March - April 2020
ISSN: 0193-4120 Page No. 293 - 298

Software Architecture Evaluation Methods for Measuring Modifiability

[1] Fatemeh Meskaran, [2] Maryam Shahpasand, [3] Maryam Var Naseri, [4] Chandra Reka Ramachandiran
[1] Asia Pacific University ,[2] Asia Pacific University,[3] Asia Pacific University
[1]fatemeh.meskaran@staffemail.apu.edu.my ,[2] maryam.shahpasand@staffemail.apu.edu.my,
[3] maryam.var@staffemail.apu.edu.my, [4] chandra.reka@staffemail.apu.edu.my

Article Info
Volume 83
Page Number: 293 - 298
Publication Issue:
March - April 2020

Article History
***Article Received**: 24 July 2019*
***Revised**: 12 September 2019*
***Accepted**: 15 February 2020*
***Publication**: 12 March 2020*

Abstract
Software systems constantly change, and it causes the architecture of the system to degenerate during the system life cycle. Definitely, any degeneration needs extra effort and delays the releases of the system. The ability to accept changes quickly and cost-effectively is considered as modifiability. In order to assess system modifiability, we need evaluation methods and tools. There are different evaluation methods for software architecture quality evaluation. In this paper, we compare two well-known software architecture evaluation methods which mostly are applying for testing the modifiability, including Scenario-Based Architecture Analysis (SAAM) and Architecture Level Modifiability Analysis (ALMA). The comparison shows that these two methods are structurally similar however, there are some differences among their activities and processes. Therefore, the common activities that are used for evaluating software architecture in these two methods can form a generic process model for evaluating modifiability.

***Keywords:** Software architecture, Software architecture evaluation, Modifiability, SAAM, ALMA.*

I. INTRODUCTION

Software architecture shows the total structure of a system. It is the blueprint of the software system. Creating a suitable software architecture has different challenges because of increasing size, complexity and demand for high-quality software systems. It is identified that system quality (such as modifiability, maintainability, performance, security, etc.) are mostly constrained by software architecture [1]. Therefore, software architecture should address the related quality attributes, their features, and possible risks. Evaluation of software architecture in the early stages is considered a crucial task during the software development process. The main reason for software architecture evaluation is to assess whether the system requirements and quality attributes are met by software architecture. In addition, it is the architecture capable to identify risks and manage them [2].

It is shown that 50% to 70% of the total cost for a software system development belongs to the evolution of the system [3]. If during the architecture design, modifiability capability is considered, the evolution cost will be reduced. Modifiability capability should be considered during architecture design. therefore

The architecture design has an important effect on the acceptance of modifiability. There are different methods have been proposed to evaluate software architecture. These methods can be scenario-based, experienced-based and mathematical-based [3]. Most of these methods are complementary techniques from other methods, so sometimes distinctive among these methods is difficult. There are some researches regarding compare different software architecture evaluation methods, especially scenario-based methods [1], [3]-[5]. These researches mostly focus on general review regarding improving the understanding of evaluation methods, and not focusing on the specific quality attribute. This study attempts to identify the most common software architecture evaluation methods which are using for evaluating modifiability. Two commonly used software architecture evaluation methods including Scenario-Based Architecture Analysis (SAAM) [6], and Architecture Level Modifiability Analysis ALMA [7] are selected for more elaboration. The purpose of this research is to compare these two evaluation methods as mostly use in modifiability examination at an architecture level. This work can help practitioners and researchers to understand and contrast alternative approaches that are available to them to evaluate software architecture, especially those they want to test the modifiability.

The rest of the paper is structured as follows: Section II discusses the background work, including a review of all possible evaluation methods. Section III presents different software architected evaluation methods. The next two sections present and discuss two common methods of modifiability evaluation along with the rationale for selecting its components finally a comparison of the two software architecture evaluation methods are presented in the last section.

