variability mechanism planning* establishes an organization level plan to utilize variability mechanisms.

— *Variability mechanism enabling* defines, maintains and assures the availability of strategies for variability mechanism operationalization, guidance for variability mechanism selection and enabling supports such as human skills and tools for variability mechanism operationalization.

— *Variability mechanism tracking* provides integrated management for variability mechanism operationalization in both domain engineering and application engineering.

Variability mechanism operationalization shall serve to do the following and to define the capabilities of tools and methods for supporting them.

— *Variability mechanism operationalization for requirements* provides variability mechanisms used for dealing with the variabilities that will be bound at requirements stage.

— *Variability mechanism operationalization for design* provides variability mechanisms used for dealing with the variabilities at architecture design stage.

— *Variability mechanism operationalization for realization* provides variability mechanisms used for dealing with the variabilities that will be bound at detailed design and implementation stage;

— *Variability mechanism in compilation* provides variability mechanisms used for dealing with the variabilities that will be bound at pre-compile and compile time.

— *Variability mechanism operationalization at post-compile time* provides variability mechanisms used for dealing with the variabilities defined for different sequence of linkages, for loading different executable files and for installing or deploying different configurations.

— *Variability mechanism operationalization at run time* provides variability mechanisms used for dealing with the variabilities defined for accessing different files (or components, interfaces) at run time.

— *Variability mechanism operationalization for test artefacts* provides variability mechanisms used for dealing with variabilities that should be realized in test assets.

Variability mechanism support shall serve to do the following and to define the capabilities of tools and methods for supporting them.

— *Relating variability mechanism to variability model* establishes and maintains information (e.g. binding time, rules and constraints adhered) necessary to conduct right binding and right configuration by using right variability mechanisms.

— *Quality assurance for variability mechanism* helps ensure whether the artefacts and processes of variability mechanism management and operation comply with predefined provisions and plans.

— *Binding time decision support* deals with the required provisions necessary to conduct the right binding at the planned binding stages (e.g. architecture rules to constrain the use of macro or makefile parameters).

— *Application configuration support* deals with the realization of configurability by determining ways to derive a member product such as compiler parameters.

The identification and analysis of the key differentiators between single-system engineering and management and product line engineering and management can help organizations to understand the product line and to formulate a strategy for successful implementation of product line engineering and management. The key aspects have been defined in ISO/IEC 26550 and Table 1 shows the category of the key aspects.

Table 1 — Key aspects for identifying product line-specific variability mechanism tasks

Category	Aspects
Reuse management	application engineering, domain assets, domain engineering, product management, platform, reusability
Variability management	binding, variability
Complexity management	collaboration, configuration, enabling technology support, reference architecture, texture, traceability
Quality management	measurement and tracking, cross functional verification and validation

The following are the descriptions for each aspect concerning variability mechanisms for product lines. The variability mechanism relevant processes and tasks shall the identified on the basis of these aspects. The concerns specific to variability mechanisms for product lines will enable an organization to understand the variability mechanism relevant processes, subprocesses, tasks, methods and tools' capabilities.

— *Application engineering:* variability mechanism has influences on the possible binding times, so the binding time of variability and other binding-related decisions adhered during application engineering are closely related to variability mechanisms used.

— *Binding:* variability mechanism is the major factor for binding decision. Binding time restricts to variability mechanism that can be used.

— *Collaboration:* variability mechanism should support collaborative development because when variability is spread across the different components due to the restriction of a mechanism. It is difficult to develop such components without collaboration.

— *Configuration:* variability mechanism is an important factor for deriving the configuration of a member product, so variability mechanism should be determined toward ensuring the configurability of member products.

— *Domain asset:* variability mechanisms differ in accordance with the types of domain assets. In the case of using plug-in mechanism. The number of component and interface assets can increase than in the case of including variability within components and interfaces.

— *Domain engineering:* variability mechanism should be classified by domain engineering stages because variability can be introduced throughout domain engineering stages.

— *Enabling technology support:* methods and tools required for enabling variability mechanism uses and management in requirements modelling, design, realization, compiling, linking, loading and running stages should be provided.

— *Measurement and tracking:* trade-off analysis for variability selection, measurements for the flexibility and reusability of a variability mechanism should be conducted, and status data from variability mechanism used until when the binding occurs should be traced and measured.

— *Platform:* variation points, where variants will be bound are implemented in a platform and they differ in accordance with variability mechanisms used for realizing variability.

— *Product management:* variability mechanism should support easy variability evolution in accordance with product line evolution.

— *Reference architecture:* reference architecture should be defined toward supporting the easy use of possible variability mechanisms in domain design.

— *Reusability:* variability mechanisms should support the reusability of domain assets.

— *Texture:* texture can define binding rules for the used variability mechanisms that should be adhered by a member product (e.g. rules for makefile parameter use).

— *Traceability:* because variability mechanism is a way to deal with variability. it should support easy traceability among domain asset, application asset and variability model.

— *Cross functional validation and verification:* variability mechanisms that realize variabilities introduced in testing stage should be classified and supported.

— *Variability:* variability mechanism provides means to realize variabilities defined and managed in a product line.

5 Variability mechanism management

Variability mechanisms in the pool should be used in consistent and controlled ways for supporting the right operation of variability mechanisms and right bindings at application engineering stages. The use of variability mechanisms should be planned and enabled and its status should be traced for the management and future improvement.

Variability mechanism management supports the following:

— variability mechanism planning;

— variability mechanism enabling;

— variability mechanism tracking.

5.1 Variability mechanism planning

5.1.1 Purpose of variability mechanism planning

The purpose of this subprocess is to produce plans for variability mechanism operationalization. For producing plans and strategies for variability mechanism operationalization, required resources such as skills and tools should be defined and evaluated in advance for establishing proper plans and strategies.

5.1.1.1 Inputs

— Organizational transition plans (from ISO/IEC 26556).

— Organizational level product line evolution plans (from ISO/IEC 26556).

— Asset proposals (from ISO/IEC 26551).

— A set of variability together with their detailed information by domain engineering stages.

5.1.1.2 Outcomes

— *Resources necessary to operate variability mechanism are defined*.

— *Assignment of responsibilities is clarified*.

— *Quality assurance measures are employed throughout the SSPL stages*.

— *Plans for variability mechanism operationalization are developed*.

5.1.1.3 Tasks

— *Estimate adequate resources needed for variability mechanism operationalization* is to estimate knowledge, skills and tools for variability mechanism use, variability mechanism pool maintenance and performing bindings.

— *Assign responsibility for variability mechanism operationalization* is to clarify roles and responsibilities related to variability mechanism operationalization.

— *Define quality assurance measures in variability mechanism operationalization* is to employ quality assurance in variability mechanism operationalization so as to achieve quality assurance throughout the SSPL stages.

5.1.2 Estimate adequate resources needed for variability mechanism operationalization

The goal of this task is to estimate knowledge, skills, engineers and infrastructures for variability mechanism operationalization and management.

The method should support estimating adequate resources needed for variability mechanism operationalization with the following capabilities:

— defining required knowledge and skills (e.g. modelling languages, programming languages) for variability mechanism operationalization and management;

— defining detailed tools and methodologies (e.g. modelling tools supporting variability mechanisms) related to variability mechanism operationalization;

— providing a way to establish and maintain variability mechanism pool (catalog);

— defining plans for resource allocation required for variability mechanism operationalization.

A tool should support estimating adequate resources needed for variability mechanism operationalization by allowing the user to do the following:

— documenting resources required for the right operation of variability mechanisms;

— providing tool capabilities for establishing and maintaining variability mechanism pool (catalogue);

— sharing resource allocation plans with the relevant participants.

5.1.3 Assign responsibility for variability mechanism operationalization

The goal of this task is to assign responsibility for the efficient and effective operation of variability mechanisms.

