Evaluating variability at the software architecture level: An Overview

Ana Paula Allian*
University of São Paulo
Brazil
ana.allian@usp.br

Bruno Sena
University of São Paulo
Brazil
brunosena@usp.br

Elisa Yumi Nakagawa
University of São Paulo
Brazil
elisa@icmc.usp.br

ABSTRACT

Software architecture are designed for developing software systems needed for a diverse of business goals. Consequently, architecture has to deal with a significant amount of variability in functionality and quality attributes to create different products. Due to this variability, the evaluation in software architectures is much more complex, as different alternatives of systems might be developed leading to an expensive and time consuming task. Several methods and techniques have been proposed to evaluate product line architectures (PLAs) aiming to asses whether or not the architecture will lead to the desired quality attributes. However, there is little consensus on the existing evaluations methods is most suitable for evaluating variability in software architectures, instead of only considering PLAs. Understanding and explicitly evaluating variations in architectures is a cost-effective way of mitigating substantial risk to organizations and their software systems. Therefore, the main contribution of this research work is to present the state of the art about means for evaluating software architectures (including, PLAs, software architectures, reference and enterprise architectures) that contain variability information. We conducted a Systematic Mapping Study (SMS) to provide an overview and insight to practitioners about the most relevant techniques and methods developed for this evaluation. Results indicate that most evaluation techniques assess variability as a quality attribute in PLAs through scenario-based; however, little is known about their real effectiveness as most studies present gaps and lack of evaluation, which difficult the usage of such techniques in an industrial environment.

CCS CONCEPTS

• **Software and its engineering → Software architectures**;

KEYWORDS

software architecture, software variability, evaluation, systematic mapping study

*This is the corresponding author

SAC '19, April 8–12, 2019, Limassol, Cyprus

ACM ISBN 978-1-4503-5933-7/19/04…$15.00
https://doi.org/10.1145/3297280.3297511

ACM Reference Format:
Ana Paula Allian, Bruno Sena, and Elisa Yumi Nakagawa. 2019. Evaluating variability at the software architecture level: An Overview. In *The 34th ACM/SIGAPP Symposium on Applied Computing (SAC '19), April 8–12, 2019, Limassol, Cyprus.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3297280.3297511

1 INTRODUCTION

Variability is understood as the ability of a software system to be efficiently changed (i.e., configured, customized, extended, adapted) to fit in different contexts, purposes and environments [21]. Although variability has been further explored in software product lines (SPL) within PLAs, it is a key fact of most systems and a relevant characteristic of their architectures. Variability in PLAs have limited scope when compared to software architectures, due to the fact PLAs focus on addressing variability as features and decisions, whereas software architecture often treat variability as quality attributes [22]. In addition, variability is a concern of a variety of stakeholders affecting multiple concerns, multiple deployment, operation, and maintenance scenarios, increasing even more the complexity of the evaluation aspect of these architectures. This is particularly true as variability in software architecture may lead to different alternatives of systems, which must be evaluated in order to mitigate architectural risks.

Several methods have been proposed to evaluate whether the software architecture fulfills its requirements based on scenarios and quality attributes, such as Software Architecture Analysis Method (SAAM) [26] and the Architecture Trade-off Analysis Method (ATAM) [27]. In the case of software architectures with variability, the evaluation must ensure their flexibility to support the development and evolution of different software products. For small scale architectures with few variabilities, the relation among elements can be tracked easily by manually analyzing the architecture model. But when we consider complex architectures such as PLAs, reference architectures, enterprise architectures, and architectures from big systems (including systems-of-systems, cyber-physical systems, and smart cities), more systematic evaluation methods are required for evaluating them. Therefore, it is important to understand the main methods and approaches employed to systematize the evaluation of software architectures with variability. With this understanding, software architects could decide which one better provide the rationale to fit into their business context.

The main goal of this paper is to present an overview of methods that have been used to evaluate software architectures with variability. To achieve this objective, we conducted an SMS following a predefined guideline aiming to answer our research question: "How software architectures that contain variability information have been evaluated?". The focus of this research is on studies that

provide evaluation methods with detailed steps and techniques for evaluating this type of architectures.

The remainder of this paper is structured as follows: Section 2 discusses related works; Section 3 presents the process used to conduct this SMS; Section 4 outlines the results of this SMS; Section 5 presents a brief discussion of the results; Section 6 presents threats to validity of this study, and in Section 7 we draw the conclusions for this work.

2 RELATED WORKS

Andersen and Carugati [2] conducted a systematic literature review (SLR) on methods and approaches for evaluating enterprise architectures. They focuses on the evaluation of business, technical and financial within enterprise architectures and they found out many of the studies were concerned with technical evaluation, including measurement of architecture complexity and interoperability of services to improve integration. However, none of the studies described the evaluation of variability in the architectures.

Barcelos and Travassos [8] described an SLR aiming to identify approaches for evaluating software architectural documentation. They classified the results in three main evaluation techniques: i) Questioning, which is characterized by the use of scenarios and checklists; ii) Measuring, which is based on quantitative results; and iii) Hybrid technique, which combines questioning and measuring techniques for the evaluation of architectures. Their classification is close to our study; however, in our SMS, we also classify the studies through evaluation criterium, evaluation-based approaches, and specific techniques for the evaluation of variability.

