A Comparative Analysis of Software Architecture Evaluation Methods

Ali Athar*, Rao Muzamal Liaqat, Farooque Azam
Department of Computer Engineering, College of EME, National University of Sciences and Technology (NUST), H-12, Islamabad, Pakistan.

* Corresponding author. Tel: +923004320647; email: Ali.athar1401@gmail.com
Manuscript submitted December 31, 2015; accepted January 22, 2016.
doi: 10.17706/jsw.11.9.934-942

Abstract: Software engineering society has suggested many methods of software architecture to enhance the preferred quality with concern of performance, reliability, maintainability, usability etc. However little effort is required for systematic comparison of these methods to mind variances and resemblances between existing methods. By using an evaluation framework, we will compare four famous software architecture evaluation approaches. This framework considers each software architecture evaluation approach from the context structure, method context, performance and different stakeholders. The systematic comparisons shows the limitation and narrowness of existing methods that are anatomically similar but there is a huge difference with respect to activities and methods, however, some methods overlap which help us to identify five common activities that can be used to develop a generic process model. With the help of this comparative study we can understand contribution in SA evaluation and their limitation; finally we proposed five activities and their flow work in this comparative study.

Key words: SAAM (scenario based architecture analysis) ALMA (architecture level modifiability analysis), ATAM (trade-off analysis method) NIMBSAD (normative information model based system analysis and design).

1. Introduction

Software architecture has a great influence on system performance as well as maintainability. Several techniques have been introduced to improve the quality of system through the software architecture evolution. A category of evolution method is quite mature technique among them. However there are some quantitative models and methods for software architecture evaluation but these models are quite important for scenario based methods.

Owing to emerging terminology and concepts in existing system it is difficult to evaluate the similarities and differences among the existing methods. Therefore a little contribution to understand the systematic evolution and comparative analysis of methods with respect to desired features. This systematic comparison of software architecture can help us to understand the methods, further we can easily imposed these methods during software architecture design. User can also get help to explore their research potential. We have considered the SAAM (Scenario Based Architecture Analysis), ALMA (Architecture Level Modifiability Analysis) and ATAM (Trade-off Analysis Method) methods for this study. The importance and selection of these particulars methods are briefly mentioned in Section 2.

There are two basic aims of this comparative analysis on software architecture evaluation methods

1) To enhance our work by developing a classification method and framework comparison.
2) A progressive technique in scenario based software architecture evaluation method and future work.

This work will be very helpful for researchers to understand and compare the alternatives approaches in this way they can easily evaluate the software architecture. Basically our focus is neither on rigorous software architecture approaches nor on to build a selection tool for appropriate methods. Conversely we believe our contribution will deliver an appropriate guidance for the selection of method and can provide a base for method selection tool.

2. Related Work

For the comparative analysis our work starts to find out all the contribution of researchers and practitioners in this field. There is no comparative paper on SA evolution methods or framework. This survey provides the support for developing the new evolution method. Bahsoon and Emmerich focused for the developing of new software evolution method in their work on Arch Options [1]. Clements has work on comparison method. In his comparison he consider three method SAAM, ARID and ATAM all these methods are urbanised by SEI (Software engineering institute) [2]. This evaluation does not incorporate the attributes that a method should have such as software architecture definition, tool support, maintenance and performance etc. we consider [3], [4] to classify the framework in software architecture evolution method, although both have narrow scope. For example author of [5] doesn't provide explanation about components during framework comparison, either he have no explanation why he is adding that particular component during framework. ATAM SA evaluation method reflects the traditional attributes such as stability and usability.

Ali Babar & M. et al. in their work, Framework Classification and Comparison of Software Architecture Evaluation Methods, has proposed a reliable framework for software architecture evaluation method. We have improved the comparison with some adjustment and additional features. However [4] does not provide the framework comparison.

We also suppress many methods that were documented in literature review, these method never contribute in software architecture evaluation. We incorporate a recently developed method to deal with performance as well as efficiency. We select the method on the basis of continuous development, Performance and reliability.