The method should support assigning responsibility for variability mechanism operationalization with the following capabilities:

— defining roles and responsibilities for variability mechanism operationalization;

— harmonizing required resources across roles and responsibilities for variability mechanism operationalization;

— allocating resources to each role and responsibility for proper variability mechanism operationalization;

— establishing commitments to provide the resources.

A tool should support assigning responsibility for variability mechanism operationalization by allowing the user to do the following:

— recording roles, responsibilities and their allocated resources;

— notifying roles and responsibilities to participants concerned.

5.1.4 Defining quality assurance measures for variability mechanism operationalization

The goal of this task is to define measures for evaluating qualities for variability mechanism operationalization and to assure whether the selection and use of variability mechanisms is performed in accordance with the defined responsibilities, rules, constraints and procedures.

The method should support defining quality assurance measures for variability mechanism operationalization with the following capabilities:

— defining metrics used for evaluating the operability (e.g. usability, understandability, verifiability) of variability mechanism;

— defining measures for measuring the operability;

- establishing ways for tracing the operation for assuring whether the selection and use of variability mechanisms is performed in accordance with the defined responsibilities, rules, constraints and procedures;
- defining ways to share and receive feedbacks for variability mechanism operationalization.

A tool should support defining quality assurance measures for variability mechanism operationalization by allowing the user to do the following:

- supporting measurement data collection by tracing variability mechanism operationalization;
- supporting the maintenance of the defined quality assurance measures;
- allowing the easy access of the defined measures;
- editing and storing quality assurance results.

5.2 Variability mechanism enabling

5.2.1 Purpose of variability mechanism enabling

The purpose of this subprocess is to establish and maintain enabling infrastructures necessary to execute variability mechanism plans. Variability mechanism enabling subprocess confirms the availability of strategies, guidance for variability mechanism selection, procedures, required skills and tools for variability mechanism operationalization.

5.2.1.1 Inputs

- Guidance for variability mechanism selection.
- Estimated resources necessary to variability mechanism operationalization.
- Measures for variability mechanism plans.

5.2.1.2 Outcomes

- *A set of variability mechanisms* is maintained and improved.
- *Guidance for variability mechanism selection* is maintained and improved.
- *Procedures necessary to operate variability mechanism* are defined and improved.
- *Measures for monitoring variability mechanism usage* are defined and improved.
- *Requirements for enabling infrastructure for variability mechanism operationalization* such as hardware, software, methods, tools, techniques and facilities are defined.
- *Enabling infrastructure for variability mechanism operationalization* is maintained and improved.

5.2.1.3 Tasks

- *Enable variability mechanism pool* to provide, maintain and improve a set of variability mechanisms in organization level so that domain engineer can use them with guidance. Variability mechanism pool includes guidance for using variability mechanisms.
- *Provide guidance for variability mechanism operationalization* to define and maintain guidance for variability mechanism selection and configuration in accordance with binding time decision and variability characteristics.
- *Enable measurement infrastructure for quantifying variability mechanism operationalization* to establish enabling infrastructures for measuring the variability mechanism operationalization.

— *Procure resources needed to perform variability mechanism operationalization* to provide knowledge and skills required for using variability mechanisms, maintaining the forms of variability mechanisms in the relevant artefacts and performing bindings.

5.2.2 Enable variability mechanism pool

The goal of this task is to establish, initiate and maintain variability mechanism pool organized by domain life cycle stages.

The method should support enabling variability mechanism pool with the following capabilities:

— developing common rules used for applying variability mechanisms for developing domain assets (reference architecture, components, etc.);

— designing the structure of variability mechanism pool that enables easy reference;

— designing the ways to access variability mechanism pool;

— providing exemplar usages of variability mechanisms for supporting the right use.

A tool should support enabling variability mechanism pool by allowing the user to do the following:

— implementing the defined structure of variability mechanism pool (catalogue);

— enabling the access of variability mechanism pool by means of the designed access method;

— allowing access for the exemplar usages of variability mechanisms (e.g. hyperlink).

5.2.3 Provide guidance for variability mechanism operationalization

The goal of this task is to produce guides, procedures and rules supporting the right variability mechanism selection and use by the decided binding time and the characteristics of variability.

The method should support providing guidance for variability mechanism operationalization with the following capabilities:

— defining variability mechanism selection criteria by domain engineering stages;

— providing detailed procedures for variability mechanism selection and use;

— defining configuration guides and rules.

A tool should support providing guidance for variability mechanism operationalization by allowing the user to do the following:

— sharing guides and rules for variability mechanism operationalization with relevant domain/application engineers;

— allowing instant access for guides and rules for performing variability mechanism operationalization.

5.2.4 Enable measurement infrastructure for quantifying variability mechanism operationalization

The goal of this task is to establish and maintain enabling infrastructure for the quantitative measurement of variability mechanism operationalization.

The method should support enabling measurement infrastructure for quantifying variability mechanism operationalization with the following capabilities:

— integrating measurement environment to development stage using variability mechanisms;

— providing ways for integrating data collected from variability mechanism operationalization;

— providing ways to find and review obstacles to the successful use of variability mechanism;

— providing documentation templates for recoding the quality of variability mechanism operationalization.

A tool should support enabling measurement infrastructure for quantifying variability mechanism operationalization by allowing the user to do the following:

— initiating measurement tool environment with variability mechanism operationalization environments;

— allowing tool supported measurement data collection;

— allowing documentation for the quality of variability mechanism operationalization.

5.2.5 Procure resources needed to perform variability mechanism operationalization

The goal of this task is to supply knowledge, skills and tools (e.g. tools supporting the selected variability mechanisms) required to incorporate variability mechanisms into domain assets.

The method should support procuring resources needed to perform variability mechanism operationalization with the following capabilities:

— assuring availability of required resources;

— providing ways to assess the alterative resources;

— monitoring and controlling the fulfilment degree of procured resources.

A tool should support procuring resources needed to perform variability mechanism operationalization by allowing the user to do the following:

— providing managerial support for the procured resources and their status;

— allowing feedback collection for monitoring and controlling the fulfilment level of procured resources.

5.3 Variability mechanism tracking

5.3.1 Purpose of variability mechanism tracking

The purpose of this subprocess is to monitor and control variability mechanism operationalization and the availability of variability mechanism supports in both domain engineering and application engineering, so as to provide proper managerial supports.

5.3.1.1 Inputs

— Measures for monitoring variability mechanism usage.

— Status of variability mechanism operationalization.

— Variability mechanism plans.

5.3.1.2 Outcomes

— *Plans versus actuals are reviewed.*

— *Measurements for monitoring variability mechanism usage* are performed.

— *Corrective actions are performed and the list including their status* is maintained.

— *Inputs for improvement are generated.*

5.3.1.3 Tasks

— *Review the plan versus actual of variability mechanism operationalization* is to assign the quality objectives achieved through variability mechanism operationalization with their pre-defined measures and metrics and monitor whether they are operated in line with the pre-defined plans.

— *Assess issues in variability mechanism operationalization* is to examine issues raised during variability mechanism operationalization so as to find obstacles for achieving quality objectives and ways to remove the obstacles.

— *Make corrective actions for variability mechanism operationalization* is to improve variability mechanism pools and operation to achieve the assigned quality objectives.

5.3.2 Review the plan versus actual of variability mechanism operationalization

The goal of this task is to check the quality of variability mechanism operationalization.

The method should support reviewing the plan versus actual of variability mechanism operationalization with the following capabilities:

— defining monitoring procedures for variability mechanism operationalization status compared to plans;

— providing ways to collect data for comparing plan versus actual operations;

— defining ways to integrate data for judging the status.

A tool should support reviewing the plan versus actual of variability mechanism operationalization by allowing the user to do the following:

— sharing consensus on the defined roles and responsibilities;

— providing (semi-)automated measurement environment for data collection and integration.

5.3.3 Assess issues in variability mechanism operationalization

The goal of this task is to generate and share issues for improving variability mechanism operationalization.