Ouhbi [41] conducted an SMS to summarize existing software architecture evaluation approaches and she classified them in six categories, including research, empirical, contribution, software quality models, quality attributes, and software architecture models. Results from this study indicated the need for evaluation approaches for the emerging new technologies such as evaluation approaches for big data and IoT architectures, but she did not mention the way architectures with variability could be evaluated.

Variabilities within the domain of SPL have been extensively studied and there seems to be an academic and industrial agreement that variability is more easily managed using the concept of feature models [10–12]. Chen et al. [13] conducted an SLR to provide an overview about approaches and notations used to describe variability in SPL. They found out 33 different approaches and none of them focus on supporting the evaluation of variability in architectures. Bashroush et al. [10] and Pereira et al. [42] conduced an SLR on SPL tools responsible for managing variability in different SPL lifecycle phases. Most studies addressed variability in terms of features models and configuration of products and none of the tools were specific for the evaluation of variability.

3 RESEARCH METHOD

The research method adopted in this study follows the guidelines proposed by Kitchenham and Charters [32] and Petersen et al. [43] for conducting a Systematic Mapping Study (SMS). The search strategy starts by the definition of the following research question (RQ): *RQ: How software architectures that contain variability information have been evaluated?*. The *search string* using the main terms that entail the most appropriate keywords pertaining to the scope of this SMS were also defined, as presented in Figure 1. Divergences about each term were discussed with experts in software architecture and variability.

(architecture AND (evaluat* OR assess*) AND (variability OR "variation point")

Figure 1: Search String

The primary studies were selected by conducting an automatic search. We selected the most important electronic databases: Association for Computing Machinery Digital Library (ACM DL)[1], IEEEXplore Digital Library[2], Ei Compendex (Engineering Village)[3], and Scopus[4]. The *search string* defined previously (Figure 1) was used against the metadata (titles, keywords, and abstracts) of each study in all selected electronic databases.

We also attempted a search on Springer[5] database; however, it overlapped the majority of studies already retrieved, because Springer is indexed by the search engines from Scopus and Ei Compendex databases. Besides, we did not include Google Scholar[6] since the search results of Google Scholar tend to be repetitive with the search results from the included electronic databases.

The selection of primary studies was based on predefined inclusion and exclusion criteria that are directly related to our RQ:

Inclusion Criteria:

IC1: The primary study addresses variability in software architectures and presents some contribution (approaches, processes, methods, techniques, or tools) for the evaluation of variability in software architectures.

Exclusion Criteria:

EC1: If two studies address the same topic, the less complete one is excluded.
EC2: Short studies below four pages;
EC3: Duplicated studies.
EC4: Studies containing an editorial, abstract, or introduction; and
EC5: Studies not written in English.

The search for the studies involved four activities as presented in Figure 2. A summary of the whole study selection process is presented as follows:

(1) **Recovery of studies:** resulted in **1,457** (showed in Table 1).
(2) **Initial exclusion:** all studies were downloaded and imported into Mendeley[7] library. 506 duplicated studies were removed by applying the exclusion criteria EC3. Furthermore, all papers that meet the exclusion criteria EC4 and

[1] http://dl.acm.org
[2] http://ieeexplore.ieee.org
[3] https://www.engineeringvillage.com
[4] https://www.scopus.com
[5] https://link.springer.com
[6] https://scholar.google.com
[7] https://www.mendeley.com/

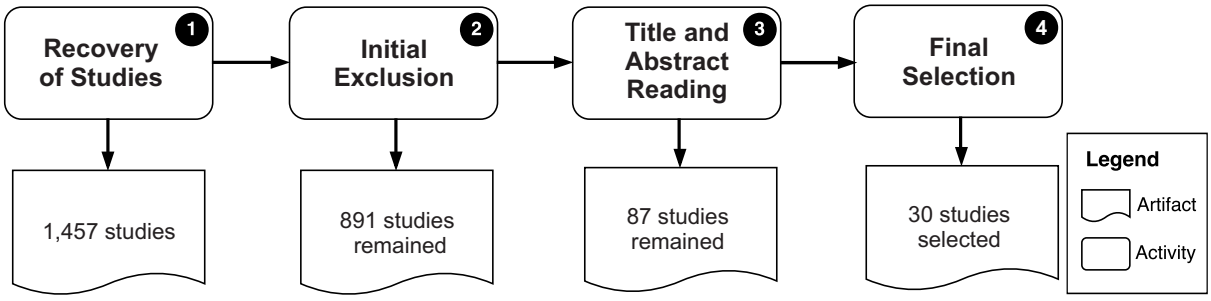

Figure 2: Selection process

EC5 were also removed, i.e., 60 studies. In the end, **891** studies were left for the third step of the study selection.

(3) **Title and abstract reading:** This consisted of reading the title and abstract of each study. In cases where the title and abstract were not enough to decide whether it should be included or excluded, the introduction and conclusion sections were read. Through this step, 804 studies were excluded and 87 studies were left for the next step.

(4) **Final selection:** After the completing reading of each paper selected in previous step, we selected **30 studies** that present means for the evaluation of architectures with variability.