3. A Comparison Frame

By using a comparative framework we will make a comparison on software Architecture evaluation techniques as shown in the table 1 given below. This effort helps us in designing a trustworthy tool that will provide some supervision in choosing a particular method selection.

We have presented three important constituent and organized these each constituent within four components of SA framework.

Normative Information Model based System Analysis and Design (NIMSAD) is a proposed evaluation framework which helps us to understand the different activities during SA evolution method. Purposed method consists of four module/components which are given below:

1) Method Framework
2) Method of Users
3) Contents of method
4) Verification & validation of methods.

According to our domain we have explore two of components, name and artefacts/elements. Although the different elements in last component incorporate to increase the satisfaction of method in this way the

consistency of system also increased. According to our observation, most of the elements incorporate to evaluate the framework. NIMSAD help us in comparative analysis though we can increase the consistency and performance in framework as well as SA evolution method. NIMSAD provide the comparison in narrow scope so this framework can be easily improvised.

Table 1. The Evaluation Questions and the Components and Features of the Framework.

Components	Elements	Explanation
Context	Software architecture Description	Does selected method clearly define SA?
	Attributes of Quality	Total no of Attribute? Supported Attribute in the method?
	Specific goal	What is the specific goal and objective of the methods?
	No of Applied Stages	Selection criteria to apply the stage in method?
	Application domain	Whatever the application domain(s) the method is frequently pragmatic?
	Identification of Input and Out.	Which important input and output should be considered?
Contents	Activities among the method	Consideration of activities to achieve a specific goal?
	Evaluation approaches	Which sorts of evaluation practises are used by the method?
	Description of SA	Recommended and explanation of selected software architecture?
	Supporting Tool	Familiarization of tools or experience to support the particular method?
Stakeholder	Advantages	Basic advantages and benefits of the method to the investors?
	Support among the process	How support is provided to incorporate several activities during process?
	Shareholders/ Investors	Contribution of different stakeholders in evaluation process?
	Resources	No of the days for the completion of process? How many team members are required?
	Organizational issues	How we can tackle different non-technical issues?
Reliability	Maturity level of method	Understanding of maturity level with the consideration of development, inspection as well as refinement?
	Validation among the methods	Authentication of selected method? Criteria for validation / How validation is carried out?

4. Overview of SA Evaluation Methods

SAAM was proposed first time in 2005. Basic goal of SAAM was to evaluate the software architecture and enhance the quality by improvising the SA evolution method. At that time SAAM aim was modifiability now it is being used for different quality attributes. SAAM is applied in design specification before the development phase. Now we state which input and output components are incorporate in SAAM. Software architecture, business drivers and requirement are comes in input. Scenario related to quality is the part of output in SAAM. Artefacts of SAAM are used in various domains such as case tool and armed combat.
SAAM provides us many benefits which are listed below

1) We can detect the problem in the early phase
2) We can improve the documentation and enhance our understanding related to software architecture problems
3) It relates the different people such as stake holder, architect, maintainer and developer.
4) It provides the various techniques to improve the quality and classify the different scenarios.

SAAM also explain the six activates, these activates are inter related and help to form an effective SA evolution method. These activates are listed below.

1) Development of scenario
2) Software architecture description
3) Classification of scenario and prioritization
4) Evaluation of individual scenario
5) Interaction of scenario
6) Overall evaluation

First two activities execute in parallel and overall interaction among activities is shown by fig. 1.

