2012 Joint Working Conference on Software Architecture & 6th European Conference on Software Architecture

A Variability Viewpoint for Enterprise Software Systems

Matthias Galster
University of Groningen, The Netherlands
mgalster@ieee.org

Paris Avgeriou
University of Groningen, The Netherlands
paris@cs.rug.nl

Abstract—**Many of today's enterprise software systems are subject to variability. For example, enterprise software systems often run in different business units of an organization, with each unit having its own detailed requirements. Systematic handling of variability allows a software system to be adjusted for different contexts, by planning for adaptation during architecture design. As variability is system-wide, it is reflected in the software architecture. To facilitate the representation and analysis of variability in the architecture of enterprise software systems, we propose an architecture viewpoint. To define a reusable variability viewpoint, we elicited stakeholders and concerns through exploratory studies. We also show how the viewpoint was applied for describing variability in a large-scale e-government system.**

Keywords-variability; software architecture; viewpoints

I. INTRODUCTION

Variability is the ability of a software to be adapted for a specific context [1] to enable multiple deployment scenarios or versions of a software system. To enable variability, parts of the architecture are not fully defined during early design, but later when more details about concrete usage scenarios are known. During early iterations, architects identify what parts of a system should be variable (e.g., in terms of "variation points") and how to resolve this variability (e.g., in terms of "variants" or ranges of variants). Later, these variants are used to resolve variability.

Many of today's software systems are built with variability in mind, e.g., product families, self-adaptive, customizable single systems, open platforms, or service-based systems that support dynamic composition of web services. One prominent category of variability-intensive software systems are enterprise software systems (ESS). ESS often run in different business units of an organization with their specific requirements, or in different countries in which a company operates with specific constraints on business processes. Differences in these deployment scenarios affect business processes as well as functionality of ESS.

Identifying and managing variability of a system early on, and in particular during architecting, is preferred over addressing variability later. As variability is pervasive and affects many stakeholders, architects need proper support for representing, reasoning about and managing variability. However, describing variability in ESS is often performed in a fragmented way where a) each architecture model only covers few variability concerns and b) dependencies between models are not taken into consideration. Thus, we propose an architecture viewpoint [2] for variability to provide a broader description of variability in the architecture of ESS. The viewpoint helps construct a view of an ESS comprising several complementary models that address detailed variability concerns, rather than treating variability as one high-level concern within one model. To reduce discrepancy between architecture practice and research, evidence for the validity of concerns and model elements exists in the sense that concerns represent real stakeholder interests derived from empirical studies. Furthermore, the viewpoint follows the conventions of ISO / IEC 42010 [2]. In contrast to enterprise frameworks, e.g., [3], we focus on variability and its impact on the architecture; thus, our work differs in stakeholders, concerns and models.

Section II of this paper outlines previous work. In Section III we discuss how we defined the viewpoint. The viewpoint itself is introduced in Section IV. A discussion is presented in Section V before we conclude in Section VI.

II. BACKROUND AND RELATED WORK

Variability has primarily been studied in the software product line (SPL) domain [4]. For example, variability in ESS from a product line perspective has been investigated in [5]. Product line architectures describe variability explicitly as in terms of "features" and "decisions" and encompass limited conceptual models, such as feature models, decision models or component-and-connector models, often in isolation. However, a holistic view on concerns and models is currently missing. Most importantly, a product line assumes the existence of a product line infrastructure and related processes. This is rarely the case for many architectures which should support variability.

Recently, architecture viewpoints have gained popularity to describe software architectures [2]. We show the proposed ESS variability viewpoint in the context of ISO / IEC 42010 in Fig. 1 (dotted elements were added to indicate that the definition of the ESS variability viewpoint is driven by tasks that stakeholders perform and that tasks are supported by the views created based on the viewpoint).

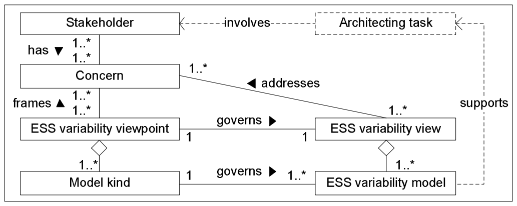


Figure 1. Concepts in the context of the ESS variability viewpoint.