II. Background Work

There are some overview researches regarding software architecture evaluation methods [1], [4], [5]. overview reported in [8] provides complete guidance on software architecture evaluation methods. In addition, there are few other research [9],[10] to provide a complete explanation of comparing the scenario-based evaluation techniques. None of the other published surveys or comparison of software architecture testing methods provides an explicit framework for comparing the methods regarding modifiability. Rather, these surveys have been published to support the need for developing a new evaluation method.

Clements et al. [10] have a chapter on evaluation technique comparison in software architecture evaluation. However, only three evaluation methods including (SAAM, ATAM, and ARID are compared. In addition, scenario-based software architecture evaluation techniques are compared in another study by Babar [4]. However, both are limited in categorization based on quality attributes. Software architecture evaluation is examining an architecture against the quality goals. The main purpose of architecture evaluation is to test and validate the software architecture and ensure that the architecture is able to satisfy quality attributes goals [11]. There are different software architecture evaluation techniques that each of them may be useful for evaluating a specific quality attribute. In this paper, we focus on Modifiability as one of the most important quality attributes, which most consider by software architecture evaluators.

III. Software Architecture EVALUATION METHODS

Evaluation of software architecture can be done in any stage of architecture design. Most of the techniques are applied after architecture design specification, and before implementation. for iterative or incremental methodologies the architectural decisions are specified at the end of each iteration [12]. Architectural evaluation is an activity related to human actions. Most of the time, the reviews are the main stakeholders such as clients, designers, and the evaluation team. Therefore it can be done by experiments, modeling, and evaluating scenarios, or even it can be done by experts who look for gaps and possible risks in the architecture by using their experience. In addition, analytic models, simulation tools can be used for software architecture evaluation, those targeting a single quality goal (e.g. performance or modifiability), or even several quality attributes.

Architecture evaluation techniques are categorized into questioning, measuring techniques, and hybrid [13]. Scenarios, questionnaires, and checklists are using in questioning techniques. On the other hand, metrics, simulation, or experimentations are applying in measuring techniques. In addition, Hybrid combines both questioning and measuring together and make a new technique. Most of the architecture evaluation methods are commonly hybrid; during the elicitation mostly we use questioning and then use sorts of measurements for reasoning.

The quality attributes are non-functional requirements such as modifiability, scalability, performance, security, and availability. Each software evaluation method has several objectives that are implemented at different phases in software development life-cycle or may concentrate on different quality attributes[14]. Research works are being processed regarding software architecture evaluation methods [4],[10]. Architectural evaluation can be accompanied based on the specification of the software architecture. The scenario-based techniques are flexible and simple [4],[15]. However, mathematical evaluation methods based on modeling are well used for evaluating the quality attributes, such as reliability and performance too. These methods especially are used in real-time software systems. This paper distinguishes the evaluation methods which are scenario-based and then identifies the most common architectural evaluation methods that are using for evaluating software architecture regarding modifiability.

Scenario-based software evaluation techniques examine software architecture's capacity considering a set of related scenarios. There are different scenario-based evaluation techniques [4] , [6], [10], [16], [17]. The scenario-based evaluation techniques propose an organized approach to evaluating software architecture by using scenarios. If the software architecture can not execute the scenario, the evaluation techniques list all required modifications regarding supporting the scenario and estimate the cost of applying the modifications. In scenario-based evaluation techniques, significant stakeholders should be presented and elicit scenarios based on requirements. In Table 1, some of the Scenario-based evaluation techniques are presented, the most considerable quality attribute is indicated in the last column:

Table 1. Scenario-based evaluation techniques

Evaluation Technique Name	References	Quality attributes
SAAM (Scenario-based Software Architecture Analysis Method)	[1], [4], [6],[17]	Modifiability
ATAM (Architecture based Tradeoff Analysis Method)	[1], [18], [19]	Multiple Quality Attributes
ALMA (Architecture-Level Modifiability Analysis)	[3], [20]	Modifiability
CBAM (Cost-Benefit Analysis Method)	[1], [21]	Costs, Benefits
FAAM (Family-Architecture Assessment Method)	[22]	Extensibility
SALUTA (Scenario-based Architecture Level UsabiliTy Analysis)	[23]	Usability
SBAR (Scenario-Based Architecture Reengineering)	[24]	Multiple Quality Attributes

In this paper, two main techniques that are suitable for evaluating modifiability including ALMA and SAAM are compared.

