Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 5, Issue. 1, January 2016, pg.31 – 38

Prediction of Unacceptable Software Performance in Sequential Development Methodology

Mohammed Faisal
Lecturer-IT, Nizwa College of Technology, OMAN
faisal31621@gmail.com

Abstract: ***In the sequential approach for the software development, if found any problems in the previous any phase we have to wait till completion of the complete waterfall phases because it won't allow us to go back to the previous phases until the completion of the last phase. This is the major drawback of the waterfall model. If we emphasize where often we required to do the changes. Let's assume from the customer side if all requirements are well known in advance, if we have nothing to do with the requirement changes, mostly wrong design will be the cause of dissatisfaction of the customer by seeing the bad performance of the software system even though all requirements are met but still they have to compromise in the quality of the software. In this paper Architecture Tradeoff Analysis Method has been used in the second phases of the waterfall model for evaluating software architectures relative to quality attribute goals such as modifiability, security, performance, and reliability, view point oriented requirement methodology also has been used to identify the quality attributes from different viewpoints before generating the utility tree. For the satisfactory performance of the software system by choosing the best alternate design. The intention of this research work is to engineer the product by evaluating the software architecture instead of try with one shot prepared software design and get stuck with low performance software product.***

Keywords: ***ATAM - Architecture Tradeoff Analysis Method, utility tree, VORD- View point Oriented Requirement Definition, Waterfall Model, software Performance.***

I. Introduction

Changes in the software after development is always a challenge for the software engineers, in this paper, fusion has been done between sequential development (waterfall model), Architecture Tradeoff analysis Method, in the second phase of waterfall model will emphasize on architecture of the system along with the design, after analyzing it by applying ATAM its easy for the developers to choose best performance alternate design. During the process of Architecture Tradeoff Analysis Method, generation of utility tree plays the very important role where we need to focus on the quality attributes from different view point, in this paper it's also proposed to use the View point Oriented Requirement Definition to generate the utility tree.

II. Problems in Waterfall development

a. In Waterfall Quality attributes evolution and risk mitigation not covered.

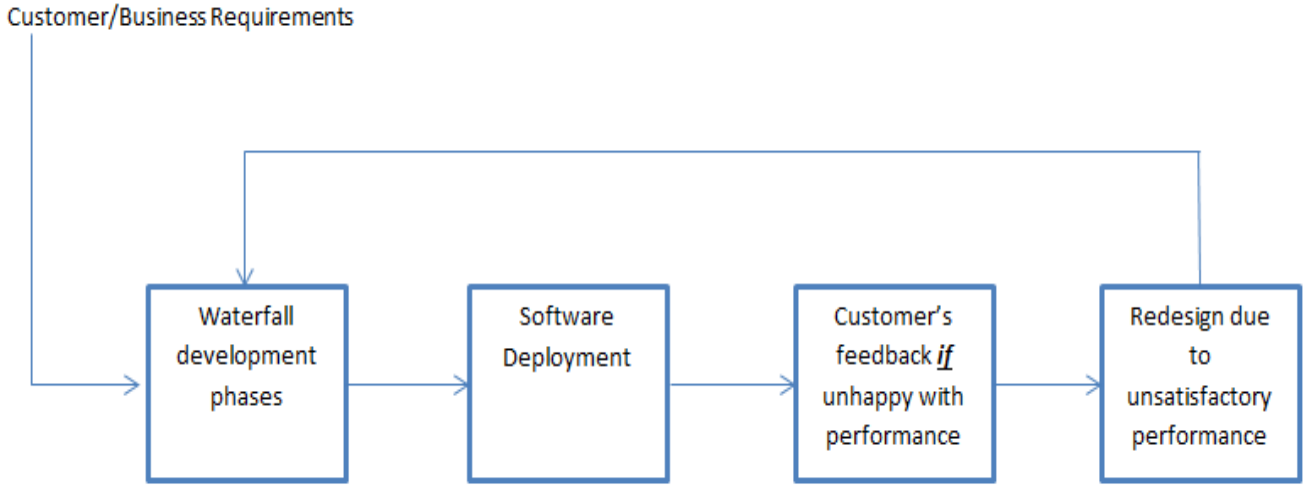
b. Customers may unhappy with the software performance because performance not evaluated before coding starts.

c. Back tracking is not possible until last phase is not finished.

d. In Waterfall development, evolution of software architecture not considered.

e. Because software architecture not considered there is no point to evaluate the architecture to know the performance of the product.