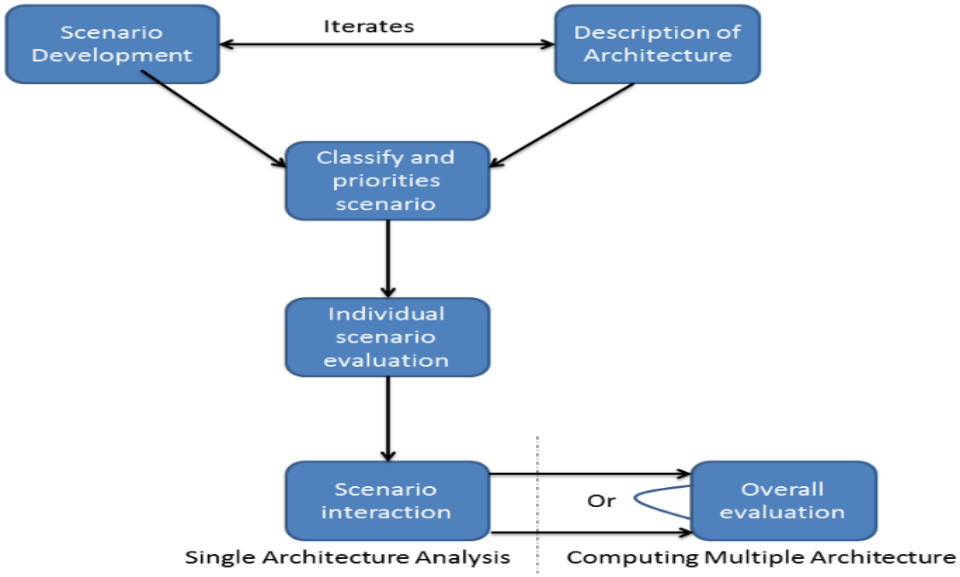

Fig. 1. Process model of SAAM.

SAAM evaluates each scenario by mapping it onto SA description and investigating whether the SA supports it (direct scenario) or not (indirect scenario). The cost of accommodating each indirect scenario is estimated by counting the number of required changes. Scenario interaction analysis reveals if many indirect scenarios affect the same component, a sign of poor separation of concern. SAAM is a mature approach, which has been validated with different case studies. Recently, SAAM has been superseded by ATAM.

4.1. Architecture Level Modifiability Analysis

Bengtson and Lassing developed a conceptual framework which was based on changing of SA resulted in Architecture Level Modifiability Analysis (ALMA). ALMA based on goal-oriented evaluation. After goal setting, remaining activities are performed in the light of evaluation goal. Figure 3 show the goal based viewpoint of the ALMA.

This method helps us to remove issues related to modifiability at SA level. The aim of modifiability can be:

- Maintenance cost prediction- approximating the effort required to fulfill software change scenarios

- Risk assessment- recognizing the types of changes for which a SA is inflexible.
- SA selection- matching two or more candidate SAs to choose the superior candidate.

Before the implementation of SAs, ALMA is mostly utilized but there is no any issue to say that it is not good for legacy system reengineering projects. The inputs comprise SA specification and quality requirements. Successful implementation of ALMA is in telecommunication, information system, and medical domain and embedded systems. The main advantages of ALMA are the identification of risks in SA, assessment of the efforts required to billet change, or selection of an optimal SA.