IV. Scenario-based Architecture analysis Method

A. Scenario-Based Architecture Analysis Method(SAAM)

is the first scenario-based software architecture analysis technique. SAAM can verify architectural values against documents that describe the characteristics of a system. The quality attributes especially those that are related to modifiability can be expressed during the SAAM application.

Problem identification in the early stages and having the comprehensive documentation are the main advantages of SAAM [6], this technique can be used regarding compare multiple software architectures.

According to SAAM has six phases: scenario development, software architecture description, scenario classification and prioritization, individual scenario evaluation, scenario interaction, and overall evaluation. The first two phases are iterative [6]. Figure 2 shows the different phases in SAAM.

As one of the scenario-based software architecture evaluation techniques, SAAM assesses each scenario by mapping it onto software architecture design and check whether the software architecture can support this scenario without any modification (direct scenario) or it needs some modification (indirect scenario). The cost including the needed time and effort for accommodating each indirect scenario is estimated by counting the number of required changes and the number of affected components. Scenario interaction analysis reveals if many indirect scenarios affect the same component, a sign of lack of encapsulation or ignoring the loose coupling principle.

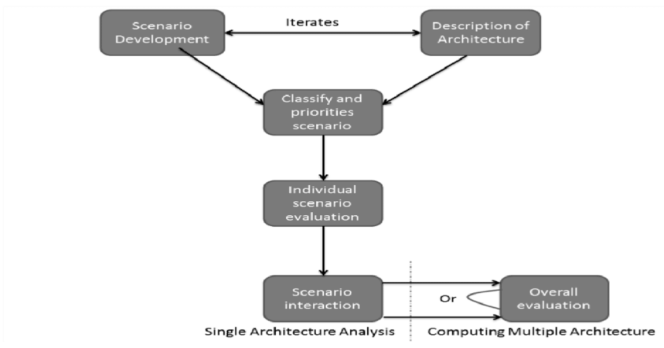
Scenario Development
Iterates
Description of Architecture
Classify and priorities scenario
Individual scenario evaluation
Scenario interaction
Or
Overall evaluation
Single Architecture Analysis
Computing Multiple Architecture

Figure 2. SAAM Process

V. Architecture Level Modifiability Analysis

Architecture Level Modifiability Analysis (ALMA) is focusing on adaption and modifiability [20], [24]. ALMA is considered as a goal-oriented evaluation technique. After goal setting, most of the activities in this approach are dealing with evaluation of the goals. The main goal of this technique is related to modifiability.

The goal of ALMA is to deliver a structured method to evaluate the three major aspects of mantainability, which includes the risk assessment, software architecture comparison and maintenance prediction. The method describes five major phases, namely to describe the software architecture, to determine the evaluation goal and elicit of relevant and evaluation scenarios, and finally results are interpretated and conclusions are drawn from them [3]. Figure 2 shows the related process of ALMA. The techniques are applied to select relevant scenarios and reduce the number of scenarios [20]. It also manages when to stop generating scenarios. ALMA The ALMA uses impact analysis to evaluate software architecture regarding modifiability. Regarding the modification, the evaluation technique identifies affected components, and how much effort it needs. The results are construed depending on the goal of evaluation. ALMA offers a framework to describe results quantitatively. As ALMA has been validated with several applications, the method is considered quite mature.

As it is mentioned, the specific goal of ALMA is to address modifiability. ALMA is usually applied before implementing the software architecture but it is still useful for the legacy systems. ALMA has successfully been useful in telecommunications, information systems, and medical domains [24].

The main benefits of using ALMA are the identification of software architecture risks, estimation of the efforts required to accommodate the changes, or selection of optimal software architecture [20].