The method should support assessing issues in variability mechanism operationalization with the following capabilities:

— providing ways to find and review obstacles to the successful operation of variability mechanism;

— defining decision criteria for classifying issues that require corrective actions;

— providing ways to defining corrective actions based on the type of obstacles reviewed in “review issues of variability mechanism operationalization” task;

— providing documentation templates for recoding assessment results and corrective action plans.

A tool should support assessing issues in variability mechanism operationalization by allowing the user to do the following:

— supporting issues collection raised by different roles and responsibility;

— allowing documentation for assessment results and corrective action plans.

5.3.4 Make corrective actions for variability mechanism operationalization

The goal of this task is to execute corrective actions for variability mechanism operationalization to achieve the defined quality level.

The method should support making corrective action for variability mechanism operationalization with the following capabilities:

— providing ways to monitor and control the status of correction action;

— communicating corrective action results with the relevant participants;

— collecting the improvement items of variability mechanism operationalization;

— implementing the improvement items of variability mechanism operationalization.

A tool should support making corrective action for variability mechanism operationalization by allowing the user to do the following:

— supporting simulation for confirming the achievement of quality level related to the variability mechanism operationalization;

— allowing traces for the status of corrective actions;

— supporting improvement input collection;

— providing communication environment among the relevant participants.

6 Variability mechanism operationalization

Domain engineers should be informed what kinds of variability mechanisms are possible at the specific domain engineering stages and how they use the variability mechanisms in the right way. For utilizing variability, mechanisms at each stage domain engineers should examine the characteristics of variability such as dependency and constraint relationships and stage-specific rules and constraints, and variability mechanisms should be assessed and selected in accordance with the guidance. Furthermore, preparations for supporting the right bindings at application engineering stages should be conducted.

Variability mechanism operationalization supports the following:

— variability mechanism operationalization for requirements;

— variability mechanism operationalization for design;

— variability mechanism operationalization for realization;

— variability mechanism operationalization at compile time;

— variability mechanism operationalization at post-compile time;

— variability mechanism operationalization at run time;

— variability mechanism operationalization for testing.

6.1 Variability mechanism operationalization for requirements

6.1.1 Purpose of variability mechanism operationalization for requirements

The purpose of this subprocess is to support the use of variability mechanisms related to variabilities that will be resolved in requirements stage.

6.1.1.1 Inputs

— Variability introduced in requirements.

— Variability that will be bound at requirements stage.

— Variability model in requirements.

6.1.1.2 Outcomes

— *External variabilities* that will be bound at requirements stage is identified.

— *Variability mechanism operationalization for requirements* is assessed, selected and specified.

— *Description for requirements level variability mechanisms* is provided.

— *Configurability in requirements* is enabled.

— *Requirements level variability mechanism* is verified.

6.1.1.3 Tasks

— *Categorize requirements variability* is to classify requirements variabilities, those bound at requirements stage or after requirements stage. This allows a member product to do the right bindings at requirements stage by providing well-categorized requirements variability in the forms of domains and subdomains by functional and non-functional requirements.

— *Assess requirements level variability mechanism* is to explore, analyse and deploy the proper variability mechanism for implementing external and/or derived requirements variability.

— *Specify requirements level variability mechanism* is to describe variability mechanisms in domain requirements artefacts such as requirements analysis models and requirements specification (e.g. "extends" and "includes" in use case diagram).

— *Prepare bindings at requirements level* is to provide the basis of selecting variants consistently for a member product such that which of variation points are supposed to be bound at requirements level.

— *Verify requirements level variability mechanism* is to confirm that variability mechanism operationalization for requirements stage adhere the deploy criteria of external variability existence, applied rule, binding rules and constraints.

6.1.2 Categorize requirements variability

The goal of this task is to identify and categorize variabilities for deciding the right variability mechanisms and for guiding the right binding in requirements analysis models and requirements specifications. This task deals with only requirements variability whose binding time is requirements stage.

The method should support categorizing requirements variability with the following capabilities:

— identifying the dependency and constraint relationship of a variability to be considered in terms of determining variability mechanisms;

— classifying external and internal (including derived) requirements variability resolved in requirements stage for the right decisions of variability mechanisms;

— classifying non-functional (including derived) requirements variability resolved in requirements stage;

— categorizing requirements variability into domain and subdomain by functional and non-functional requirements in terms of their distinct characteristics (constraints to platform, process, technology or design).

A tool should support categorizing requirements variability by allowing the user to do the following:

— storing the classification results of requirements variability;

— importing the required requirements artefacts or their information for deciding variability mechanisms used;

— editing requirements variability category;

— editing the classification of requirements variability by the defined category;

— displaying the categorized requirements variability.

6.1.3 Assess requirements level variability mechanism

The goal of this task is to assess and select a variability mechanism for implementing variability that will be bound at requirements stage.

The method should support assessing requirements level variability mechanism with the following capabilities:

— adapting guidance for variability mechanism selection to requirements stage;

— providing guides for variability mechanism selection for non-functional requirements;

— performing detailed analysis for the requirements level variability mechanisms based on the guidance for variability mechanism selection;

— providing rationale and assumptions (if necessary) related to the decisions;

— defining templates used for assessment (e.g. assessment result template).

A tool should support assessing requirements level variability mechanism by allowing the user to do the following:

— storing the defined guidance for variability mechanism selection specific to requirements stage;

— supporting assessment progress by guiding assessment steps;

— allowing the user to access required information necessary for assessment;

— providing templates for documentation;

— storing rationale and assumptions for decisions.

6.1.4 Specify requirements level variability mechanism

The goal of this task is to provide ways to specify variability mechanism operationalization for requirements level artefacts.

The method should support specifying requirements level variability mechanism with the following capabilities:

— providing notations for specifying requirements level variability mechanism operationalization for requirements artefacts (e.g. requirements analysis models, requirements specification);

— providing ways to specifying variability mechanisms for non-functional requirements;

— allowing easy requirements bindings (e.g. by allowing variant selection in variability model carrying over bindings in requirements artefacts);

— allowing mapping to variability model from requirements artefact, using a variability mechanism.

A tool should support specifying requirements level variability mechanism by allowing the user to do the following:

— describing variability mechanisms including a variation point and its variants in domain requirements models;

— specifying variability mechanisms for non-functional requirements;

— providing ways to replace parts of requirements models with bound variants;

— linking variability mechanisms used in requirements model with the relevant parts of variability model so that the product specific requirement models can be derived automatically in accordance with bindings.

6.1.5 Prepare bindings at requirements level

The goal of this task is to support preparation for bindings at application requirements engineering stage from variability mechanism views.

The method should support preparing bindings at requirements level with the following capabilities:

— preparing requirements variability bindings by providing ways for validating the binding results in a variability mechanism dimension;

— preparing the convert of a variation point within requirements model, specification or document to a bound variant in a variability mechanism dimension.

A tool should support preparing bindings at requirements level by allowing the user to do the following:

— implementing the ways for validating the binding results in a variability mechanism dimension;

— enabling the transformation of requirements configuration based on the variability binding results.

6.1.6 Verify requirements level variability mechanism

The goal of this task is to verify requirements level variability mechanisms in their selection, specification and utilization.

The method should support verifying requirements level variability mechanism with the following capabilities:

— confirming requirements variability mechanisms whether they can correctly implement the defined variability dependencies;

— confirming requirements variability mechanisms whether they can correctly implement the defined constraints;

— defining requirements model variability mechanism evaluation algorithm;

— verifying the configurability of variability mechanisms used in requirements level;

— verifying whether variability mechanisms used are appropriate to proceed architecture design;

— verifying the testability of variability mechanisms used in requirements level.

A tool should support verifying requirements level variability mechanism by allowing the user to do the following:

— making requirements artefacts including variability mechanisms available;

— supporting verification whether the variability mechanism properly supports the defined variability including dependencies and constraints;

— implementing evaluation algorithm;

— supporting the verification of the testability of requirements based on used variability mechanisms.

6.2 Variability mechanism operationalization for design

6.2.1 Purpose of variability mechanisms in domain design

The purpose of this subprocess is to support variability mechanisms related to variabilities that will be resolved in architecture design stage.