A viewpoint frames concerns which are addressed in views. A viewpoint consists of conventions for constructing and interpreting a view. A model uses conventions specified by the model kind governing that model and prescribed by the viewpoint. Previous works proposed viewpoints for change or evolution, but do not express variability explicitly.

III. Procedure to Define Variability Viewpoint

An overview of our procedure to define the ESS variability viewpoint is shown in Fig 2.

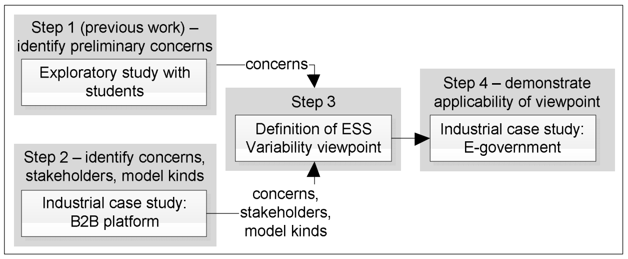

Step 1 (previous work) – identify preliminary concerns
Exploratory study with students
concerns
Step 3
Definition of ESS Variability viewpoint
Step 4 – demonstrate applicability of viewpoint
Industrial case study: E-government
Step 2 – identify concerns, stakeholders, model kinds
Industrial case study: B2B platform
concerns, stakeholders, model kinds

Figure 2. Procedure to develop variability viewpoint.

A. *Exploratory Study with Students*

In a previous study we identified eleven problems that occur when performing variability-related tasks during software architecting [6]. Subjects were software engineering graduate students with practical experience. As we aim at a viewpoint that helps software architects perform their tasks, problems identified in this study can be interpreted as potential concerns to be framed by a viewpoint. Concerns identified based on problems are discussed in Section IV.

B. *Industrial Case Study: B2B Platform*

We conducted an exploratory case study in a large software organization. The goal was to identify and confirm stakeholders, concerns and model kinds to define a variability viewpoint for ESS. Stakeholders, concerns and model kinds elicited from industry lead to more useful viewpoints for practitioners. The case in the single case study [7] was a project for a large customer of the software organization. The customer did not agree to release any details that could reveal its identity. The project we studied was about a web selling platform to sell a broad variety of physical products in a business-to-business context. We selected this case because a) to get relevant stakeholders, concerns and insights into relevant model kinds, we aimed at a large-scale project, b) the produced software in the project is used by one customer organization with substantial variability in its business processes, c) using a "crucial case selection" strategy [8], we aimed at a representative software development organization / project. The unit of analysis in the holistic study design was variability and challenges that lead to concerns, and related model kinds. The system we studied is variability-intensive as it needs to support many variation points for many business units with different business rules. Furthermore, each business unit has different backend systems. Finally, there are different roll-outs of the system (e.g., for each region). For data collection we used direct contact with representatives of the software organization. We took notes and copies from whiteboard discussions. We described the business processes of the B2B platform, variability, concerns and stakeholders. We discussed how to address concerns to identify model kinds. The analysis of results is presented in Section IV.

C. *Industrial Case Study: E-government*

The goal of this case study was to assess the ESS variability viewpoint to evaluate its applicability from the perspective of software architects of a large software-intensive system. The single case was a software solution in the context of Dutch local e-government. The software supports the implementation of the Dutch law that mandates rules for providing social support to citizens, such as domestic care (so called WMO law). We selected this case for similar reasons as in the B2B case study. The unit of analysis was the ESS variability viewpoint and the variability view and models created based on this viewpoint. The e-government system is variability-intensive because even though the law has been approved by the Dutch national government and all municipalities must provide the same service to citizens, the solutions chosen to implement this law can differ substantially between municipalities. Differences between municipalities are too big to implement solutions as one product for all municipalities, yet not too big to be covered by one generic solution. For data collection we interviewed stakeholders from municipalities as well as software vendors and domain experts, and studied documents related to implementing software systems for the WMO law. We described the business process of the WMO law and framed variability concerns according to the variability viewpoint. Data analysis happened through content analysis of interview data and process descriptions, and through constructing the variability view and related (partially shown in Section IV).