Table 1: Electronic databases included in the SMS.

Electronic database	Metadata used	#
ACM DL	Paper title, abstract	78
IEEExplore	Paper title, keywords, abstract	376
Ei Compendex	Paper title, abstract	408
Scopus	Paper title, keywords, abstract	595

4 ANALYSIS OF THE RESULTS

Existing literature provides several methods to assess the variability and, consequently, improve quality of software architectures. The analysis of results allowed us to present the frequencies of studies for different architecture level, including PLAs, software architectures, reference and enterprise architectures. This makes it possible to see what types of architectures have been emphasized in past research and thus identify gaps for future research. Methods such as SAAM [26] and ATAM [27] are widely referenced and applied both in academia and industry and a number of extensions of these methods for different types of architectures were proposed as follows.

4.1 Scenario-based Evaluation

Scenario-based architecture evaluation is used to refine quality attributes and requirements into concrete scenarios created by experts stakeholders to fulfill their concerns related to such requirements. In this SMS, we found a variety of assessment proposals for architectures using scenario-based approaches and we briefly describe each study as follows.

Eickelmann and Richardson [17] (S26) extended SAAM and presented a scenario-based framework to evaluate architectural quality attributes. Lutz and Gannod [33] (S2) combined the scenario analysis for addressing modifiability in the PLA with automated analysis of quality attributes. Avgeriou [4] (S20) proposed a description of a reference architecture based on an IEEE standard and UML. His work introduced an approach that describes reference architecture, instantiates it into a software architecture by making implementation decisions and evaluates it with respect to quality attributes. Babar [5] (S3) combined the relationship between scenarios, quality attributes and architecture patterns for the assessment of PLAs. Olumofin and Misic [39, 40] (S5 and S7), Graaf et al. [24] (S6), Angelov et al. [3] (S22), Kim et al. [30] (S9), Etxeberria and Sagardui [18, 19] (S10 and S11), Reijonen et al. [44] (S12), and Tan and Lian [46] (S16) extended ATAM to consider trade-offs in the variability and quality attributes scenarios of PLAs. OliveiraJr et al. [15, 38] (S17) proposed a systematic method for evaluating PLA, which has a meta-process for trade-off analysis and quality attributes based on scenarios, qualitative, and quantitative evaluation. Barber et al. [6, 7] (S19) proposed a dynamic evaluation tool to assess properties of reference architecture domain using model checking, simulation guides and scenarios. McGregor et al. [35] (S13) combined the use of economic impact and scenarios for analyzing the economic impact of adding variation points in PLAs. Guana and Correal [25] (S15) proposed a model-driven method for the evaluation of quality attributes from PLAs based on evaluation scenarios. Dobrica and Ovaska [16] (S14) proposed a qualitative analysis to address variation points of applications domain through usage scenarios changes.

Benefits: Scenario-based evaluation method supports the identification of risks by analyzing anticipated changes to be made in the software systems and architectures, what in consequence result in suitable mitigation actions introduced before the software system is completed designed. Moreover, another significant benefit is the communication between different stakeholders about the software architecture, increasing the knowledge design and experience among the participants and avoiding misunderstanding and error propagation.

Limitations: Although scenario-based approaches help to identify multiple hypotheses for a particular architecture quality aspect, they are time-consuming and require the perception and previous experience of experts during the creation of scenario and selection

of variants to be further analyzed. It is also expensive and complicated to be applied in big companies, as it is almost impossible for getting all stakeholders in the same meeting for more than two days. In addition, it is a time-consuming method as quality attributes and requirements must be refined into scenarios and further discussed by each stakeholder.

4.2 Questionnaire-based Evaluation

Questionnaire-based evaluation is a set of written questions which aims to extract specific information from stakeholders.

Kim et al. [28, 29] (S4 and S8) defined a framework to assess execution and implementation of PLAs and guidelines to quality attributes using a questionnaire. Such questionnaire assesses five development practices, including layered architecture, variability, user interface, documentation, and abstraction mechanism. Ahmed and Capretz [1] (S18) introduced a process model to assess the PLA maturity based on a list of questions and rating. Such model takes into consideration five maturity levels and six processes, including domain engineering, requirements management, commonality management, variability management, architecture analysis, and architecture artifact management. Current literature also presents assessment methods for reference models to guide systems quality improvement by assessing maturity levels. Zimmermann et al. [48, 49] (S29 and S30) proposed ESARC and SOAMMI based on CMMI and TOGAF to evaluate different service-oriented platforms using a questionnaire-based assessment. Zalewski [47] (S21) proposed an approach based on the Goal-Question-Metric (GQM) framework and scenario-based method aiming to assess systems complex control, organizational adequacy and satisfactory performance, and reliability. Galster et al. [23] (S28) combined a checklist questionnaire with scenario-based evaluation. The checklist is structured based on stakeholders and their interest in the software architecture variability. The scenario-based evaluation focused on adaptation scenarios for agile projects. Santos et al. [45] (S24) proposed a checklist-based approach, known as FERA, for the identification of problems in the architectural description of reference architectures. FERA can be also used either as a standalone approach or complimentary to other architectural evaluation methods such as the scenario-based ones.