6.2.1.1 Inputs

— Variability introduced in design.

— Variability bound at design stage.

— Variability model in design.

6.2.1.2 Outcomes

— *Architectural decisions on binging times are determined* and managed.

— *Variability mechanisms in architectural structure are assessed* and specified.

— *Guides and rules on variability mechanisms in architectural texture* is provided.

— *Architectural structure level bindings* are performed.

— *Configurability in architectural structure* is enabled.

— *Architectural structure level variability mechanism* is verified.

6.2.1.3 Tasks

— *Make architectural decisions on binding times* is to determine architectural decisions for the structural elements and texture of reference architecture affected by binding time decisions.

— *Assess variability mechanisms depending on the binding time* is to analyse variability mechanisms that realize variation points resolved at the determined binding time and deploy proper variability mechanisms. The selection of variability mechanisms affects to easy configuration.

— *Define guides and rules on variability mechanisms in architectural texture* is to provide guides and rules adhered by the subsequent stages for the right construction of variability mechanisms and for the right bindings at the member product derivation.

— *Specify architectural variability mechanisms* is to describe variability mechanisms in architectural artefacts (e.g. stereotype and tagged values in UML component diagram).

— *Prepare bindings at architecture level* is to prepare bindings that a member product selects variants consistently for which variation points are supposed to be bound at design level (e.g. right component selection or declaring parameters for selecting right component variants).

— *Verify architectural variability mechanisms* is to analyse how well the defined variability mechanism operationalization for design support variability described and enabled in architecture design artefacts.

6.2.2 Make architectural decisions on binding times

The goal of this task is to provide rationales and decisions for binding times at architecture level.

The method should support making architectural decisions on binding times with the following capabilities:

— providing restrictions used for variability mechanism related decisions at architecture level;

— providing materials, architecturally important domain characteristics and available variability mechanisms used for binding time related decisions;

— allowing architecture level decisions on binding times with rationales.

A tool should support making architectural decisions on binding times by allowing the user to do the following:

— providing architecture level variability mechanisms pool with restrictions;

— allowing the access of materials, architecturally important domain characteristics and possible variability mechanisms used for binding time related decisions;

— storing architectural level decisions on binding times with rationales.

6.2.3 Assess variability mechanisms depending on the binding time

The goal of this task is to assess and select an optimal variability mechanism that appropriates for binding time and the characteristics of variability in architecture design.

The method should support assessing variability mechanisms depending on the binding time with the following capabilities:

— reviewing the available variability mechanisms whether they comply with the predetermined architectural decisions on binding times;

— assuring whether a variability mechanism fully covers the variability including dependencies and constraints.

A tool should support assessing variability mechanisms depending on the binding time by allowing the user to do the following:

— providing detailed design level variability mechanisms pool with their general usage guidance;

— storing rationales for variability mechanism selection.

6.2.4 Define guides and rules on variability mechanisms in architectural texture

The goal of this task is to provide guides and rules or constraints used to execute variability mechanisms from domain architecture stage through domain realization. Guides and rules govern the structural elements and textural variation points planned to be bound at design time and they can govern most bindings that will occur at later stages (e.g. regulation for macro usage).

The method should support defining guides and rules on variability mechanisms in architectural texture with the following capabilities:

— supporting rules and constraints (e.g. bindings, application specific adaptations) descriptions for variability mechanisms for guiding the further detailed design and realization;

— providing different guidance in accordance with organization's product line maturity;

— providing ways to describe rules and constraints on variability mechanisms in architectural texture.

A tool should support defining guides and rules on variability mechanisms in architectural texture by allowing the user to do the following:

— supporting rules and constraints descriptions for variability mechanisms in architectural texture;

— storing the defined guidance.

6.2.5 Specify architectural variability mechanisms

The goal of this task is to provide ways to specify variability mechanisms in architecture level artefacts.

The method should support specifying architectural variability mechanisms with the following capabilities:

— providing standard notations for dealing with variability mechanisms in architecture level artefacts;

— supporting variability abstraction and expansion at the different layers of the domain architecture;

— supporting the generic representation of variability mechanisms in architecture for achieving consistency and traceability;

— allowing the configuration of member products by providing ways to define boundaries of a variation point in accordance with the used variability mechanism;

— supporting different level (abstraction and expansion) configurations combine;

— providing ways to reason about valid configurations.

A tool should support specifying architectural variability mechanisms by allowing the user to do the following:

— supporting architecture level variability mechanism specification with the standard notations;

— allowing layered domain architecture specification in various variability mechanisms;

— supporting definition of boundaries of a variation point in accordance with the used variability mechanism;

— supporting representation of boundaries of a variation point in accordance with the used variability mechanism;

— allowing variability mechanisms available in domain-specific editors for proceeding configuration.

6.2.6 Prepare bindings at architecture level

The goal of this task is to support preparation for bindings at application architecture design stage from variability mechanism views.

The method should support preparing bindings at architecture level with the following capabilities:

— preparing variability bindings according to variability mechanisms in architecture;

— preparing the convert of a variation point within architecture design model, specification or document to a bound variant according to the used architecture level variability mechanisms;

— selecting the variants in architecture structure complying with variability mechanisms;

— converting variation point in architecture into the concrete architecture structure.

A tool should support preparing bindings at architecture level by allowing the user to do the following:

— storing and displaying a variation point with its variants according to variability mechanism used in architecture design stage for supporting proper binding;

— notifying when bindings in architecture are not properly conducted.

6.2.7 Verify architectural variability mechanisms

The goal of this task is to verify architectural structure level variability mechanisms in their selection, specification and utilization.

The method should support verifying architectural variability mechanisms with the following capabilities:

— confirming variability mechanisms in architecture whether they can correctly implement the defined variability dependencies;

— confirming variability mechanisms in architecture whether they can correctly implement the defined constraints;

— providing model evaluation algorithm for verifying whether a variability mechanism conforms to the defined variability dependencies and constraints;

— verifying the configurability of variability mechanisms used;

— verifying the testability of variability mechanisms used.

A tool should support verifying architectural variability mechanisms by allowing the user to do the following:

— making architecture design artefacts including variability mechanisms available;

— supporting the verification whether the variability mechanisms in detailed design artefacts properly support the defined variability including its dependencies and constraints;

— implementing model evaluation algorithm;

— supporting the testability verification of variability mechanisms used in architecture design.

6.3 Variability mechanism operationalization for realization

6.3.1 Purpose of variability mechanisms in domain realization

The purpose of this subprocess is to support variability mechanisms related to variabilities that will be resolved in realization stage.

6.3.1.1 Inputs

— Variability introduced in realization.

— Domain variability model in realization.

— Variability that will be bound at realization stage.

6.3.1.2 Outcomes

— *Architectural decisions and architectural texture on realization time* are extracted.

— *Variability mechanisms in detailed design* are assessed and specified.

— *Post-detailed design guides on variability mechanisms* are provided.

— *Detailed design level variability mechanism* is verified.

— *Variability mechanisms in implementation level* are assessed and specified.

— *Realization level bindings* are prepared.

— *Realization level configurability* is enabled.

— *Implementation level variability mechanism* is verified.

6.3.1.3 Tasks

— *Examine architectural decisions and architectural texture on realization* is to find decisions, rules and constraints affecting decisions on variability mechanisms in domain realization time and bindings in application realization time.

— *Assess detailed design level variability mechanisms* is to explore, analyse and deploy proper variability mechanisms for encoding variation points to detailed design.

— *Specify detailed design level variability mechanisms* is to describe variability mechanisms in detailed design artefacts (e.g. stereotype or tagged values in class diagram and entity model).

— *Define post-detailed design guides on variability mechanisms* is to allow the right implementation and bindings of variability realized by variability mechanisms. A set of run time choices for system reconfiguration during system execution can be defined.

— *Verify detailed design level variability mechanisms* is to analyse how well the defined variability mechanism operationalization for design support variability described and enabled in architecture design artefacts.