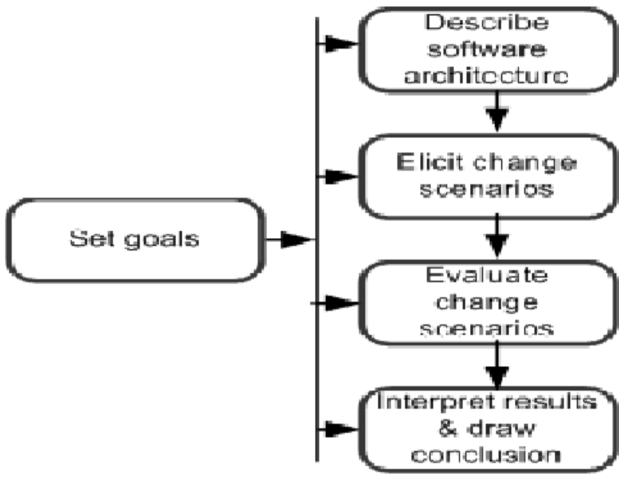
Set goals
Describe software architecture
Elicit change scenarios
Evaluate change scenarios
Interpret results & draw conclusion

Figure 2. ALMA Process

VI. Method Comparison

In scenario-based methods, there are many activities that remains to be the same at the top pevel; however, if it is gone through in detail into the activities, there are a number of differences being revealed. For example, in the scenario based methods, scenario development and evaluation activities are found be so common, however, the techniques used to perform these activities are entirely different. ALMA uses scenario profiles to categorize the generated scenarios [20], and software architecture is also documented using various views [1], [4]. Both of SAAM and ALMA use software architecture views, however, the number and type of views vary for them. For example, logical and module views suit both SAAM and ALMA, however, SAAM is purely scenario-based, ALMA uses a variety of methods related evaluation goals. ALMA also provides analytical models for modifiability, while SAAM uses those provided analytical models [25]. In addition, we can compare these two techniques regarding the type of involved stakeholders. SAAM works with all major stakeholders however ALMA deals with a small set of stockholders such as architects or developers [20].

Generally, we can say AlMA can be used for the purpose of risk evaluation, however, SAAM can be applied for finding out the complexity and interactions in software architecture.

On the other hand, both the evaluating techniques do recognize the importance of suitable tool support, but, only SAAM provides a tool (SAAMTOOL) [21] to support the evaluation process partially. Further, another feature of automation is the management of knowledge for the purpose of reusability, which is considered to be one of the most important way to increase the quality, productivity as well as the cost-effectiveness [23]. None of these evaluation techniques provides guidance on generating and utilizing the reusable artifacts, i.e., identified risks, scenarios, quality attributes etc [13].

Software architecture evaluation methods can also be compared to the aspect of maturity as it may raise confidence in method users. It is to be concluded that these two software architectures can be so classified within one of the four maturity phases of the software architecture lifecycle, especially inception, dormant and development refinement [2]. ALMA and SAAM are considered in the refinement and development stage, respectively. The method development process and techniques employed in order to validate it, may lead to encourage or discourage the evalautors, so that they may select any one method as compared to the other methods [24]. Methods reviewed are validated in various several domains. The summary of the comparison between ALMA and SAAM is presented in Table 3.

Table 2. Summary Table

ALMA	SAAM
Goal-Oriented	Scenario-Based
Risk Assessment	Find out complexity and interactions
Estimate cost based on goal	Estimate cost for indirect scenarios
Small set of stockholders (Architect- developers)	All major stockholders
Can not use in legacy systems	Can be use in legacy systems
Apply in the refinement stage	Apply in the development stage

REFERENCES

[1] P. Clements, R. Kazman, and M. Klein, “Evaluating Software Architectures: Methods and Case Studies”, Addison-Wesley, 2002.

[2] C.-H. Lung and K. Kalaichelvan, "An Approach to Quantitative Software Architecture Sensitivity Analysis," International Journal of Software Engineering and Knowledge Engineering, vol. 10, no. 1, pp. 97-114, 2000.

[3] P. Bengtsson, "Towards Maintainability Metrics on Software Architecture: An Adaptation of Object-Oriented Metrics," First Nordic Workshop on Software Architecture, Ronneby, 1998.