Benefits: Questionnaire-based approaches for the evaluation of software systems and architectures provide guidelines for supporting stakeholders during the design and assessment of architectures. They are a flexible and practical way to gather data from stakeholders about specific design issues. Questionnaire can be developed in less time and it is cost-effective as can be conducted remotely preventing geographical dependence. Advanced statistical techniques can be applied to analyze data collected and determine their validity, reliability, and statistical significance.

Limitations: Questionnaire-based evaluation are largely theoretical and stakeholders may not feel encouraged or aware to provide accurate answers. Questionnaire with closed-ended questions may have lower validity rate compared to open questions. In addition, data errors due to non-responses questions or unclear data because of miss-interpretation may exist. A proper questionnaire design is vital to enable analysis of results and accuracy of data analyzed.

4.3 Other Evaluation approaches

Misic [36] (S1) developed an approach to measure cohesion of internal properties of PLAs with the use of formal mathematics definition and, thus, linking the quality of PLA to a higher cohesion. Martínez-Fernández et al. [34] (S23) presented a framework and guidelines to evaluate reference architectures using empirical studies through surveys, including i) Survey to check the value of organizations; and ii) Survey to understand the impact of using a reference architectures. They also propose an evaluation model to calculate the ROI (Return Of an Investment) of the use of reference architecture and a qualitative evaluation to understand the effectiveness of the reference architecture.

4.4 Evaluation-based approaches overview

Table 2 summarizes the assessment methods previously presented. Studies are classified by their extension basis, evaluation criterium, evaluation technique, and industrial participation during the evaluation of each method. Most of proposals (20 studies) address evaluation of quality attributes, as they affect the architecture construction. However, there are other important elements that must be evaluated to ensure quality of reference architectures, such as legal regulation, decisions and rationale, business qualities, stakeholders concerns, and other elements treated by RAModel [37]. Assessment methods proposed in the studies S27, S29, and S30 are concerned to quality and maturity of reference models; however, they do not consider elements usually connected to implementation practices that a reference architecture description must provide. The evaluation framework proposed in S23, S24, S28 are concerned to a more general evaluation based on the analysis of benefits, weakness, and suitability of reference architecture in business context. Studies S13 and S14 propose means for the evaluation of variation points and variants in PLA through scenarios. These approaches consider changes in common components of the architectures of software products and apply variability as possible modifications for each component. Figure 3 presents an overview of the finding results.

5 DISCUSSIONS

Most of studies previously described evaluate variability as a quality attribute in PLAs through scenario-based approaches. PLAs are a core asset of SPL and they assume the existence of an SPL infrastructure to support the identification, modeling, evaluation, and derivation of software products. This is rarely the case for reference architectures and software architectures which must also support variability. The main problem in assessing variability is the number of possible components combination to assemble inside the architecture. In addition, variability is also an interest of a variety of stakeholders affecting multiples concerns and influencing different architecture levels. Given the huge amount of configuration to derive software products, and the variety of quality dimensions to be evaluated through scenarios, stakeholder manual analysis is impractical, error-prone, and a time-consuming task. Even having an amount of evaluation approaches for architectures (as well as for PLAs and reference architectures), none of such approaches concentrate attention on the variability in different architecture abstract levels. This is important when we consider reference architectures and enterprise architectures that not only consider variability in