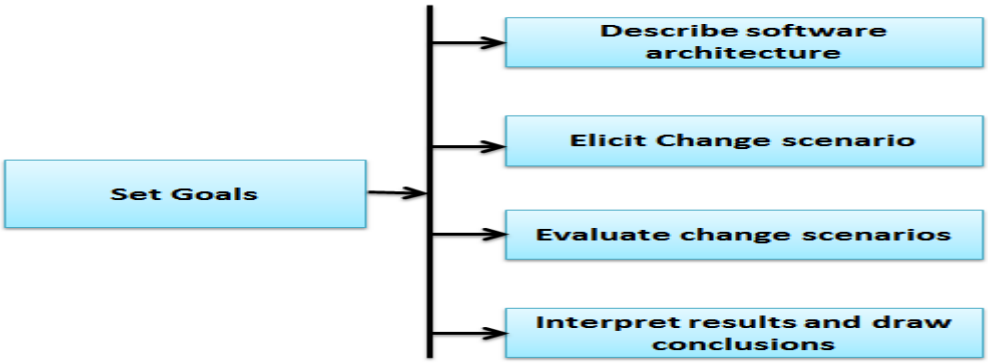

Fig. 2 . Process model of ALMA

ALMA frequently contains only a small set of stakeholders, which include software architect and development team. Both methods top-down and bottom up can be applied, in top-down approach starting from a predefined scenario classification while in bottom up approach starting from concrete scenario and building up categories of scenarios. ALMA helps to provide some techniques to select relevant scenarios and to decrease the number of scenarios. ALMA also guide that when to stop generating scenario. ALMA comprises of goal setting, defining the SA, stimulating scenarios, assessing scenarios, and understanding results and illustration conclusions. To evaluate the SA against change scenarios ALMA uses impact analysis. Impact analysis achieved by recognizing the parts affected by the scenarios, assuming the modification required, and finding the current effects. The results are construed depending on the goal of evaluation. A framework provided by the ALMA, describe the result quantity. ALMA is deliberate quite mature because it is validated through many applications. The process model of ALMA is given in fig 2.

4.2. Performance Assessment of Software Architecture

Williams and Smith offered a method to measure performance related problems at SA level in, called Assessment of Software Architecture (PASA). This method contain anti-patterns and performance sensitive SA styles as analysis tools and validate the SA analysis activity of the performance engineering process reported in.

The main objective of PASA is to measure the ability of the candidate SA with respect to performance objective. PASA helps the SA analysis activity using scenarios related to performance as a source of reasoning. Furthermore, this analysis also checks other quality attributes like maintainability. PASA also used to find the differences between different SAs.

PASA can be applied in early in the development process which is called post-deployment or during up-gradation of a legacy system. This method has been applied to embedded systems, real-time system, and Web-based systems and in the financial systems. By using various views PASA require SA descriptions document. If the documentation of SA is not well defined then the problem may be arise which is architectural information is take out from software code, developers and some other artefacts.

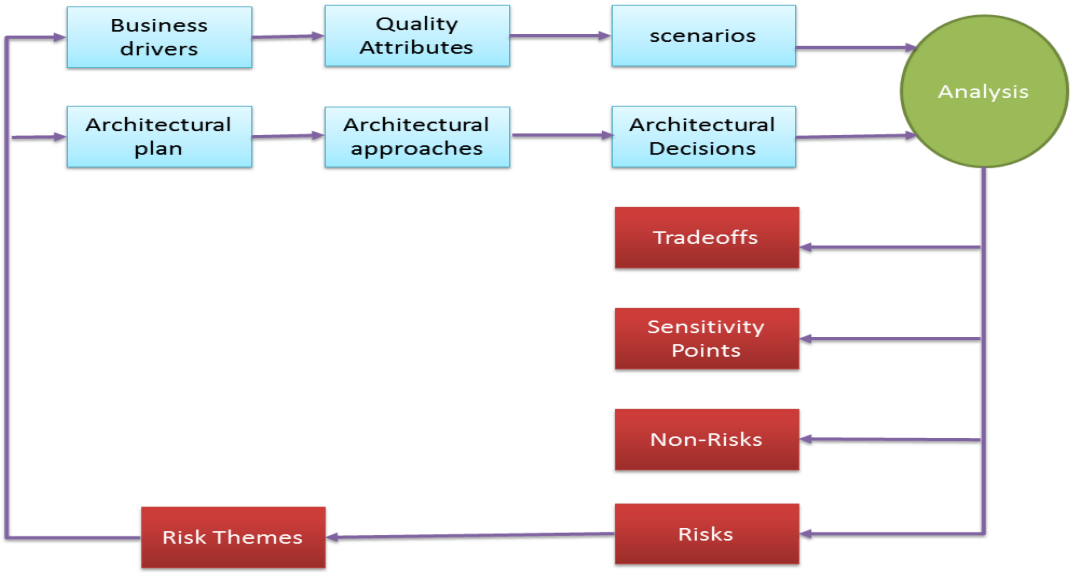

Fig. 3. Framework of PASA.