[4] M. A Babar and I. Gorton, “Comparison of Scenario-Based Software Architecture Evaluation Methods”, *Asia-Pacific Software Engineering Conference, APSEC.* 2004.

[5] A. Athar, R. M. Liaqat and F. Azam, “A Comparative Analysis of Software Architecture Evaluation Methods”, *Journal of Software,* vol. 11, no.9. 2016.

[6] R. Kazman, L. Bass, G. Abowd, and M. Webb, "SAAM: A Method for Analyzing the Properties of Software Architectures," Proceedings of the 16th International Conference on Software Engineering, 1994.

[7] P. Bengtsson, N. Lassing, J. Bosch, and H. V. Vliet, "Architecture-Level Modifiability

Analysis," Journal of Systems and Software, vol. 69, 2004.

[8] N. Lassing, D. Rijsenbrij, and H. v. Vliet, "The goal of software architecture analysis: Confidence building or risk assessment," Proceedings of First BeNeLux conference on software architecture, 1999.

[9] C.-H. Lung, S. Bot, k. Kalaichelvan, and R. Kazman, "An Approach to Software Architecture Analysis for Evolution and Reusability," Proceedings of CASCON, 1997.

[10] P. C. Clements, "Active Reviews for Intermediate Designs," SEI, Carnegie Mellon University CMU/SEI-2000-TN-009, 2000.

[11] D.L. Parnas. Software Aging. In: I6th International Conference on Software Engineering, Sorento, Italy, pp. 279-287.1994.

[12] C.D Rosso. Continuous evolution through software architecture evaluation: a case study. *Journal of Software Maintenance and Evolution: Research and Practice. Vol.18, pp. 351-383, 2006*

[13] S. Angelov, Patrick de Beer, Software Architecture, vol. 9278, pp. 157, 2015.

[14] B. Eilouti. *Architectural design process automation.* Applications of Informatics and Cybernetics in Science and Engineering, Orlando, Florida, USA (2015), pp. 370-375. 2015

[15] M. A. Babar, L. Zhu and R. Jeffery. A Framework for Classifying and Comparing Software Architecture Evaluation Methods. In the Proceedings on Australian Software engineering, pp. 309-318, 2004.

[16] H. Cervantes, R. KazmanDesigning Software Architectures: A Practical Approach
Addison-Wesley, Bostonm .2016.

[17] H. Cervantes, R. KazmanDesigning Software Architectures: A Practical Approach. Addison-Wesley, Bostonm.2016.

[18] Reijonen, V., Koskinen, J., and Haikala, I.. Experiences from scenario-based architecture evaluations with atam. In European Conference on Software Architecture, pages 214–229. Springer.2010.

[19] Zalewski, A. and Kijas, S. Beyond atam: Early architecture evaluation method for large-scale distributed systems. Journal of Systems and Software, vol. 86, no.3, pp.683–697. 2013.

[20] P. Bengtsson, N. Lassing, J. Bosch, and H. V. Vliet. Architecture-Level Modifiability Analysis. Journal of Systems and Software, vol. 69, 2004

[21] "CBAM: Cost Benefit Analysis Method http://www.sei.cmu.edu/ata/products_services/ cbam.html

[22] A. Alkussayer and W. H. Allen, "A scenario-based framework for the security evaluation of software architecture,". 3rd International Conference on Computer Science and Information Technology, Chengdu, 2010, pp. 687-695.2010.

[23] E. Folmer, J. Gurp and J. Bosch. Software Architecture Analysis of Usability. In the Proceedings on 9th IFIP Working Conference on Engineering Human Computer Interaction and Interactive Systems, pp. 321-339, 2004.

[24] Bengtsson, PO., Lassing, N., and Bosch, J., "Architecture Level Modifiability Analysis (ALMA)," Journal of Systems and Software, vol. 69, pp. 129-147, 2002.

[25] M. Svahnberg, C. Wohlin, L. Lundberg, and M. Mattsson. A Method for Understanding Quality Attributes in Software Architecture Structures. In Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering, 2002.