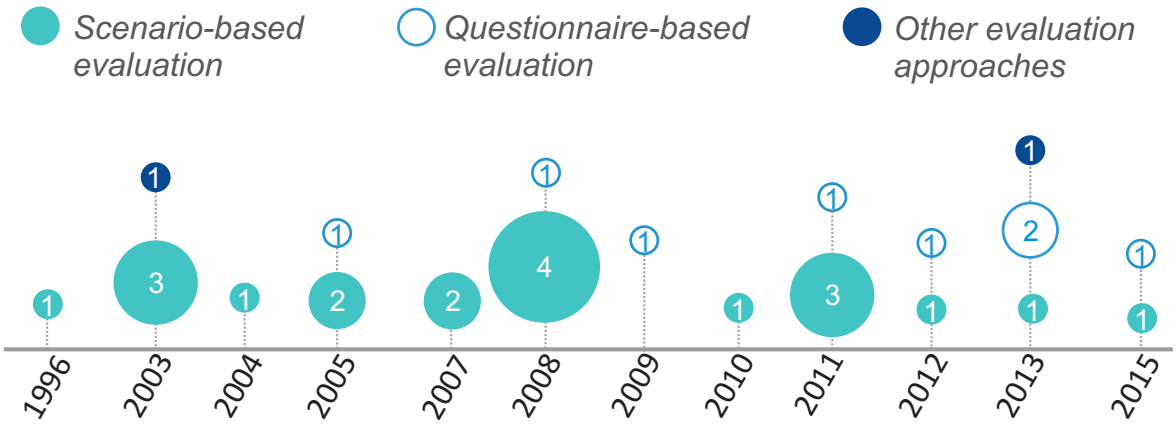

Figure 3: Evaluation-based approaches

Table 2: Methods for software architecture evaluation

Architecture level	ID	Author	Year	Extended from	Evaluation Criterium	Evaluation Technique	Industrial participation
Product Line Architectures	S1	[36]	2003	-	Artifacts	Cohesion measure	
	S2	[33]	2003	SAAM ATAM	Quality attributes	Scenario-based	
	S3	[5]	2004	ATAM	Quality attributes	Scenario-based	
	S4	[29]	2005	-	Overall evaluation	Questionnaire-based	
	S5	[39]	2005	ATAM	Quality attributes	Scenario-based	
	S6	[24]	2005	SAAM	Quality attributes	Scenario-based	Yes
	S7	[40]	2007	ATAM	Quality attributes	Scenario-based	Yes
	S8	[28]	2008	-	Development practices	Questionnaire-based	
	S9	[31]	2008	ATAM	Quality attributes	Scenario-based	
	S10	[19]	2008	ATAM	Quality attributes	Scenario-based	
	S11	[18]	2008	ATAM	Quality attributes	Scenario-based	
	S12	[44]	2010	ATAM	Quality attributes	Scenario-based	Yes
	S13	[35]	2011	SAAM	Variation points and variants	Scenario-based	
	S14	[16]	2011	SAAM	Variation points and variants	Scenario-based	
	S15	[25]	2011	-	Quality attributes	Scenario-based	
	S16	[46]	2012	QADA	Quality attributes	Scenario-based	
	S17	[15] [38]	2013	ATAM	Quality attributes	Scenario-based	
	S18	[1]	2015	BAPO	Process maturity	Questionnaire-based	Yes
Reference Architectures	S19	[6], [7]	2003	RARE	Quality attributes	Scenario-based	Yes
	S20	[4]	2003	Standards	Quality attributes	Scenario-based	
	S21	[47]	2007	ATAM	Quality attributes	Scenario-based	
	S22	[3]	2008	ATAM	Quality attributes	Scenario-based	
	S23	[34]	2013	-	Overall evaluation	Qualitative and Quantitative	Yes
	S24	[45]	2013	-	Overall evaluation	Questionnaire-based	
	S25	[14]	2015	ATAM	Quality attributes	Scenario-based	Yes
Software Architectures	S26	[17]	1996	SAAM	Quality attributes	Scenario-based	
	S27	[9]	2009	-	Quality and Maturity	Questionnaire-based	Yes
	S28	[23]	2013	-	Overall evaluation	Questionnaire-based and scenario-based	Yes
Enterprise Architectures	S29	[48]	2011	TOGAF, SOAMMI, CMMI	Quality and Maturity	Questionnaire-based	
	S30	[49] [20]	2012	SOAMMI, CMMI, TOGAF	Quality and Maturity	Questionnaire-based	Yes

software products but also variability in the organization (i.e., variability in e-government services), economic status (i.e., wealthy e-governments provide more free services), demographic situation (i.e., municipalities with more senior citizens provide more senior

services), variability in the interoperability among different organizations, and variability in legislations (i.e., different countries have their specific laws and regulations). In addition, besides the number of such architectures emerging continually, they have not been still adequately evaluated, including the analysis of variability in their documentation, architecture views, requirements, and stakeholders concerns.

Questionnaire-based approaches (i.e., open questions, checklists, surveys) are used not only for evaluating, but also as a way of learning and supporting the design of software architectures. It is a flexible way of evaluation and can be adapted for specific context, projects and needs. Questionnaire-based approaches are used for deciding whether architectures might be improved, evolved, and to detect defects or problems in the architecture documentation. Regarding the evaluation of variability in architectures, a questionnaire must be elaborated for potential stakeholders interested in the variability and for evaluating each part of the design interaction. However, questionnaire-based approaches rely strongly on individuals participating in the evaluation requested for providing their agreement to each statement. This type of evaluation is largely based on the subjective assessment of each stakeholders who can misunderstand a question and provide wrong or unreliability answers.

Several observations are possible from these SMS analysis.

1. Maturity level evaluation is ignored during architectural assessment: Most studies propose means for assessing quality attributes in PLAs, but few initiatives have been reported evaluation at maturity level in the context of software architectures. The evaluation approaches does not provide guidelines for the improvement architectural process in an organization.

2. Assessment of variability in different architecture levels: Most approaches propose the evaluation of variability in architecture components, software products, and quality requirements in the context of PLAs, but they neglected the assessment of variability in different architecture levels including stakeholders concerns, business process, organizations and market needs within reference and enterprise architectures.

3. The need for combining different evaluation techniques: The combination of scenario-based evaluation methods and questionnaire-based techniques might be an adequate option for covering many evaluation aspects when assessing variability in architectures. Scenarios allows the discussion and analysis of thoughts and ideas about possible changes in the architecture and questionnaires verifies the consistency and completeness of architecture description as well as to its content. These combination may overlap to obtain more complete results and are subjective which means the software architecture is accepted when architects are satisfied with the outcome of scenario and questionnaire evaluations.

4. Constrains may affect the evaluation: Regardless PLAs that encompass one organization for the development of SPL products, reference architectures encompasses different organizations that have their own development infrastructure and domain constraints. These constraints related to stakeholders must have highest priority during architecture evaluation, specially when there is conflict between constraints and evaluation criteria (i.e., constraints that may affect the evaluation).