— *Assess implementation level variability mechanisms* is to explore, analyse and deploy proper variability mechanisms for constructing variation points in the forms of tangible or executable implementation (e.g. modules or codes).

— *Specify implementation level variability mechanisms* is to describe variability mechanism in implementation artefacts (e.g. generics in codes).

— *Enable realization time configurability* is to prepare for the easy configuration of implementations for member products.

— *Prepare bindings at realization time* is to prepare bindings for a member product to select variants consistently for which variation points are supposed to be bound at realization level.

— *Verify implementation level variability mechanisms* is to analyse how well the defined variability mechanisms implemented support variability described and enabled in detailed design artefacts.

6.3.2 Examine architectural decisions and architectural texture on realization

The goal of this task is to examine the predetermined architectural decisions and textures adhered for selecting or using variability mechanisms in domain realization.

The method should support examining architectural decisions and architectural texture on realization with the following capabilities:

— exploring detailed design and implementation options among architectural decisions and textures;

— allowing the examination of restrictions on variability in realization stage.

A tool should support examining architectural decisions and architectural texture on realization by allowing the user to do the following:

— providing the discrimination of realization stage relevant architectural decisions and textures among others;

— storing restrictions on variability in realization stage.

6.3.3 Assess detailed design level variability mechanisms

The goal of this task is to assess and select an optimal variability mechanism that appropriates for binding time and the characteristics of variability in detailed design.

The method should support assessing detailed design level variability mechanisms with the following capabilities:

— reviewing the available variability mechanisms whether they comply with the explored architectural decisions and textures in realization time;

— assuring whether a variability mechanism fully covers the defined variability including dependencies and constraints.

A tool should support assessing detailed design level variability mechanisms by allowing the user to do the following:

— providing detailed design level variability mechanisms pool with their usage guidance;

— storing rationales for variability mechanism selection.

6.3.4 Specify detailed design level variability mechanisms

The goal of this task is to provide ways to specify variability mechanisms in detailed design artefacts.

The method should support specifying detailed design level variability mechanisms with the following capabilities:

— providing standard notations for dealing with variability mechanisms in detailed design artefacts (i.e. different kinds of models that depict static and behavioural aspects of a product line);

— providing ways to abstract and expand variability mechanism specification at the different views of detailed design artefacts;

— supporting fields definition for a parameter table (e.g. the name of parameter, parameter type, allowed values);

— allowing cooperation with configuration activity for providing suitable configuration parameters;

— supporting the generic representation of variability mechanisms in detailed design for achieving consistency and traceability.

A tool should support specifying detailed design level variability mechanisms by allowing the user to do the following:

— supporting variability mechanism specification in detailed design artefacts with the standard notations;

— allowing abstraction and expansion at the variation points of detailed design artefacts using variability mechanism in detailed design.

6.3.5 Define post-detailed design guides on variability mechanisms

The goal of this task is to provide guides on variability mechanisms that lead post-detailed design activities.

Post-detailed design guides on variability mechanisms include constraints and rules on activities occurred after detailed design related to variability mechanisms used in detailed design (e.g. implementation guides on variation point and is variants designed using inheritance and stereotyped variability mechanisms),

The method should support defining post-detailed design guides on variability mechanisms with the following capabilities:

— determining details of constraints and rules on adhered at post detailed design stages;

— preparing parameter list for components using parameterization including relevancies with variability;

— providing ways to trace the defined constraints and rules with the relevant points using variability mechanism.

A tool should support defining post-detailed design guides on variability mechanisms by allowing the user to do the following:

— storing constraints and rules on variability mechanisms so as to be referred by relevant participants;

— supporting the maintenance and usage of parameter list;

— supporting traceability between the defined constraints/rules and the relevant points using variability mechanism.

6.3.6 Verify detailed design level variability mechanisms

The goal of this task is to verify detailed design level variability mechanisms in their selection, specification and utilization.

The method should support verifying detailed design level variability mechanisms with the following capabilities:

— confirming variability mechanisms in detailed design whether they can correctly implement the defined variability dependencies;

— confirming variability mechanisms in detailed design whether they can correctly implement the defined constraints;

— providing evaluation algorithm for verifying whether a variability mechanism conforms to the defined variability dependencies and constraints;

— verifying the configurability of variability mechanisms used in detailed design;

— verifying the testability of variability mechanisms used in detailed design.

A tool should support verifying detailed design level variability mechanisms by allowing the user to do the following:

— making detailed design artefacts including variability mechanisms available;

— supporting the verification of whether the variability mechanism properly supports the defined variability including dependencies and constraints;

— implementing model evaluation algorithm;

— supporting the testability verification of variability mechanisms used in detailed design artefacts.

6.3.7 Assess implementation level variability mechanisms

The goal of this task is to assess and select an optimal variability mechanism that appropriates for implementation binding time and the characteristics of variability in implementation level.

The method should support assessing implementation level variability mechanisms with the following capabilities:

— reviewing the available variability mechanisms whether they comply with the explored architectural decisions and textures in realization time;

— assuring whether a variability mechanism fully covers the defined variability including dependencies and constraints that should be bound in implementation stage;

— evaluating ease of binding and configuration.

A tool should support assessing implementation level variability mechanisms by allowing the user to do the following:

— providing detailed implementation level variability mechanisms pool with their usage guidance for selection;

— storing rationales for variability mechanism selection.

6.3.8 Specify implementation level variability mechanisms

The goal of this task is to provide ways to specify variability mechanisms in implementation artefacts.

The method should support specifying implementation level variability mechanisms with the following capabilities:

— providing ways (syntax and semantic definition) for representing variability mechanisms in implementation artefacts;

— providing ways for differentiating among relevant blocks within implementation artefacts (e.g. coloured block for relevant codes);

— allowing easy of achieving the consistency and traceability of variability between implementation artefacts and variability model.

A tool should support specifying implementation level variability mechanisms by allowing the user to do the following:

— providing variability mechanism representation in implementation artefacts;

— providing explicit representation for differentiating variability mechanism relevant boundaries from others;

— maintaining trace links between the variation points specified using variability mechanism in implementation level and variability model.

6.3.9 Enable implementation level configurability

The goal of this task is to prepare for producing member products' configurations.

The method should support enabling implementation level configurability with the following capabilities:

— incorporating variability mechanisms into domain detailed design (i.e. interface, component) with ways to derive configurations for member products;

— allowing the derivation of the detailed design models (i.e. static and behavioural models) and different aspect configurations for providing integrated view of them;

— preparing configuration parameters for variability mechanisms using parameters;

— providing ways to reason about valid configurations in detailed design level.

A tool should support enabling implementation level configurability by allowing the user to do the following:

— defining boundaries of a variation point in accordance with the used variability mechanism;

— representing boundaries of a variation point in accordance with the used variability mechanism;

— allowing variability mechanisms available in domain-specific editors for proceeding configuration;

— making preparation for code generation at detailed design level based the defined parameter table.

6.3.10 Prepare bindings at realization time

The goal of this task is to support preparation for bindings at application realization stage from variability mechanism views.

The method should support preparing bindings at realization time with the following capabilities:

— selecting the variants in detailed design artefacts complying with variability mechanisms;

— converting variation point in detailed design artefacts into those for a member product.

A tool should support preparing bindings at realization time by allowing the user to do the following:

— storing and displaying a variation point with its variants according to variability mechanism used in detailed design stage for supporting proper binding;

— notifying when bindings at realization are not properly conducted.

6.3.11 Verify implementation level variability mechanisms

The goal of this task is to verify implementation level variability mechanisms in their selection, specification and utilization.

The method should support verifying detailed design level variability mechanisms with the following capabilities:

— confirming variability mechanisms in implementation (e.g. variability mechanisms in codes, component, modules) whether they can correctly implement the defined variability dependencies;

— confirming variability mechanisms in implementation whether they can correctly implement the defined constraints;

— providing evaluation algorithm for verifying whether a variability mechanism conforms to the defined variability dependencies and constraints;

— verifying the testability of variability mechanisms used in implementation.