As shown in Fig. 4, PASA has ten steps. This evaluation method start with process presentation session aimed at setting the main objectives, finding the stakeholder's anticipation, recognizing the information required and relating the various aspects of the method. In next step of PASA, evaluators get a overview of the SA whiteout any extra descriptions.

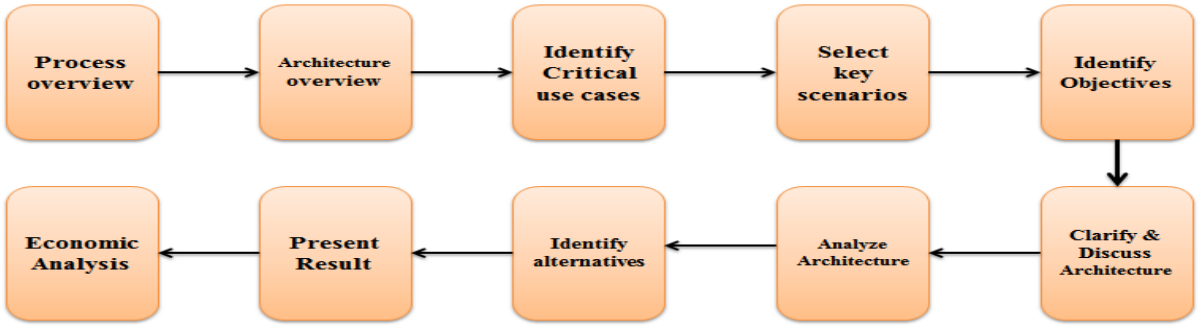

Fig. 4. Process model of PASA.

After this step the next one is to identify the critical use cases. The critical uses are those, for which there is major performance risk. The evaluation team and developer both work together to select key performance scenarios within each use case. Following the overall practices of SPE, PASA needs the selected scenarios to be documented using improved UML sequence diagrams. Every key scenario has one or more than one objective related to it. The goals of the performance can be defined in relations of throughput, response time or conditions on resource usage.

The next step of PASA is to identify architectural styles or patterns that is used in SA. If there is any deviation from the archetype of the style or pattern, the evaluators try to find whether there is any adverse effect due to that deviation. The evaluators do refactoring if there are any anti patterns found. PASA also uses multiple quantitative method and techniques for performing modelling counting software and system execution models. This process ends with a presentation of the result to clients and economic analysis of the assessment workout.

This method includes both qualitative and quantitative techniques to prove the potential risks that may be intrinsic in a SA. In addition this method also clarify that how different scenarios can be useful in

symbolizing run-time quality attributes like performance. By using different case studies, PASA and its various techniques have been validated.

4.3. Architecture Trade off Analysis Method

Architecture trade off Analysis Method (ATAM) was initially presented as a software design method. It provides the reasoning about software architecture and quality attribute. Trade-off Analysis Method can be implemented at any stage of software development life cycle. Business goals and software description can be provided as an input in ATAM, it also provides the many social and technical environments. Architecture trade-off Analysis Method consist of four phases, these phases are further divided into nine activities. There are many activities which are repeated in phase one and two. Stakeholders and technical staff are considered in first and second phase respectively. ATAM doesn't provide any particular technique. Anyhow ATAM is considered an evaluating technique in many software architecture designs. Process model of ATAM is given in Fig 5.