IV. ESS Variability Viewpoint

In the B2B case study we identified the following stakeholders: architects who describe architecture (SH1), customers who operate and use a product (SH2), evaluators who evaluate the architecture (SH3) and domain experts who identify commonalities and variations in a software product (SH4). When expressing needs, practitioners think of tasks they perform [9]. We found the following tasks reflected in the case study [10] (for each task we state the corresponding stakeholders): T1: Variability identification (decide what variability is needed and where; SH2 and SH4). T2: Variability constraining (ensure that just enough flexibility is provided in the architecture, rather than limitless flexibility; SH1, SH2, SH3 and SH4). T3: Variability implementation (select suitable realization techniques; SH1 and SH3). T4: Variability management (evolution, maintenance; SH1 and SH3). Table I shows the concerns of stakeholders framed by the viewpoint and elicited from the exploratory study from Section III.A (S1) and from the B2B case study (S2). In Table II we map concerns to stakeholders, as identified in the B2B case study. Their relevance was confirmed in the e-government case study.

TABLE I. CONCERNS FRAMED BY ESS VARIABILITY VIEWPOINT

		Source		Task			
ID	Concern	S1	S2	T1	T2	T3	T4
C1	Where in the business process does variability occur?		x	x			x
C2	What types of variability occur in the business process?		x	x	x		x
C3	What variants are available to resolve a variation point?	x	x	x	x		x
C4	When would variability in the business process be resolved?	x	x		x		x
C5	Where in the architecture is variability needed?	x	x			x	x
C6	How does a variation point in the business process map to variability in the architecture?		x			x	x
C7	Is a variant a valid option at a variation point?	x	x		x		
C8	What are relationships between variation points and variants?	x	x			x	x
C9	How do variants and quality attributes interact?	x	x			x	x

TABLE II. RELATION BETWEEN STAKEHOLDERS AND CONCERNS

ID	Stakeholder	Concern
SH1	Architect	C1, C2, C3, C4, C5, C6, C7, C8, C9
SH2	Customer	C1, C2, C3, C4
SH3	Evaluator	C1, C2, C3, C4, C5, C6, C7, C8, C9
SH4	Domain expert	C1, C2, C3, C4

A. Viewpoint Metamodel

The viewpoint metamodel describes conceptual entities of the viewpoint. Inputs to defining the metamodel were model elements identified in the B2B case study. To frame all concerns identified in the previous section, the metamodel (Fig. 3) includes aspects of a variable business process and architecture in terms of software implementation artifacts.

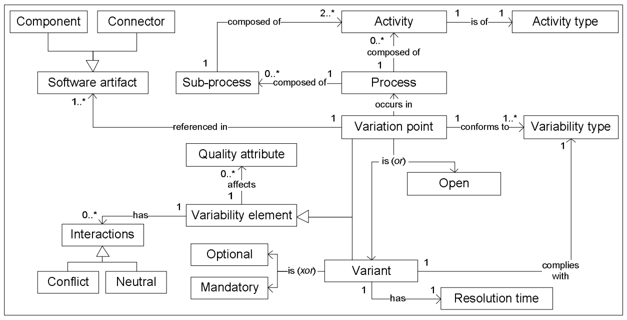


Figure 3. ESS variability viewpoint metamodel.

We use *process*, *sub-process* and *activity* to define a business process. *Activity type* includes: services (automated tasks performed by ESS), activities performed by users with the help of the ESS, and manual activities with no support from the ESS. To represent software artifacts, the metamodel utilizes basic *software artifacts* in terms of *components* and *connectors* because the viewpoint codifies architecture knowledge independent of a domain or technology. The metamodel can be "instantiated" (e.g., in the domain of service-oriented architecture, a component could be a service and a connector could be a web service request). *Variation points* are referenced in software artifacts, i.e., are resolved by the implementation in software artifacts. A variation point can be *open*, i.e., no variants are specified to resolve it, or can have a (*optional* or *mandatory*) *variant* assigned to it. We include two *interactions* between *variability elements* (*variants* and *variation points*): *conflict* or *neutral*. Furthermore, a variant has a *resolution time*, which can be runtime or design time. *Variability types* include activity (one activity can be replaced by another), sub-process (one sub-process is replaced by another), parameter (parameters used to invoke an activity or sub-process vary), parameter value (parameter values vary), flow and composition. The flow type refers to the fact that a business process describes a sequence of a workflow, and that activities and sub-processes can alternatively or optionally be used. Composition means that software components are composed depending on the variation point to resolve in the implementation. We include the impact of variability elements on *quality attributes*.