A tool should support verifying detailed design level variability mechanisms by allowing the user to do the following:

— visualizing variability mechanism included implementation;

— supporting the verification of whether the variability mechanism properly supports the defined variability including dependencies and constraints;

— supporting the testability verification of variability mechanisms used in implementation.

6.4 Variability mechanism operationalization at compile time

6.4.1 Purpose of variability mechanism operationalization at compile time

The purpose of this subprocess is to support variability mechanisms related to variabilities that will be resolved during compilation.

6.4.1.1 Inputs

— Variability bound at compile time.

— Domain variability model.

— Application variability model.

6.4.1.2 Outcomes

— *Architectural decisions and architectural texture on compile time* are extracted.

— *Variability mechanisms at compile time* are assessed and specified.

— *Post-compilation guides on variability mechanisms* are provided.

— *Compile time bindings* are performed.

— *Configurability in compile time* is enabled.

— *Compile time variability mechanism* is verified.

6.4.1.3 Tasks

— *Examine architectural decisions and architectural texture on compile time* is to find decisions, rules and constraints affecting decisions on variability mechanism operationalization at compile time.

— *Assess compile time variability mechanisms* is to explore, analyse and deploy proper variability mechanism for generating different configurations of member products.

— *Specify compile time variability mechanisms* is to describe variability mechanisms in artefacts that include variability that will be bound at compile time (E.g. directives and macros for conditional compilation).

— *Enable compile time configurability* is to prepare for the easy configuration of compiled objects for member products.

— *Prepare bindings at compile time* is to allow a member product to select variants consistently for which variation points are supposed to be bound at compile time.

— *Verify compile time variability mechanisms* is to analyse how well the defined variability mechanisms at compile time support variability described and enabled in compilation artefacts.

6.4.2 Examine architectural decisions and architectural texture on compile time

The goal of this task is to find rules and constraints on the usage of variability mechanism operationalization at compile time defined in architectural texture.

The method should support examining architectural decisions and architectural texture on compile time with the following capabilities:

— describing architecture rules in architectural texture for constraining admissible compile time variability mechanisms;

— exploring architecture rules among architectural decisions and textures;

— examining restrictions on the usage of variability mechanism operationalization at compile time.

A tool should support examining architectural decisions and architectural texture on compile time by allowing the user to do the following:

— providing description ways for architectural rules related to compile time variability mechanisms;

— storing restrictions on the usage of variability mechanism operationalization at compile time.

6.4.3 Assess compile time variability mechanisms

The goal of this task is to assess and select an optimal variability mechanism that appropriates for compilation binding time and the characteristics of variability.

The method should support assessing compile time variability mechanisms with the following capabilities:

— reviewing the admissible variability mechanisms whether they comply with the characteristic of variability;

— assuring whether a variability mechanism fully covers the defined variability including dependencies and constraints.

A tool should support assessing compile time variability mechanisms by allowing the user to do the following:

— providing compile time variability mechanisms pool with their usage guidance (e.g. where and how variability mechanisms have to be applied);

— storing rationales for variability mechanism selection.

6.4.4 Specify compile time variability mechanisms

The goal of this task is to provide ways to specify variability mechanism operationalization at compile time.

The method should support specifying compile time variability mechanisms with the following capabilities:

— representing (syntax and semantic definition) variability mechanisms in the artefacts, of which binding times are compile time;

— allowing easy of achieving the consistency and traceability of variability between artefacts including variability related to compile time binding and variability model.

A tool should support specifying compile time variability mechanisms by allowing the user to do the following:

— supporting notation for variability mechanism representation in the artefacts including variability, of which binding times are compile time;

— maintaining trace links between variability specified using compile time variability mechanism and variability model.

6.4.5 Enable compile time configurability

The goal of this task is to prepare the configurations of member products.

The method should support enabling compile time configurability with the following capabilities:

— incorporating variability mechanisms into domain artefacts (e.g. macros or conditional compilation constants in codes) with ways to derive configurations for member products;

— providing ways (e.g. parameters in conditional compilation) to generate different object file configurations;

— providing ways to reason about valid configurations at compile time.

A tool should support enabling compile time configurability by allowing the user to do the following:

— generating object file configuration in accordance with the defined ways (e.g. dialog box for setting conditional constants).

6.4.6 Prepare bindings at compile time

The goal of this task is to support preparation for bindings at compile time from variability mechanism views.

The method should support preparing bindings at compile time with the following capabilities:

— selecting the variants at compile time complying with variability mechanisms;

— converting variation point in domain artefacts of which binding times are compile time into those for a member product.

A tool should support preparing bindings at compile time by allowing the user to do the following:

— storing and displaying a variation point with its variants according to compile time variability mechanism for supporting proper binding;

— notifying when bindings are not properly conducted.

6.4.7 Verify compile time variability mechanisms

The goal of this task is to verify compile time variability mechanisms in their selection, specification and utilization.

The method should support verifying compile time variability mechanisms with the following capabilities:

— verifying the selection of compile time variability mechanisms (adhering the architecture rules);

— confirming variability mechanisms at compile time (e.g. compile time variability mechanisms in codes) whether they can correctly implement the defined variability dependencies;

— confirming variability mechanisms at compile time whether they can correctly implement the defined constraints;

— providing evaluation algorithm for verifying whether a variability mechanism conforms to the defined variability dependencies and constraints;

— verifying the testability of variability mechanisms used in compilation.

A tool should support verifying compile time variability mechanisms by allowing the user to do the following:

— visualizing variability mechanisms related to compilation;

— supporting the verification of whether the variability mechanism properly supports the defined variability including dependencies and constraints;

— supporting the testability verification of variability mechanisms used in compilation.

6.5 Variability mechanism operationalization at post-compile time

6.5.1 Purpose of variability mechanism operationalization at post-compile time

The purpose of this subprocess is to support variability mechanisms related to variabilities that will be resolved during linking and loading.

6.5.1.1 Inputs

— Variability bound at link time.

— Domain variability model.

— Application variability model.

6.5.1.2 Outcomes

— *Architectural decisions and architectural texture affecting post-compile time* are extracted.

— *Variability mechanisms at post-compile time* are assessed and specified.

— *Parameter table for post-compile time reconfiguration* is refined.

— *Post-compile time bindings* are performed.

— *Configurability at post-compile time* is enabled.

— *Post-compile time variability mechanism* is verified.

6.5.1.3 Tasks

— *Examine architectural decisions and architectural texture affecting post-compile time* is to find the forms of post-compilation bindings and to know how and where a member product is built and deployed.

— *Assess post-compile time variability mechanisms* is to explore, analyse and deploy proper variability mechanisms that make possible to build and deploy different sequence of components.

— *Specify link time variability mechanisms* is to describe variability mechanisms that make different sequence of linkages.

— *Specify load time variability mechanisms* is to describe variability mechanisms that makes the different combinations of configurations.

— *Specify deployment time variability mechanisms* is to describe variability mechanisms in deployment and installation scripts that include variability that will be bound at run time.

— *Enable post-compile time configurability* is to prepare for the easy reconfiguration or redeployment of executables for a member product (e.g. makefile generator).

— *Prepare bindings at post-compile time* is to allow a member product to select variants consistently for which variation points are supposed to be bound at post-compile time.

— *Verify post-compile time variability mechanism* is to analyse how well the defined variability mechanisms at post-compile time.

6.5.2 Examine architectural decisions and architectural texture affecting post-compile time

The goal of this task is to find rules and constraints on the usage of variability mechanism operationalization at post-compile time defined in architectural texture.

The method should support examining architectural decisions and architectural texture affecting post-compile time with the following capabilities:

— describing architecture rules in architectural texture for constraining the use of variability mechanisms in link time (e.g. sequences and dependencies among binary components);

— describing architecture rules in architectural texture for constraining the use of variability mechanisms in load time (e.g. rules for the right localization and initialization of configuration files including variability);

— exploring architecture rules related to post-compile time binding among architectural decisions and textures;

— examining restrictions on the usage of variability mechanism operationalization at post-compile time.