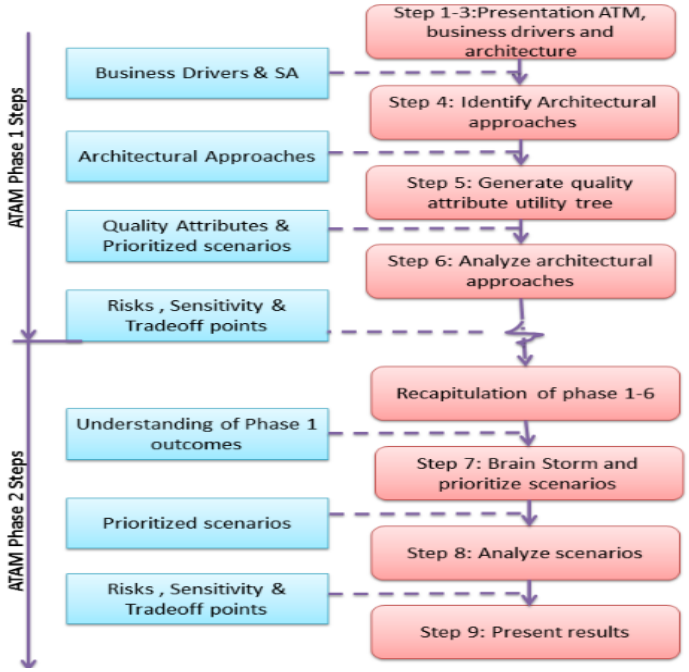

Fig. 5. ATAM process model.

5. Method Comparison

It is necessarily to define the unique definition for software architecture evaluation (SA), in some cases it isn't possible to define it with quality attribute. There are many methods which defined the SA architecture in respective perspective. Every SA method has a unique goal and specific view to achieve this goal. Some common methods with respective to specific goal are represented by following table 2.

Table 2. Comparison of SA Methods

SAAM	Identify the SA risk
ALMA	Identify the quality attribute
PASA	Identify the performance issue related risk
ATAM	Identify the trade-off points
SAAMTOOL	Identify the evaluation process

Proposed Activities flow graph for Generic SA Evolution is show in Fig. 6.

Fig. 2. Flow graph of proposed activities.

6. Conclusion

This paper is based on scenario based study of SA and there systematically comparison. Evaluation of SA is carried out with the help of proposed comparison framework. This framework is equally supported with little bit modification for other domains. SA evaluation comparison helps us to understand SA views and supportive features. We can quantify and qualify the SA attributes and scenario based approach.

Comparison help us to highlight the no of issues with the existing method expect ATAM which provide the comprehensive support for SA. Comparison also reveals no exact definition for SA not supported tool is available. That's mean a lot of work is required in this field. With the help of this comparison we have proposed five activities for generic process of SA evaluation which are given below

1) Planning and preparation for SA evaluation.
2) Approaches in SA evaluation
3) Implicit and explicit quality scenario in SA evaluation
4) Analysis of SA Approaches
5) Final result comparison and analysis.

References

[1] Bahsoon, R., & Emmerich. W. (2003). Evaluating software architectures: Development, stability, and evolution. *Proceedings of the ACS/IEEE Int. Conf. on Computer Systems and Applications*. July, 2003. Tunis, Tunisia.

[2] Clements, P., *et al*. (2002). *Evaluating Software Architectures: Methods and Case Studies*. Addison-Wesley.

[3] Bass, L., *et al*. *Software Architecture in Practice*. Addison-Wesley.

[4] Ali-Babar, M., *et al*. (2004). A framework for classifying and comparing software architecture evaluation methods. *Proceedings of the Australian Software Engineering Conference.*

[5] Dobrica, L., & Niemela. F. (2002). A survey on software architecture analysis methods. *IEEE Transactions on Software Engineering*, 28(7).

Ali Athar is a research scholar at College of Electrical and Mechanical Engineering, NUST, Rawalpindi. He is with the Department of Computer Engineering, College of EME, National University of Sciences and Technology (NUST), H-12, Islamabad, Pakistan.

Rao Muzamal Liaqat is a research scholar at College of Electrical and Mechanical Engineering, NUST, Rawalpindi. He is with the Department of Computer Engineering, College of EME, National University of Sciences and Technology (NUST), H-12, Islamabad, Pakistan.

Farooque Azam is a research supervisor at College of Electrical and Mechanical Engineering, NUST, Rawalpindi, He is with the Department of Computer Engineering, College of EME, National University of Sciences and Technology (NUST), H-12, Islamabad, Pakistan.