International Journal of Scientific Research and Engineering Development-– Volume 5 Issue 4, July- Aug 2022

Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

SOFTWARE ARCHITECTURE: A REVIEW

Mr. Rahul Joshi*, Ms. ArjinderKaur**,Ms. AanchalMadaan***, Er. RamandeepKaur****, Ms.Pooja*****

*(Computer Applications, GGI, Amritsar
Email: rahuljoshi11198@gmail.com)
** (Computer Applications, GGI, Amritsar
Email: karjinder6262@gmail.com)
*** (Computer Applications, GGI, Amritsar
Email: aanchalmadaan20@yahoo.com)
**** (Computer Applications, GGI, Amritsar
Email: rdhillon1223@gmail.com)
***** (Computer Applications, GGI, Amritsar
Email: ramansavlia@gmail.com)

----------**********----------

Abstract:

Over thepast 10 years, Software Architecture has emerged as the prominent paradigm in large-system development. The software architecture is the blue print of whole system. Software architecture is a concept that is easy to understand, and that most engineers intuitively feel, especially with a little experience, but it is hard to define precisely. In particular, it is difficult to draw a sharp line between design and architecture. Architecture is one aspect of the design that concentrates on some specific features. In this paper I am representing some concept related to the software architecture.

Keywords **—ASR, Quality attributes, Stakeholder**

----------**********----------

INTRODUCTION

The software architecture is the blue print of whole system. Architecture is one aspect of the design that concentrates on some specific features.*The software architecture of a system is a set of structures needed to reason about the system, which comprise software element, relations among them and properties of both*. [1]

Implication of this definition

(1)Structure means set of element held together.

(2)Software Architecture hide the most concrete detail and only show details related to the reason about the system.

(3)The behavior of the system can be the part of architecture if behavior used to reason about the system.

(4) Only externally-visible properties of elements are assumed that one element can make about another: provided services, required services, performance characteristics, fault handling, resource usage.

Basically software architecture is a bridge between *Requirements Specification* and the final resulting system. The requirement can be of three types

- System functionality requirement,
- Quality attribute requirement and
- Constraints on the system.

System is built for some gain in terms of functions (working) these gains are called Functional requirements. Quality attributes are hidden types of policies which indicate how well the system satisfies the needs of its stakeholders. Examples of quality attributes areportability ,usability ,reusability etc.

Constraints are the rulesthat must be followed by the system to be developed.
Constraints are of two types

- Technical
- Nontechnical

Technical constraints:
Technical constraints related with the questions like particular programming language must be used; particular model must be reuse etc.
Nontechnical constraints**:**
Nontechnical constraints related with government policies, business policies.

The software architecture of a program or computing system is a depiction of the system that aids in the understanding of how the system will behave. Software Architecture is method of visualizes the system before implementation. Visualization means depiction of the system that aids in the understanding of how the system will behave after implementation. These are early design decision which taken by an architect to show not only the functional but other quality attribute to reason about the system.

II. FORMAL DEFINITION

According IEEE 1471-2000, Software architecture is the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution. According Perry and Wolf, A set of architectural (or design) elements that have a particular form. According Boehm et al., Software system architecture comprises collection of software and system components, connections and constraints. A collection of system stakeholders' need statements. A rationale which demonstrates that the components, connections, and constraints define a system that, if implemented, would satisfy the collection of system stakeholders' needs statements.
Clements et al., 1997The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them.

III. Why is software Architecture important

Software Architecture is assuring that stakeholders' requirements have been met and aids in discovering flaws while modification is still a fraction of time and cost compared with later updates [2] Software Architecture is the set of design decisions which, is made incorrectly, may cause your project to be cancelled [3] Software architecture is very important to understand the system, reduce risk, to quality assurance, to meet user requirement. These are very critical decision about the system those are responsible for success and failure of the software. Software architecture not only provides guidelines for design decision but also behaves as common language between all stakeholders (user, manager, architect, developer, tester etc.).Using that software architectureall stakeholders' can understand the system before implementation on their own perspective. "On their own perspective" means that every stakeholders' have their own reason about the software as user want some functionality. There are some reason showings below why use architecture
(a) The required quality attributes such as high performance, modifiability, high secure, scalability, reusability is Permits/Precludes by architecture.(b)Depend on the business goal the architect omen the system qualities.(c)Software Architecture is the mechanism to communicate between stakeholders. (d)The early design decision taken by architect defines constraints on to the implementer. (e) Architecture not only influences the system being developed but it also affects the organizational structure.(f) Software Architecture also predicts cost and schedule estimate with the help of project manager and developer.(g) Software Architecture emphasis on development of independent element.(h)Architectural pattern restrict lexicon of design alternatives. This minimizes the complexity of the system (i) Architecture is used to provide training to new

member of team.(j)The architecture not only support to achieve the goal of the system but also influence basis of test plans ,testing, division of team, Incremental deployment, basis of maintenance etc.

