NUML International Journal of Engineering and Computing
https://numl.edu.pk/journals/nijec
Print ISSN: 2788-9629
E-ISSN: 2791-3465
Volume: 2 Issue: 2
DOI:https://doi.org/10.52015/nijec.v2i2.56

A Comprehensive Survey of Cutting-Edge Methods for Software Architecture Evaluation

Amna Sajid[a,*], Muhammad Waqas Arshad[b]

[a,*] *School of Computing, National University of Computer and Emerging Sciences, Islamabad*
[b] *Department of Computer Science & Engineering, University of Bologna, Bologna-Italy (muhammadwaqas.arsha2@unibo.it)*
**Corresponding author: First Author (amnasajid1996@gmail.com)*

Submitted	Revised	Published
01-Dec-2023	13-Dec-2023	21-Feb-2024

Abstract

The crucial responsibility of assessing software architecture is of utmost importance in ensuring that a software system conforms to superior qualities. It is a crucial tool for cutting expenses and labor during the course of the software development lifecycle. The main goal of software architecture evaluation is to provide reliable methods for determining and improving the quality characteristics that are innate in software. This study is deeply devoted to investigating the wide range of techniques used in software architecture assessment. One of the main focuses of these evaluation techniques is scenario-based assessments, which offer a comprehensive picture of the software's behavior under different circumstances. Within the scope of this study, a thorough analysis is carried out on eighteen different methods. Thirteen of these fall into the category of early evaluation techniques, which are positioned to detect and address architectural problems at the outset of development. The other five strategies are classified as late assessment techniques and focus on validating and optimizing the software architecture in the latter stages of the development process.

Keywords: Software Architecture, Evaluation methods, Quality attributes

1. Introduction

In order to make sure that a software architecture is efficient, high-quality, and in line with the intended goals, software architecture evaluation techniques are essential. Scenario-based evaluations are a popular technique that analyzes realistic usage scenarios to evaluate the architecture's performance in a range of situations and spot bottlenecks and dangers. Workshops on quality attributes let stakeholders work together to identify and rank the qualities that are most important to the success of a project. While SAAM concentrates on comprehending the

connections between architectural aspects, techniques such as ATAM systematically assess trade-offs in architectural decisions [1].

To promote traceability and comprehension, DARWIN places a strong emphasis on recording design rationales. Modularity, coupling, and cohesiveness can be quantitatively evaluated using architectural metrics, while cost-benefit analysis brings economic factors into decision-making. System performance is predicted by performance modeling and simulation, which makes optimization easier. Using these techniques, separately or in combination, provides a comprehensive approach to evaluating software architectures, supporting informed decision-making and ensuring the architecture's resilience and alignment with project goals [2].

This research endeavors to cast a spotlight on the multifaceted landscape of software architecture evaluation by delving into the diverse array of methods that populate the literature. By categorizing these methods, the study sheds light on two distinctive yet interrelated classifications: early evaluation methods and late evaluation methods. Early evaluation methods, positioned at the inception of software development, facilitate the identification and resolution of architectural issues, offering a preemptive strike against potential challenges. On the other hand, late evaluation methods come into play during subsequent phases, focusing on the validation and refinement of the architecture in response to evolving project requirements and insights gained during development.

The survey conducted as part of this research illuminates the prevailing trends and nuances within the realm of software architecture evaluation. It underscores the symbiotic relationship between early and late evaluation methods, each playing a crucial role in fortifying the robustness and resilience of software architectures. Through a comprehensive exploration of these methodologies, this research aims to provide a valuable resource for practitioners and researchers alike, offering insights into best practices and emerging trends in the dynamic field of software engineering [3].

2. Literature Review

The evaluation of software architecture encompasses a diverse array of methodologies tailored to comprehensively assess its quality, effectiveness, and alignment with project goals. Among these methodologies, several distinct types of evaluation methods have emerged as critical tools in the software engineering landscape. These include scenario-based evaluations, where realistic usage scenarios are scrutinized to gauge the architecture's performance under various conditions. Mathematical modeling techniques are employed to provide a quantitative understanding of the architecture's behavior, offering a formalized approach to evaluation. Usage-based assessments focus on real-world usage patterns and performance metrics derived from actual usage scenarios.

Experience modeling is another avenue, leveraging insights from previous projects or domains to inform architectural evaluations. Prototyping involves the creation of prototype architectures to validate design decisions and assess their feasibility. Information modeling, on the other hand, utilizes models to represent and analyze the information flows within the architecture [3].