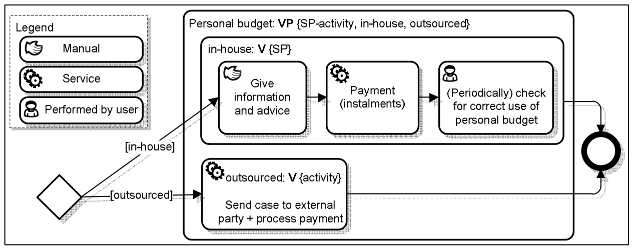
B. Viewpoint Model Kinds

The viewpoint supports six model kinds (Table III). All model kinds emerged from the B2B case study and comply with the shared viewpoint metamodel.

TABLE III. MODEL KINDS, CONCERNS AND STAKEHOLDERS

Model kind	Concern	Stakeholder
Business process variability	C1, C2, C3	SH2, SH3, SH4
Business process variation point	C2, C4, C7, C8	SH3, SH4
Variability distribution	C1	SH1, SH3
Variability mapping	C1, C5, C6	SH1
Variability interaction	C7, C8	SH1, SH3
Variability quality influence	C9	SH1, SH3

Business process variability model kind. This model kind governs models that describe the business level of the architecture of an ESS in terms of business processes. We use an annotated version of the Business Process Modeling Notation (BPMN) as the notation for models of this kind. We constructed the business process variability model in the e-government case study by eliciting business processes from municipalities and interviewing stakeholders. The business process consists of six sub-processes ("phases" in the WMO law) and more than 13 variation points.



Legend
Manual
Service
Performed by user
Personal budget: VP {SP-activity, in-house, outsourced}
in-house: V {SP}
Give information and advice
Payment (instalments)
(Periodically) check for correct use of personal budget
[in-house]
[outsourced]
outsourced: V {activity}
Send case to external party + process payment

Figure 4. Partial business process variability model.

Fig. 4 shows the variability in a business process, including variation points and variants of a sub-process to determine the personal budget of a citizen that requested social support through the WMO law ("Personal budget phase"). The start and end points are connected to other sub-processes of the WMO process. Boxes around variation points show the scope of the variation point. The variation point in this model is "Personal budget". Its type is a "SP-activity" (sub-process or activity), and variants include "in-house" (sub-process) and "outsourced" (activity).

Business process variation point model kind. Models based on this kind describe details about variation points in a business process. Fig. 5 shows the business process variation point model for variation point "Personal budget" from the business process variability model depicted in Fig. 4.

Type	Variant (type)	Mandatory	Resolution
Sub-process-activity	in-house (sub-process)	no	design
	outsourced (activity)	no	design

Figure 5. Partial business process variation point model.

Variability distribution model kind. Models created based on this model kind show where in the business process variability occurs (in terms of sub-processes of a business process), as well as what types of variability occur. The model created based on this model kind visualizes a frequency distribution of variation points and their type to help architects and evaluators identify sub-processes in a business process which require most attention with regard to their impact on the software architecture. The variability distribution model for the WMO process is shown in Fig. 6. "Research and decision phase" includes three variation points of different types. Variability in activities is usually easier to handle than variability in sub-processes which cause more ripple effects in the architecture.

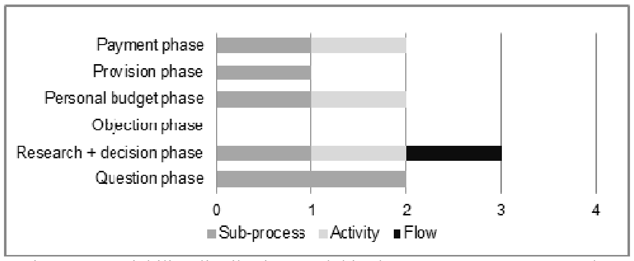


Figure 6. Variability distribution model in the e-government case study.