5. Evaluation might be an obstacle to both stakeholders learning and process improvements: The evaluation of variability in architecture implies an additional training for stakeholders who have to be aware about what may vary in the architecture and what must be critically evaluated. In addition, a strong communication is necessary to support the acquisition of knowledge within the organization. Moreover, organizations must continuously improve their process for the architecture evaluation with innovative methods.

6 THREATS TO VALIDITY

The results obtained by this study might have been affected by some threats. This section presents such threats with regard to:

Conclusion validity

A potential conclusion validity is the confidence and reliability of the selection of studies and data extraction. To minimize this threat, we performed several brainstorming sessions to define the research question and search string for this study.

Construct validity

It is possible some studies included in this SMS might not have the same terms used for achieving the goal of this SMS. However, we feel quite confident about the terms used, as different combination of keywords were tested against Scopus database[8]. After carefully analyzing the results, we confirmed that the outputs from these search strings matched with the one we previously defined in Section 3, which minimizes the chances of misinterpretation. Another construct validity threat might be the authors judgment with respect to the selection of each study. The decisions of inclusion or exclusion of studies were discussed by researchers from this SMS.

External validity

The main idea of an SMS is to gather as much as possible studies available in the literature to avoid bias during the selection. This threat was mitigated with a review protocol proposed by Kitchenham and Charters [32] and Petersen et al. [43]. This protocol was created by two researchers and was further validated by a third researcher with extensive experience in software architecture and variability management. Additionally, we discussed the protocol among all researchers before starting this SMS to avoid potential bias in the study search.

Internal validity

During data extraction, some studies did not provide enough details about the proposed approaches. Consequently, we had to interpret some pieces of information during our analysis. To minimize this threat, we look for other related studies describing the approach with further details to get more reliable information. In the cases

[8]https://www.scopus.com

where we did not find enough information, we discussed extracted data among researchers to avoid ambiguity during our SMS process.

7 CONCLUSION

We conducted an SMS about methods and techniques used for the evaluation of architectures containing variability information. Overall, we found that variability in software architecture is more explored in SPL context through academic researchers by using scenario-based approaches.

Results from our SMS indicate that variability in software architecture lacks more industrial participation and empirical evaluations. Most studies proposed by academia validate the approach through illustrative examples. Such validation adds realism and guides stakeholders during the approach use, but it does not bring concrete indication of the approach effectiveness. Moreover, variability in software architecture lacks a fully support tool capable of handling variability from identification, representation, and evaluation to the derivation of concrete architectures. Thus, to better increase such approaches, we recommend more empirical evaluation with industrial partners aiming to demonstrate the importance of handling variability in software architecture. Results from such evaluations would allow other researchers to improve their approaches to industrial needs.

ACKNOWLEDGMENTS

This work is supported by São Paulo Research Foundation (FAPESP) (Grant: 2016/15634-3, 2016/05919-0, 2018/20882-1, 2017/22237-3, and 2017/06195-9).

REFERENCES

[1] Faheem Ahmed and Luiz Fernando Capretz. 2015. An Architecture Process Maturity Model of Software Product Line Engineering. *CoRR* abs/1507.06901 (2015), 191–207.

[2] Peter Andersen and Andrea Carugati. 2014. Enterprise Architecture Evaluation: a Systematic literature Review. In *8th Mediterranean Conference on Information Systems, MCIS*. AISeL, Verona, Italy, 41.

[3] Samuil Angelov, Jos J. M. Trienekens, and Paul W. P. J. Grefen. 2008. Towards a Method for the Evaluation of Reference Architectures: Experiences from a Case. In *Software Architecture, Second European Conference, ECSA*. Springer, Paphos, Cyprus, 225–240.

[4] Paris Avgeriou. 2003. Describing, instantiating and evaluating a reference architecture: A case study. *Enterprise Architecture Journal* 342 (2003), 1–24.

[5] Muhammad Ali Babar. 2004. Scenarios, Quality Attributes, and Patterns: Capturing and Using their Synergistic Relationships for Product Line Architectures. In *11th Asia-Pacific Software Engineering Conference APSEC*. IEEE Computer Society, Busan, Korea, 574–578.

[6] K. Suzanne Barber, Thomas J. Graser, and Jim Holt. 2003. Evaluating dynamic correctness properties of domain reference architectures. *Journal of Systems and Software* 68, 3 (2003), 217–231.

[7] K. Suzanne Barber, Jim Holt, and Geoff Baker. 2002. Performance evaluation of domain reference architectures. In *14th international conference on Software engineering and knowledge engineering, SEKE*. ACM, Ischia, Italy, 225–232.

[8] Rafael Ferreira Barcelos and Guilherme Horta Travassos. 2006. Evaluation Approaches for Software Architectural Documents: a Systematic Review. In *Memorias de la IX Conferenci a Iberoamericana de Software Engineering (CIbSE 2006)*. La Plata, Argentina, 433–446.

[9] Balbir S. Barn. 2009. On the evaluation of reference models for software engineering practice. In *2nd Annual India Software Engineering Conference, ISEC*. ACM, Pune, India, 111–116.