In the scope of this research, a meticulous examination of literature has been undertaken to delve into prominent evaluation methods. Notable approaches covered include SAAM (Software Architecture Analysis Method), ATAM (Architecture Tradeoff Analysis Method), ALMA (Architecture-Level Modifiability Analysis),

SABR (Scenario-Based Reliability Analysis), ISAAMCS (Improving Software Architecture Assessment Method by Considering Social Factors), SALUTA (Scenario-Based Architectural Liveness Evaluation), ESAAMI (Early Software Architecture In-Progress Monitoring and Improvement), and ALPSM (Architecture-Level Performance Simulation Model). These methods contribute to a comprehensive understanding of the intricacies involved in evaluating software architecture.

Late evaluation approaches, such as those proposed by Tvedt et al., Lindvall et al., Fiutem and Antonio, Murphy et al., and Sefika et al., play a crucial role in validating and refining architectural decisions in later stages of the development process. Some of these approaches are tool-based, further emphasizing the integration of technology in the evaluation process [4].

By synthesizing insights from these diverse evaluation methods, this research aims to contribute to the body of knowledge in software architecture evaluation. It provides a nuanced understanding of how these approaches, ranging from scenario-based analyses to tool-supported late evaluations, can be strategically employed to enhance the robustness and effectiveness of software architectures.

3. Evaluation Methods

The realm of software architecture evaluation is characterized by a rich tapestry of methodologies, each designed to comprehensively assess and enhance the quality and effectiveness of software structures. While numerous evaluation methods have been documented in the literature, this research takes a focused and comprehensive approach by delving into 18 distinct methods. These methods collectively represent a diverse set of tools and techniques, each contributing to the multifaceted landscape of software architecture evaluation.

To systematically categorize and understand these methods, six overarching types of evaluation approaches have been identified. The first among these is scenario-based evaluation, a method that involves scrutinizing the architecture's performance under various realistic usage scenarios. Mathematical modeling comes into play as a formalized and quantitative approach, offering a structured framework to analyze the intricate behavior of the software architecture. Usage-based evaluations draw insights from real-world usage patterns, leveraging actual usage scenarios to assess performance and usability [5].

Experience modeling, another category, taps into the wealth of insights garnered from previous projects or domains to inform architectural evaluations, offering a knowledge-driven perspective. Prototyping, on the other hand, involves the creation of prototype architectures to validate design decisions and assess their feasibility, providing a tangible and practical dimension to the evaluation process. Information modeling, the sixth type, deploys models to represent and analyze the information flows within the architecture, facilitating a comprehensive understanding of how data moves within the system.

By explicitly outlining these six types and delving into 18 specific methods within this framework, this research contributes to a nuanced understanding of the software architecture evaluation landscape. It provides a structured and organized approach to navigating the multitude of evaluation methods, offering insights into their unique strengths, applications, and contributions to the overarching goal of enhancing software architecture quality [6].

3.1 Early Evaluation Methods

In the early stages of software development, when key architectural decisions are being formulated, scenario-based evaluation provides a tangible and practical framework for validating design choices. It allows stakeholders to visualize the potential behavior of the system and gain insights into its strengths and weaknesses before substantial resources are invested in implementation. This proactive assessment aligns with the principle of risk mitigation, as issues identified early in the development lifecycle are generally less costly to rectify than those discovered in later stages [7].

Table 1: Software Architecture Early Evaluation Methods Comparison

Methods	Quality Attribute	Strengths	Weakness
SAAM	Modifiability	Identify areas of high complexity	not execute trade-off inquiry
ATAM	Modifiability	Applicable for static & dynamic properties.	No identification of architecture features
ALMA	Modifiability	Scenario generation stopping criterion	Not use quantitative study
CBAM	Modifiability	Business measures for particular system changes	Not perform trade-off analysis
FAAM	Interoperability, extensibility	Empowering the teams in applying FAAM session	No tool support
SAAMCS	Modifiability, flexibility	Measurement instruments to identify complex scenario.	No validation is done
SBAR	Performance, reliability	An iterative process for architecture evaluation.	Doesn't involves goal selection
ALPSM	Maintainability	Prediction with an understanding of requirements	No tool support
ESAAMI	Reusability	Introduces "protoscenario" that deployed during method's steps	Doesn't perform trade-off analysis

SACAM	Maintainability, interoperability	Compare several architecture from different domains.	Doesn't predict maintenance effort with size changing of components
SALUTA	Usability	Analyze the extracted usability patterns & properties.	Doesn't predict maintenance effort with size changing of components
SAAMER	Reusability	Designers, end-users are involved	No tool support
ISAAMCR	Flexibility. reusability	Provide architectural views analysis template.	No tool support