Variability mapping model kind. The variability mapping model kind describes the mapping of a variation point from the business process to software components (e.g., services). Fig. 7 shows the high-level variability mapping model including the variation point "Personal budget" from the sub-process "Personal budget phase" in the e-government case study. We also include the variation point "Question clarification" from sub-process "Question phase". The software artifacts are taken from a real system to support the WMO law that utilizes multi-tenancy to host tenants of different municipalities. The implementation is provided by a software vendor involved in the e-government case study and implemented in around 20 municipalities over the past five years. Variation point "Personal budget" maps to a software component "Process" which is an entity in a case template provided in a case template catalogue. This variation point is resolved by choosing a respective case template that implements the respective variant. Municipalities use this catalogue through their tenant. Case templates are customizable templates for products (e.g., a building permit) or processes (e.g., a WMO request) offered by municipalities. Similarly, the variation point "Question clarification" maps to tenants. Boxes in Fig. 7 denote components, and lines between boxes connectors. Variation points, including type and variants, are shown as ovals. Connections between variation points and software artifacts stem from the metamodel ("referenced in"). Connectors between software artifacts (e.g., "hosts", "contains") are not in the metamodel as it only defines generic connectors.

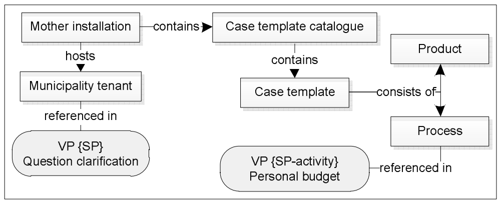


Figure 7. Partial variability mapping model in the e-government case study.

Variability interaction model kind. The variability interaction model kind provides conventions for models to shows relationships between variation points, variations, and variation points and variants. Fig. 8 shows a partial variability interaction model concerning interactions between variants for the WMO process. We used red color coding do highlight conflicts. Neutral interactions are simply represented by "-".

		Variant			
		Medical advice	Home visit	Phone conversation	Personal meeting
Variant	Medical advice	-	-	-	-
	Home visit	-	-	-	-
	Phone conversation	-	conflict	-	conflict
	Personal meeting	-	conflict	conflict	-

Figure 8. Variability interaction model in the e-government case study.

Variability quality influence model kind. The variability quality influence model kind describes conventions for specifying how variants (if implemented) could affect quality attributes. If a variation point is open then models show how variation points are related to quality attributes. Fig. 9 shows a partial variability quality influence model for the WMO process. Key drivers in e-government are privacy, performance and security.

		Key drivers		
		Privacy	Performance	Security
Variant	Medical advice	+	-	-
	Home visit	o	o	o
	Phone conversation	-	o	o
	Personal meeting	o	o	o

Figure 9. Partial variability quality influence model.

We express dependencies as "--" (strong negative impact), "-" (negative impact), "o" (neutral), "+" (positive impact), and "++" (strong positive impact). Color coding highlights influences of special interest.

V. Discussion

Applicability of viewpoint. Overall, the viewpoint is applicable in projects with similar characteristics as the two case study projects reported in this paper. Furthermore, the viewpoint and model kinds are not bound to any variability implementation technique. This makes the viewpoint applicable to other variability-intensive ESS than the ones we studied in this paper. Moreover, if needed, additional model kinds can be added to address other variability-related concerns; the viewpoint metamodel is extensible. We found that the viewpoint provides a structured way to analyze variability with regard to concerns elicited from real software projects. The proposed model kinds help address the concerns of stakeholders. The model kinds used in the viewpoint are adequate to understand for example the interaction between variation points and quality attributes, or the interaction between variants in the e-government case study. Furthermore, even though stakeholders and concerns where identified in Step 1 and 2, we were able to confirm their relevance in the e-government case study. In some situations only a subset of model kinds might be selected, in particular when product lines are developed. In these situations, some models might already exist that are similar to the model kinds proposed in the viewpoint (e.g., the variability interaction model could be replaced by a conventional feature model). The ESS variability view in the e-government case study has been created using standard software modeling tools. Dedicated and more mature tools could further increase the applicability of the viewpoint.