A tool should support examining architectural decisions and architectural texture affecting post-compile time by allowing the user to do the following:

— providing description ways for architectural rules related to link time variability mechanisms;

— providing description ways for architectural rules related to load time variability mechanisms;

— storing restrictions on the usage of variability mechanism operationalization at post-compile time.

6.5.3 Assess post-compile time variability mechanisms

The goal of this task is to assess and select an optimal variability mechanism that appropriates for post-compile time binding and the characteristics of variability.

The method should support assessing post-compile time variability mechanisms with the following capabilities:

— reviewing the admissible variability mechanisms whether they comply with the characteristic of variability;

— assuring whether a variability mechanism fully covers the defined variability including dependencies and constraints.

A tool should support assessing post-compile time variability mechanisms by allowing the user to do the following:

— providing post-compile time variability mechanisms pool with their usage guidance (e.g. where and how variability mechanisms have to be applied);

— storing rationales for variability mechanism selection.

6.5.4 Specify link time variability mechanisms

The goal of this task is to provide ways to specify variability mechanisms in link time.

The method should support specifying link time variability mechanisms with the following capabilities:

— representing variability mechanism for supporting different linkages (e.g. makefile) related to link time variability;

— allowing easy of achieving the consistency and traceability of variability between artefacts including variability related to link time binding and variability model.

A tool should support specifying link time variability mechanisms by allowing the user to do the following:

— supporting notation for variability mechanism representation in the artefacts (e.g. makefile) including variability, of which binding times are link time;

— maintaining trace links between the variability specified using link time variability mechanism and variability model.

6.5.5 Specify load time variability mechanisms

The goal of this task is to provide ways to specify variability mechanisms in load time.

The method should support specifying load time variability mechanisms with the following capabilities:

— representing variability mechanism for supporting different loadings (e.g. configuration file);

— allowing easy of achieving the consistency and traceability of variability between artefacts including variability related to load time binding and variability model.

A tool should support specifying load time variability mechanisms by allowing the user to do the following:

— supporting notation for variability mechanism representation in the artefacts including variability, of which binding times are load time;

— maintaining trace links between the variability specified using load time variability mechanism and variability mode.

6.5.6 Specify deployment time variability mechanisms

The goal of this task is to provide ways to specify variability mechanisms in deployment time.

The method should support specifying deployment time variability mechanisms with the following capabilities:

— representing variability mechanism in artefacts related to deployment time variability;

— allowing easy of achieving the consistency and traceability of variability between artefacts including variability related to deployment time binding and variability model.

A tool should support specifying deployment time variability mechanisms by allowing the user to do the following:

— supporting notation for variability mechanism representation in the artefacts including variability, of which binding times are deployment time;

— maintaining trace links between the variability specified using deployment time variability mechanism and variability mode.

6.5.7 Enable post-compile time configurability

The goal of this task is to prepare the configurations of member products.

The method should support enabling post-compile time configurability with the following capabilities:

— incorporating variability mechanisms into post-compile artefacts (e.g. makefile, configuration file) with ways to derive configurations for member products;

— providing ways (e.g. parameters in makefile) to generate different executable configurations;

— providing ways to reason about valid configurations at post-compile time.

A tool should support enabling post-compile time configurability by allowing the user to do the following:

— generating executable configuration in accordance with the defined ways (e.g. dialog box for setting conditional constants);

— validating the generated executable configurations.

6.5.8 Prepare bindings at post-compile time

The goal of this task is to support preparation for bindings at post-compile time from variability mechanism views.

The method should support preparing bindings at post-compile time with the following capabilities:

— selecting the variants at post-compile time (e.g. parameter values in makefile and configuration file) complying with variability mechanisms;

— converting variation point in post-compilation artefacts, of which binding times are post-compile time into those for a member product.

A tool should support preparing bindings at post-compile time by allowing the user to do the following:

— storing and displaying a variation point with its variants according to post-compile time variability mechanism for supporting proper binding;

— notifying when bindings are not properly conducted.

6.5.9 Verify post-compile time variability mechanism

The goal of this task is to verify post-compile time variability mechanisms in their selection, specification and utilization.

The method should support verifying post-compile time variability mechanisms with the following capabilities:

— verifying the right selection of post-compile time variability mechanisms (adhering the architecture rules);

— confirming variability mechanisms at post-compile time whether they can correctly support the defined variability dependencies;

— confirming variability mechanisms at post-compile time whether they can correctly support the defined constraints;

— providing evaluation algorithm for verifying whether a variability mechanism confirms the defined variability dependencies and constraints;

— verifying the testability of variability mechanisms used in post-compilation.

A tool should support verifying post-compile time variability mechanisms by allowing the user to do the following:

— visualizing variability mechanisms related to post-compilation;

— supporting the verification of whether the variability mechanism properly supports the defined variability including dependencies and constraints;

— supporting the testability verification of variability mechanisms used.

6.6 Variability mechanism operationalization at run time

6.6.1 Purpose of variability mechanism operationalization at run time

The purpose of this subprocess is to support variability mechanisms related to variabilities that will be resolved during execution. Selecting variants at start-up and user preference setting is typically used run time binding example. Variability mechanism operationalization at run time should allow user intervention or automatically detect context conditions such as self-adaptive and self-healing systems.

6.6.1.1 Inputs

— Variability bound at run time.

— Domain variability model.

— Application variability model.

6.6.1.2 Outcomes

— *Architectural decisions and architectural texture affecting run time reconfiguration* are extracted.

— *Variability mechanism operationalization at run time* are assessed and specified.

— *Run time reconfiguration decisions* are defined and managed.

— *Run time bindings* are performed.

— *Configurability in run time* is enabled.

— *Run time variability mechanism* is verified.

6.6.1.3 Tasks

— *Examine architectural decisions and architectural texture affecting run time reconfiguration* is to capture decisions and rules related to the run time changes or parts modified during execution.

— *Assess run time variability mechanism* is to explore, analyse and deploy proper variability mechanisms that make possible to execute different interfaces or different components among the loaded executables for execution.

— *Enable run time configurability* is to prepare for the easy configuration of the different executions for member products.

— *Prepare bindings at run time* is to allow a member product to select variants consistently for which variation points are supposed to be bound at run time.

— *Verify run time variability mechanism* is to analyse how well the defined variability mechanism operationalization at run time support variability described and enabled in run time artefacts.

6.6.2 Examine architectural decisions and architectural texture affecting run time reconfiguration

The goal of this task is to find rules and constraints on the usage of variability mechanism operationalization at run time defined in architectural texture.

The method should support examining architectural decisions and architectural texture affecting run time reconfiguration with the following capabilities:

— describing architecture rules in architectural texture for constraining admissible run time variability mechanisms;

— exploring architecture rules among architectural decisions and textures;

— examining restrictions on the usage of variability mechanism operationalization at run time.

A tool should support examining architectural decisions and architectural texture affecting run time reconfiguration by allowing the user to do the following:

— providing description ways for architectural rules related to run time variability mechanisms;

— storing restrictions on the usage of variability mechanism operationalization at run time.

6.6.3 Assess run time variability mechanism

The goal of this task is to assess and select an optimal variability mechanism that appropriates for run time binding and the characteristics of variability.

The method should support assessing run time variability mechanism with the following capabilities:

— reviewing the admissible variability mechanisms whether they comply with the characteristic of variability;

— assuring whether a variability mechanism fully covers the defined variability including dependencies and constraints.

A tool should support assessing run time variability mechanism by allowing the user to do the following:

— providing run time variability mechanisms pool with their usage guidance (e.g. where and how variability mechanisms have to be applied);

— storing rationales for variability mechanism selection.

6.6.4 Enable run time configurability

The goal of this task is to prepare the configurations of member products.