3.2 Late Evaluation Methods

Late evaluation methods represent a distinct phase in the software architecture evaluation process, focusing on the validation and refinement of architectural decisions in later stages of development. This phase often occurs after the initial design has been implemented or as the project progresses towards completion. In this context, several notable late evaluation methods have been introduced, each offering unique perspectives and methodologies. The late evaluation approaches covered in this research include Tvedt et al.'s Approach, Lindvall et al.'s Approach, Fiutem and Antonio's Approach, Murphy et al.'s Approach, and Sefika et al.'s Approach [8].

Table 2: Late Evaluation Methods Comparison

Methods	**Quality Attribute**	**Strengths**	**Weakness**
Tvedt et al.'s Approach	Accuracy	Identify actual and planned architecture, Changes are placed	Classes and design patterns are violated.
Lindvall et al.'s Approach	Maintainability	Compare the new, previous and planned architecture.	Inter-module coupling violated
Fiutem and Antonio's	Consistency	Compare and determine the inconsistency	Code is violated

Approach (Tool based)			
Murphy et al.'s Approach (Tool based)	Compliances	Check the declarative mapping between the two models	Calls between modules are violated
Sefika et al.'s Approach (Tool based	Integration	Integrates logic, static and dynamic visualization	Design patterns are violated

4. Conclusion

In conclusion, the survey of software architecture evaluation methods highlights a predominant reliance on scenario-based approaches in the current landscape. These methods, rooted in envisioning and analyzing realistic usage scenarios, prove to be foundational in understanding and enhancing the quality attributes of software systems. The emphasis on scenarios allows for a proactive assessment, enabling early identification and mitigation of potential architectural issues. The implementation of scenario-based methods has demonstrated efficacy in delivering singular or multiple quality attributes within software systems, contributing to the robustness and adaptability of architectures.

However, a comprehensive comparative analysis of these methods conducted during the survey has brought to light certain challenges and areas for improvement. Identifying these challenges is crucial for advancing the field and refining existing evaluation methodologies. Whether it be in the realm of scenario-based methods or other types of evaluation approaches, recognizing the limitations and addressing the identified issues is integral to the continual evolution of software architecture evaluation practices.

Moving forward, research and development efforts should focus on innovating and diversifying evaluation methods to address the identified challenges. This includes exploring novel approaches that can complement or enhance the effectiveness of scenario-based methods. Additionally, attention should be given to tool-based evaluations, incorporating technological advancements to streamline and automate the evaluation process.

In essence, while scenario-based methods have played a central role in software architecture evaluation, the field is dynamic, and there is an ongoing need for refinement and innovation. The survey sets the stage for further exploration and improvement, fostering a continuous cycle of research and enhancement to meet the evolving demands and complexities of contemporary software systems.

References

[1] M. A. Babar, B. Kitchenham, and R. Jeffery, “Comparing distributed and face-to-face meetings for software architecture evaluation: A controlled experiment,” Empirical Software Engineering, vol. 13, no. 1. pp. 39–62, 2008.

[2] P. Shanmugapriya and R. Suresh, “Software Architecture Evaluation Methods - A survey,” International Journal of Computer Applications, vol. 49, no. 16. pp. 19–26, 2012.

[3] M. A. Babar and I. Gorton, “Comparison of scenario-based software architecture evaluation methods,” Proceedings - Asia-Pacific Software Engineering Conference, APSEC. pp. 600–607, 2004.

[4] B. Roy and T. C. N. Graham, “Methods for Evaluating Software Architecture : A Survey,” Computing, vol. 545, no. 2008–545. p. 82, 2008.

[5] R. Kazman, L. Bass, G. Abowd, and M. Webb, “SAAM: a method for analyzing the properties of software architectures,” Proceedings of 16th International Conference on Software Engineering. pp. 81–90.

[6] S. Abrahão and E. Insfran, “Evaluating Software Architecture Evaluation Methods: An Internal Replication,” Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering. pp. 144–153, 2017.

[7] A. Patidar and U. Suman, “A Survey on Software Architecture Evaluation.” pp. 967–972, 2015.

[8] M. Konersmann, A. Kaplan, T. Kühn, R. Heinrich, A. Koziolek, R. Reussner, ..., J. P. Töberg, "Evaluation methods and replicability of software architecture research objects," in *2022 IEEE 19th International Conference on Software Architecture (ICSA)*, March 2022, pp. 157-168.