Limitations of constructing the viewpoint. We used an exploratory study with graduate students to identify problems when handling variability during software architecting. Limitations related to this study are discussed in [6]. Further limitations are related to using the case study research methodology for defining and applying the viewpoint (e.g., construct validity, external validity, reliability). In both case studies we used multiple sources of data to identify and evaluate concerns, stakeholders and model kinds. Furthermore, we chose representative cases of variability-intensive systems but cannot claim complete generalizability of the concerns and model kinds. However, using the model kinds to construct an ESS variability view in the e-government case study showed that concerns in this case study were in fact accommodated by the view. We documented the characteristics of both case studies to help decide if our findings might be applicable in other similar situations (see also previous section). To mitigate the risk of forcing our idea of architecture documentation on the documentation of the e-government system, we involved stakeholders in the e-government system.

Limitations of viewpoint. Even though all model kinds are related and the full power of the viewpoint is leveraged by using all model kinds, the creation of views requires effort. We have not conducted a thorough cost analysis of using the viewpoint. However, neither in the B2B case study nor in the e-government case study architecture descriptions for variability existed. Thus, the ESS variability view for these systems is complementary to existing architecture descriptions. Also, the viewpoint currently does not support variability in quality attributes. We did not find an indication in the two case studies for the need to express variability in quality attributes. Furthermore, stakeholders and concerns were elicited from empirical studies but particular projects might have additional stakeholders or concerns. The viewpoint metamodel can be extended to accommodate these concerns. Finally, we do not describe correspondence rules to express "cross-model" correspondences. Constraints are currently determined by the shared metamodel.

VI. Conclusions

The ESS variability viewpoint provides reusable architecture knowledge and helps construct variability views based on the needs derived from industrial organizations. An ESS variability view helps reason about variability concerns. We have shown how variability models can be constructed starting from business process variability and describe how variability affects the software architecture design. As part of our future work, we study additional model kinds (e.g., to express variability in quality attributes). Moreover, we will integrate the viewpoint in a reference architecture design process for variability-intensive software systems. Finally, we will relate the ESS variability viewpoint to viewpoints for documenting runtime behavior and architecture decisions.

Acknowledgment

We thank the organizations involved in the studies, and R. Hilliard and U. van Heesch for discussions. This research has been sponsored by NWO SaS-LeG, 638.000.000.07N07.

References

[1] F. Bachmann and P. C. Clements, "Variability in Software Product Lines," SEI CMU, Pittsburgh, PA, Technical Report CMU/SEI-2005-TR-012, 2005.

[2] ISO/IEC, "Systems and Software Engineering - Architecture Description." ISO/IEC 42010, 2011.

[3] J. Zachman, "A Framework for Information Systems Architecture," *IBM Systems Journal,* vol. 38, pp. 454-470, 1999.

[4] L. Chen, M. A. Babar, and N. Ali, "Variability Management in Software Product Lines: A Systematic Review," in *13th International Software Product Line Conference (SPLC)* San Francisco, CA: Carnegie Mellon University, 2009, pp. 81-90.

[5] Y. Ishida, "Software Product Lines Approach in Enterprise System Development," in *11th International Software Product Line Conference* Kyoto, Japan: IEEE Computer Society, 2007, pp. 44-53.

[6] M. Galster and P. Avgeriou, "Handling Variability in Software Architecture: Problems and Implications," in *9th IEEE/IFIP Working Conference on Software Architecture* Boulder, CO: IEEE Computer Society, 2011, pp. 171-180.

[7] R. K. Yin, Case Study Research - Design and Methods. London, UK: Sage Publications, 2009.

[8] J. Gerring, Case Study Research - Principles and Practices. Cambridge, NY: Cambridge University Press, 2006.

[9] T. B. C. Arias, P. America, and P. Avgeriou, "Defining and Documenting Execution Viewpoints for a Large and Complex Software-intensive System," *Journal of Systems and Software,* p. 15, 2010.

[10] M. Svahnberg, J. van Grup, and J. Bosch, "A Taxonomy of Variability Realization Techniques," *Software - Practice and Experience,* vol. 35, pp. 705-754, April 2005.