IV. DOCUMENTING SOFTWARE ARCHITECTURE

Documentation is very important not because it is required but because documentation speaks for architect. Documentation of architecture is required when new member join the system, when we want to enhanced the functionality of the system, when this decision is reused in some other system etc.

Architecture documentation is used to

(i) Understand system
(ii) Implement the system
(iii) Basis for evaluation
(iv) Generating automatic code
(v) Train a new member of the project
(vi) Enhanced communication between architect and stakeholder

Documentation of architecture is called view. View is the language of software architecture. Visualizing, documenting and specifying architecture is exactly the purpose of view. The views are used to describe the system from the viewpoint of different stakeholders, such as end-users, developers and project managers. Several architectural views share the fact that they address a static structure, a dynamic aspect, a physical layout, and the development of the system [4].System has many structures (view) no single structure can be the architecture. What view or structure will architect select depends on the goal of the software system. Different view reveals different quality goal. Therefore that quality goal which is most important in the system will affect the way to choose the view. Generally there are number of quality goal related with the system so the documentation contains collection of views.

There are number of views related with the system. Architect chooses the view related to its goal and quality attribute specification.

In 1974 Parnas observed that software is composed of many structures. After 90's it has been realize that, similar to buildings (with plumbing and electrical and wall diagrams), different views of a system are required (Perry and Wolf) then in 1995: Kruchten defined the "4+1 views" approach to software architecture.

The 4+1 view model is widely used and accepted by Software intensive industry to describe architectural blueprint. This view describes the system from the viewpoint of different stakeholders.

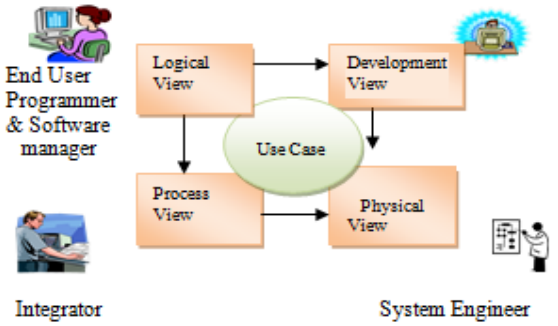

Fig.1 4+1 view model

Logical View: This view is concern with the functionality of system that provides to the end user of the system.Development View: This view illustrates a system from a programmer's perspective and this view also describes how the software is organized into modules during development process.

Physical View: This view depicts the system from a system engineer's point of view. This view describes how the software components are mapped onto the environment.

Process View: This view is related with run time behavior of the system. This view depicts what process or threads will be created and how will they communicate and share resource.

These views combines using another view called *use case or scenario view* which is used to help

developer to understand and reasoning about architectural design. According to Bass[1] there are three way to express (write) view. These are Informal notation, Semiformal notation, Formal notation.

Representation of software architecture through general purpose programming tool called informal notation. Informal notation also known as "box-and-line" diagram. In "box-and-line" design the box is consider as component and lines usually represent connection between these components. Simple software architecture can be depicted but more complex design can't show using this approach. Generally power point is used as informal notation. Semiformal and formal notations are the standardized notation to depict the software architecture. UML (Unified Modeling Language) is less informal then box–and-line notation so it is called semi-formal notation. UML is graphical modeling language used to describe the system through different types of diagrams [6]. UML can represent important structural aspect of a system. UML has still no structure and specification for modelling user interfaces. There are number of ADLs (Architecture Description Language) proposed which are consider as formal notation. Formal notation not only provides graphical vocabulary but also an underlying semantics for architecture representation. ADLs are based on mathematical formality, including precisely defined syntax and semantics. ADLs are "… aimed at giving practitioners better ways of writing down architectures so that they can be communicated to others and in many cases analyzed with tools" [9]. ADL supports in automatic code generation and automation of analysis through associated tools. ADLs example includes ACME [7][8] , Unicon [9], Aesop[9], Darwin[10].