[10] Rabih Bashroush, Muhammad Garba, Rick Rabiser, Iris Groher, and Goetz Botterweck. 2017. CASE Tool Support for Variability Management in Software Product Lines. *ACM Comput. Surv.* 50, 1 (2017), 14:1–14:45.

[11] J. Bosch, R. Capilla, and R. Hilliard. 2015. Trends in Systems and Software Variability. *IEEE Software* 32, 3 (2015), 44–51.

[12] Rafael Capilla, J. Bosch, and K. C. Kang. 2013. *Systems and Software Variability Management: Concepts, Tools and Experiences*. Springer, Berlin Heidelberg.

[13] Lianping Chen, Muhammad Ali Babar, and Nour Ali. 2009. Variability management in software product lines: a systematic review. In *13th International Conference on Software Product Lines (SPLC)*. ACM, San Francisco, California, USA, 81–90.

[14] Simone da Silva Amorim, John D. McGregor, Eduardo Santana de Almeida, and Christina von Flach G. Chavez. 2015. Tailoring the ATAM for Software Ecosystems. In *9th European Conference on Software Architecture, ECSA*. Springer, Dubrovnik/Cavtat, Croatia, 372–380.

[15] Edson Alves de Oliveira Junior, Itana Maria de Souza Gimenes, and José Carlos Maldonado. 2011. A Meta-Process to Support Trade-Off Analysis in Software Product Line Architecture. In *23rd International Conference on Software Engineering & Knowledge Engineering (SEKE)*. Knowledge Systems Institute Graduate School, Miami Beach, USA, 687–692.

[16] Liliana Dobrica and Eila Ovaska. 2011. Analysis of a cross-domain reference architecture using change scenarios. In *5th European Conference on Software Architecture, ECSA*. ACM, Essen, Germany, 10.

[17] Nancy S. Eickelmann and Debra J. Richardson. 1996. An Evaluation of Software Test Environment Architectures. In *18th International Conference on Software Engineering, ICSE*. IEEE Computer Society, Berlin, Germany, 353–364.

[18] L. Etxeberria and G. Sagardui. 2008. Quality assessment in software product lines. In *10th international conference on Software Reuse: High Confidence Software Reuse in Large Systems (ICSR)*. Beijing, China, 178–181.

[19] L. Etxeberria and G. Sagardui. 2008. Variability driven quality evaluation in software product lines. In *12th International Software Product Line Conference (SPLC 2008)*. Limerick, Ireland, 243–252.

[20] Michael Falkenthal, Dierk Jugel, Alfred Zimmermann, René Reiners, Wilfried Reimann, and Michael Pretz. 2012. Maturity Assessments of Service-oriented Enterprise Architectures with Iterative Pattern Refinement. In *Jahrestagung der Gesellschaft für Informatik e.V. (GI)*. GI, Braunschweig, Deutschland, 1095–1101.

[21] Matthias Galster. 2015. Architecting for Variability in Quality Attributes of Software Systems. In *2015 European Conference on Software Architecture*. ACM, Dubrovnik/Cavtat, Croatia, 23:1–23:4.

[22] Matthias Galster and Paris Avgeriou. 2011. Handling Variability in Software Architecture: Problems and Implications. In *9th Working IEEE/IFIP Conference on Software Architecture (WICSA)*. IEEE Computer Society, Colorado, USA, 171–180.

[23] Matthias Galster and Paris Avgeriou. 2013. Supporting Variability Through Agility to Achieve Adaptable Architectures. In *Agile Software Architectures–Aligning Agile Processes and Softare Architectures*. Vol. 1. Elsevier, Waltham, Massachusetts, USA, 139–159.

[24] Bas Graaf, Hylke W. van Dijk, and Arie van Deursen. 2005. Evaluating an Embedded Software Reference Architecture – Industrial Experience Report. In *9th European Conference on Software Maintenance and Reengineering CSMR*. IEEE Computer Society, Manchester, UK, 354–363.

[25] V. Guana and D. Correal. 2011. Variability quality evaluation on component-based software product lines. In *15th International Software Product Line Conference (SPLC 2011)*. Munich, Germany, 19:1–19:8.

[26] R. Kazman, L. Bass, G. Abowd, and M. Webb. 1994. SAAM: a method for analyzing the properties of software architectures. In *16th International Conference on Software Engineering (ICSE 1994)*. Sorrento, Italy, 81–90.

[27] R. Kazman, M. Klein, M. Barbacci, T. Longstaff, H. Lipson, and J. Carriere. 1998. The architecture tradeoff analysis method. In *4th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 1998)*. Monterey, CA, USA, 68–78.

[28] Kangtae Kim, Hyungrok Kim, Sundeok Kim, and Gihun Chang. 2008. A Case Study on SW Product Line Architecture Evaluation: Experience in the Consumer Electronics Domain. In *3rd International Conference on Software Engineering Advances, ICSEA*. IEEE Computer Society, Sliema, Malta, 192–197.

[29] Soo Dong Kim, Soo Ho Chang, and Hyun Jung La. 2005. A Systematic Process to Design Product Line Architecture. In *International Conference Computational Science and Its Applications (ICCSA)*. Springer, Singapore, 46–56.