The method should support enabling run time configurability with the following capabilities:

— providing technology to combine components at run time (e.g. registry);

— evaluating configuration parameter values at run time.

A tool should support enabling run time configurability by allowing the user to do the following:

— supporting the combination of variable components into the system at run time;

— supporting the validation of run time re-configuration.

6.6.5 Prepare bindings at run time

The goal of this task is to support preparation for bindings at run time from variability mechanism views.

The method should support preparing bindings at run time with the following capabilities:

— providing the ways to register components (e.g. initialization mechanism to register components to registry) with interfaces to be bound at run time;

— combining components to different components in accordance with the needs of a member product.

A tool should support preparing bindings at run time by allowing the user to do the following:

— initializing components with their required interfaces for run time binding;

— notifying when bindings are not properly conducted.

6.6.6 Verify run time variability mechanism

The goal of this task is to verify run time variability mechanisms in their selection, specification and utilization.

The method should support verifying run time variability mechanisms with the following capabilities:

— verifying the right selection of run time variability mechanisms (adhering the architecture rules);

— confirming variability mechanism operationalization at run time whether they can correctly support the defined variability dependencies;

— confirming variability mechanism operationalization at run time whether they can correctly support the defined constraints;

— providing evaluation algorithm for verifying whether a variability mechanism confirms the defined variability dependencies and constraints;

— verifying the testability of variability mechanisms used in run time.

A tool should support verifying run time variability mechanisms by allowing the user to do the following:

— visualizing variability mechanisms related to run time;

— supporting the verification of whether the variability mechanism properly supports the defined variability including dependencies and constraints;

— supporting the testability verification of variability mechanisms used.

6.7 Variability mechanism operationalization for test artefacts

6.7.1 Purpose of variability mechanism operationalization for test artefacts

The purpose of this subprocess is to support variability mechanisms related to variabilities that will be resolved during testing.

6.7.1.1 Inputs

— Test strategy both organization level and product line level.

— Variability mechanisms relevant decisions made in requirements, architecture and realization.

— Variability of which binding time is testing stage.

— Domain variability model.

— Application variability model.

6.7.1.2 Outcomes

— *Decisions on variability mechanisms in test relevant development stages* are extracted.

— *Variability mechanisms at test level* are assessed and specified.

— *Bindings at testing stage* are performed.

— *Reusability in testing* is enabled.

— *Test level variability mechanism* is verified.

6.7.1.3 Tasks

— *Examine test strategy on variability mechanisms* is to find ways for achieving variability in test artefacts in line with test strategic decision.

— *Assess the decisions on variability mechanisms of requirements, architecture and realization* is to explore and analyse variability mechanisms used in requirements engineering, architecture and realization artefacts for the proper dealing with the variability in test artefacts.

— *Specify variability mechanisms in each test level* is to describe variability mechanisms in unit test, integration test and system test artefacts for domain.

— *Enable reusability in testing* is to support the easy reuse of a set of relevant test artefacts in line with the configuration of member products.

— *Prepare bindings at test stage* is to allow a member product to select variants consistently for which variation points are supposed to be bound at testing level.

— *Verify variability mechanism operationalization for test artefacts* is to analyse how well the defined variability mechanism operationalization for test artefacts stage support variability described and enabled in test artefacts.

6.7.2 Examine test strategy on variability mechanisms

The goal of this task is to find strategic decisions on the usage of variability mechanism operationalization for test artefacts stage.

The method should support examining test strategy on variability mechanisms with the following capabilities:

— providing description ways to variability mechanism related strategy in test strategy;

— examining rules, constraints or decisions on variability mechanisms in organizational level test strategy;

— identifying restrictions on variability mechanisms in test in accordance with product line test strategy (e.g. sample application strategy does not use variability mechanisms in test related to variable domain artefacts. It only uses variability mechanisms for implementing variability introduced in test).

A tool should support examining test strategy on variability mechanisms by allowing the user to do the following:

— supporting the description of variability mechanism related strategy (e.g. elements in template);

— sharing test strategy with relevant participants both organization level and product line level;

— storing examined test strategy on variability mechanisms.

6.7.3 Assess the decisions on variability mechanisms of requirements, architecture and realization

The goal of this task is to assess variability mechanisms used for implementing variability bound at requirements, architecture and realization stage respectively, so as to select proper variability mechanism operationalization for test artefacts.

The method should support assessing the decisions on variability mechanisms of requirements, architecture and realization with the following capabilities:

— evaluating decisions on variability mechanism operationalization for requirements engineering objectively;

— providing objective evaluation for decisions on variability mechanisms in architecture;

— evaluating decisions on variability mechanism operationalization for realization objectively;

— selecting and/or recommending proper variability mechanism operationalization for test artefacts.

A tool should support assessing the decisions on variability mechanisms of requirements, architecture and realization by allowing the user to do the following:

— accessing decisions on variability mechanisms in requirements, architecture and realization;

— accessing defined rules, constraints and guides related to variability mechanisms;

— recording evaluation results;

— storing information of selected variability mechanism operationalization for test artefacts.

6.7.4 Specify variability mechanisms in each test level

The goal of this task is to provide ways to specify variability mechanisms in unit testing, integration testing and system testing artefacts.

The method should support specifying variability mechanisms in each test level with the following capabilities:

— representing variability mechanisms in test artefacts for the used variability mechanisms;

— representing variability mechanism in test artefacts for test variability;

— allowing easy of achieving the consistency and traceability of variability between artefacts including test level variability and variability model.

A tool should support specifying variability mechanisms in each test level by allowing the user to do the following:

— supporting notation for variability mechanism representation in the test artefacts including variability;

— maintaining trace links between specified test variability and variability model.

6.7.5 Enable reusability in testing

The goal of this task is to support the reusability of test assets by organizing them well-coordinated with the configurations of member products.

The method should support enabling reusability in testing with the following capabilities:

— developing ways to trace variability in test assets with variability in development assets;

— selecting test assets for testing configurations at unit test, integration test and system test level.

A tool should support enabling reusability in testing by allowing the user to do the following:

— implementing ways to trace variability in test assets with variability in development assets;

— providing automatic selection for test assets in line with configurations at unit test, integration test and system test level.

6.7.6 Prepare bindings at test stage

The goal of this task is to support preparation for bindings at each test level from variability mechanism views.

The method should support verifying run time variability mechanisms with the following capabilities:

— providing ways to make correct binding in testing in accordance with the binding results in development stages;

— completing the test assets including variability;

— preparing correct bindings for test variability (test variability means a variability for allowing different test coverage, test techniques, etc.).

A tool should support verifying run time variability mechanisms by allowing the user to do the following:

— linking binding information of development stage to test assets including relevant variability;

— presenting binding information (e.g. dependency, constraints, relevant rules) for test variability.

6.7.7 Verify variability mechanism operationalization for test artefacts

The goal of this task is to verify variability mechanisms in test for their selection, specification and utilization.

The method should support verifying run time variability mechanisms with the following capabilities:

— verifying the right selection of run time variability mechanisms (adhering the architecture rules);

— confirming variability mechanism operationalization at run time whether they can correctly support the defined variability dependencies;

— confirming variability mechanism operationalization at run time whether they can correctly support the defined constraints;

— providing evaluation algorithm for verifying whether a variability mechanism confirms the defined variability dependencies and constraints;

— verifying the testability of variability mechanisms used in run time.

A tool should support verifying run time variability mechanisms by allowing the user to do the following:

— visualizing variability mechanisms related to run time;

— supporting the verification of whether the variability mechanism properly supports the defined variability including dependencies and constraints;

— supporting the testability verification of variability mechanisms used.

7 Variability mechanism support

For the correct operation of variability mechanisms variability mechanisms used should have traces with relevant variability in variability model. When variability mechanisms are applied to variability realization, decisions related to bindings should be made and documented so that application engineering stages adhere or refer those decisions when bindings occur.

Variability mechanism support provides the following supports:

— relating variability mechanism to variability model;

— quality assurance for variability mechanism;

— binding time decision support;