V. SOFTWARE ARCHITECTURE ANALYSIS

Architecture is the product of early design phase and it has a profound effect on our project.Architecture Analysis has different goal. This includes assessing system maintainability, usability, sustainability, and security and resilience against attack [5].Basically Software architecture is made up of number of design decision which is taken by an architect on response to different requirement of different stakeholders. As end user concern with functionality, project manager related with schedules (cost, money), and tester concern with testability etc. How the architecture follow these concerns. Is the architecture satisfy all stackholders' concern, will resulting software satisfy all business concerns or business goals, are the architecture perform according to quality goals. These questions are answered by analysis of software architecture. The analysis of architecture enables early prediction of a system's quality [5]. There are two types of method to evaluate the architecture. One is Early Software Architecture analysis method and second Late Software Architecture analysis method. Early analysis methods are applied to any stage in the architecture creation process to examine those architectural decisions already made and choose among architectural options that are pending. Different requirement are checked against architecture to justify that requirement is fulfilled or not. Early analysis methods are very useful in cost and schedule prediction.[11] There are four types of approaches for early software architecture evaluation scenario-based, mathematical model based, simulation based, experience based. Among them scenario based method is simple and flexible[12]. SAAM (Scenario based Architecture Analysis method) and ATAM (Architecture Tradeoff Analysis Method) are the example of scenario based method.While Late software architecture evaluation methods are applied when not only the architecture is completed but implementation is complete as well. Sometime it is needed to change in the requirement is desirable [13] .The developer who work under intense time pressure and heavy work load cannot always follow the best way to implement change ,so the actual architecture deviates from planned one. These methods identify the

difference between actual and planned architecture. These methods used to guide how to *reconstruct* actual architecture so that it conforms to plan one. There are two types of approaches for late software architecture evaluation metric based, and tool based. After completion of evaluation two questions can be answered.

1. Is this architecture suitable for the system for which it was designed?
2. Among the number of architectural solution which one is most suitable for the system?

Here suitability means the system that follows that architecture will fulfill its entire Quality goal. That is specified in ASR (Architecturally significant requirements). Somewhat architecture shows the tradeoff between two quality goals. For ex performance and availability has negative effect on security. As numbers of resources are increase to increase performance and availability but it will negative impact on security. It is the work of analysis to find this type of tradeoff between goals and choose that architecture which compromise between the goals depends on requirement of major stakeholder. Architectural evaluation doesn't provide result like this software is good or bad but it only tells the analyst about risk at the system.

VI. CONCLUSION

In this paper I am representing topics related with the software architecture which is the road map for implementer. Software Architecture Analysis is the latest topic on which numbers of researches are going on. Automation of the tool to analysis of software architecture is current topic for researcher.

REFERENCES

[1] L Bass , P Clements, R Kazman , *Software Architecture in Practice*, SEI Series in Software Engineering, 3rd edition, pp 4
[2] A Alkussayer and W H Allen ,"security risk analysis of software architecture based on AHP" In proc. 7th Int conf. on Network computing,2011,pp 60-67
[3] Eoin Woods
[4] BananiRoy , T.C. Nicholas Graham , *"Method of evaluating software architecture : A Survey"*, Queen's University at Kingston Ontario Canada , April 2008
[5] M Almorsy, J Grundy and A s Ibrahim,*" Automated Software Architecture Security Risk Analysis using Formalized Signatures",*Center for computing and engineering Software System, Melbourne ,Australia
[6] J. Rumbaugh, I. Jacobson and G. Booch, *The Unified Modeling Language Reference Manual,* Addition-Wesley,1999
[7] Carnegie Mellon University, *The Acme Architectural Description Language, On-line at http://www-2.cs.cmu.edu/~acme/, 1998.*
[8] D. Garlan, R. T. Monroe, and D. Wile, "*Acme: An Architecture Description Interchange Language", Proceedings of CASCON* '97, Toronto Canada, November 11 1997, pp. 169-183.
[9] M. Shaw and D. Garlan, Software Architecture, Perspectives on an Emerging Discipline, Prentice Hall, Upper Saddle River NJ, 1996.
[10] M. Jazayeri, A. Ran, and F. van der Linden, *Software Architecture for Product Families, Principles and Practice,* Addison-Wesley, Boston MA, 2000.
[11] M. A. Babar and I. Gorton, "Software Architecture Review: The State of Practice," in Computer, vol. 42, no. 7, pp. 26-32, July 2009, doi: 10.1109/MC.2009.233.
[12] D. Batool, Y. H. Molta, A. Sarwar, M. A. Abbasi and J. Jabeen, "Software architecture and requirements: A systematic literature review," 2015 International Conference on Information and Communication Technologies (ICICT), 2015, pp. 1-5, doi: 10.1109/ICICT.2015.7469582.
[13] A. Baabad, H. B. Zulzalil, S. Hassan and S. B. Baharom, "Software Architecture Degradation in Open Source Software: A Systematic Literature Review," in IEEE Access, vol. 8, pp. 173681-173709, 2020, doi: 10.1109/ACCESS.2020.3024671.