[30] Taeho Kim, In-Young Ko, Sungwon Kang, and Danhyung Lee. 2008. Extending ATAM to assess product line architecture. In *8th IEEE International Conference on Computer and Information Technology, CIT*. IEEE Computer Society, Sydney, Australia, 790–797.

[31] T. Kim, I.-Y. Ko, S.-W. Kang, and D.-H. Lee. 2008. Extending ATAM to assess product line architecture. In *IEEE 8th International Conference on Computer and Information Technology (CIT 2008)*. Sydney, NSW, Australia, 790–797.

[32] Barbara Kitchenham and Stuart Charters. 2007. *Guidelines for performing Systematic Literature Reviews in Software Engineering*. Technical Report. Keele University and Durham University Joint Report. 57 pages.

[33] Robyn R. Lutz and Gerald C. Gannod. 2003. Analysis of a software product line architecture: an experience report. *Journal of Systems and Software* 66, 3 (2003), 253–267.

[34] Silverio Juan Martínez Fernández, Claudia Patricia Ayala Martínez, Javier Franch Gutiérrez, Helena Martins Marques, and David Ameller. 2013. A framework for software reference architecture analysis and review. In *Memorias del X Workshop Latinoamericano Ingeniería de Software Experimental, ESELAW*. Montevideo, Uruguay, 89–102.

[35] John D. McGregor, J. Yates Monteith, and Jie Zhang. 2011. Quantifying value in software product line design. In *15th International Conference on Software Product Lines, SPLC*. ACM, Munich, Germany, 40.

[36] Vojislav B. Misic. 2003. Measuring the Coherence of Software Product Line Architectures. In *International Conference on Software Engineering Research and Practice, SERP*. CSREA Press, Las Vegas, Nevada, USA, 364–372.

[37] Elisa Yumi Nakagawa, Flávio Oquendo, and Martin Becker. 2012. RAModel: A Reference Model for Reference Architectures. In *Joint Working IEEE/IFIP Conference on Software Architecture and European Conference on Software Architecture (WICSA/ECSA)*. Helsinki, Finland, 297–301.

[38] E.A. Oliveira Junior, I.M.S. Gimenes, J.C. Maldonado, P.C. Masiero, and L. Barroca. 2013. Systematic evaluation of software product line architectures. *Journal of Universal Computer Science* 19, 1 (2013), 25–52.

[39] Femi G. Olumofin and Vojislav B. Misic. 2005. Extending the ATAM Architecture Evaluation to Product Line Architectures. In *5th Working IEEE / IFIP Conference on Software Architecture (WICSA)*. IEEE Computer Society, Pittsburgh, Pennsylvania, USA, 45–56.

[40] Femi G. Olumofin and Vojislav B. Misic. 2007. A holistic architecture assessment method for software product lines. *Information & Software Technology* 49, 4 (2007), 309–323.

[41] Sofia Ouhbi. 2018. Software Architecture Evaluation: A Systematic Mapping Study. In *13th International Conference on Evaluation of Novel Approaches to Software Engineering, ENASE*. SciTePress, Funchal, Madeira, Portugal, 447–454.

[42] Juliana Alves Pereira, Kattiana Constantino, and Eduardo Figueiredo. 2015. A Systematic Literature Review of Software Product Line Management Tools. In *14th International Conference on Software Reuse for Dynamic Systems in the Cloud and Beyond (ICSR)*. Springer International Publishing, Miami, FL, USA, 73–89.

[43] Kai Petersen, Sairam Vakkalanka, and Ludwik Kuzniarz. 2015. Guidelines for conducting systematic mapping studies in software engineering: An update. *Information & Software Technology* 64, C (2015), 1–18.

[44] Ville Reijonen, Johannes Koskinen, and Ilkka J. Haikala. 2010. Experiences from Scenario-Based Architecture Evaluations with ATAM. In *4th European Conference Software Architecture, ECSA*. Springer, Copenhagen, Denmark, 214–229.

[45] J.F.M. Santos, Milena Guessi, M Galster, Daniel Feitosa, and Elisa Nakagawa. 2013. A checklist for evaluation of reference architectures of embedded systems. In *International Conference on Software Engineering and Knowledge Engineering (SEKE 2013)*, Vol. 2013. Boston, USA, 451–454.

[46] L. Tan, Y. Lin, and H. Ye. 2012. Modeling Quality Attributes in Software Product Line Architecture. In *2012 Spring Congress on Engineering and Technology*. IEEE, Xian, China, 1–5.

[47] Andrzej Zalewski. 2007. Beyond ATAM: Architecture Analysis in the Development of Large Scale Software Systems. In *1st European Conference on Software Architecture, ECSA*. Springer, Aranjuez, Spain, 92–105.

[48] Alfred Zimmermann, Helge Buckow, Hans-Jürgen Groß, Oliver F. Nandico, Gunther Piller, and Karl Prott. 2011. Capability Diagnostics of Enterprise Service Architectures Using a Dedicated Software Architecture Reference Model. In *IEEE International Conference on Services Computing, SCC*. IEEE Computer Society, Washington, DC, USA, 592–599.

[49] Alfred Zimmermann and G Zimmermann. 2011. Esarc-enterprise services architecture reference cube for capability assessments of service-oriented systems. *SERVICE COMPUTATION* 2011 (2011), 63–68.