III. Performance issues in Waterfall



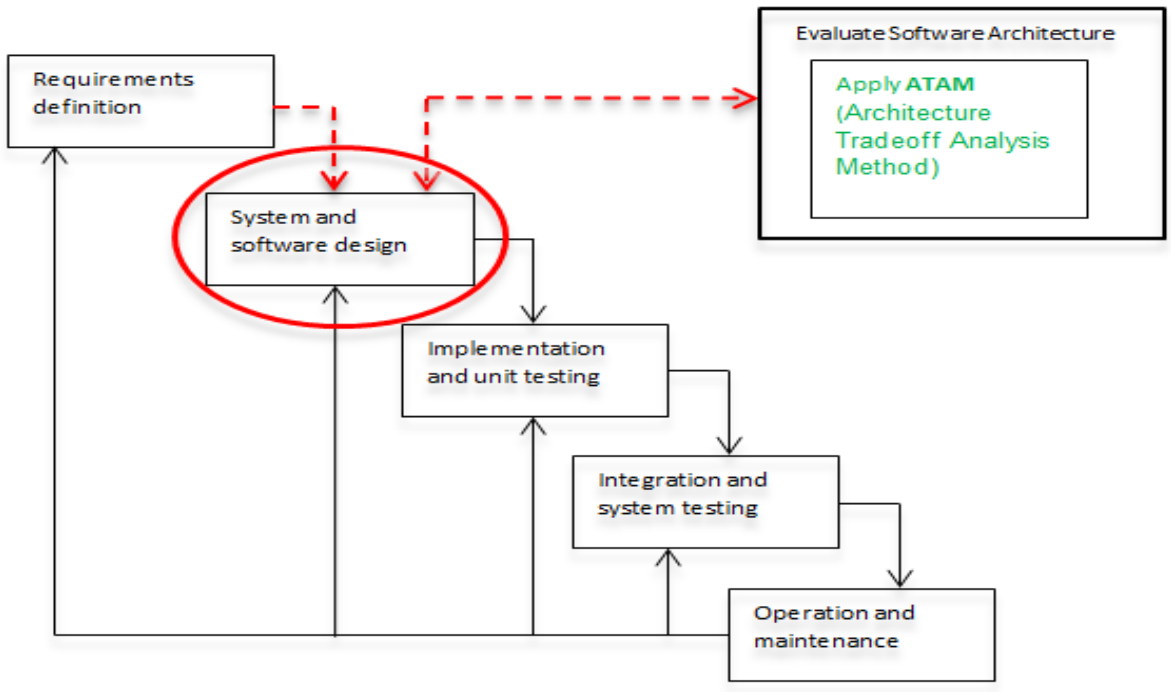
IV. Proposed Solution to early identify the unsatisfactory performance of the Software Architecture

a. Emphasize the System Quality attributes (Non Functional Requirements).
b. Adopt the Architecture Tradeoff Analysis Methodology (ATAM) in Waterfall Model

A. What's ATAM?

- To evaluate the architecture, SEI has developed the Architecture Tradeoff Analysis Method (ATAM).
- Purpose: To assess the consequences of architectural decisions in light of quality attribute requirements.
- Primarily a risk identification mechanism

V. Adaptation of ATAM in Waterfall Model



Architecture Tradeoff Analysis Method (ATAM)

ATAM Steps:

Evaluators and decision makers Present's:

a. ATAM
b. Business drivers
c. Architecture

- Identify architectural approaches
- Generate quality attribute utility tree
- Analyze architectural approaches

Evaluators, decision makers along with stakeholders

- Brainstorm and prioritize scenarios
- Present results

Present ATAM:

Evaluation Team presents an overview of the ATAM

- ATAM steps in brief
- Techniques
  - Utility tree generation
  - Scenario brainstorming/mapping
- Outputs
  - Utility tree
  - Scenarios
  - Risks and "non-risks"
  - Sensitivity points and tradeoffs

Present the Architecture:

Architect presents an overview of the architecture including (for example):

* Technical constraints such as an OS, hardware, or middle-ware prescribed for use
* Other systems with which the system must interact
* Architectural approaches/styles used to address quality attribute requirements

Present Business Drivers:

ATAM customer representative describes the system's business drivers including:

Business context for the system

High-level functional requirements

High-level quality attribute requirements like:

-High availability

-Time to Market

-High Security

Identify Architectural Approaches:

* **Start to identify places in the architecture that are key for realizing quality attribute goals.**

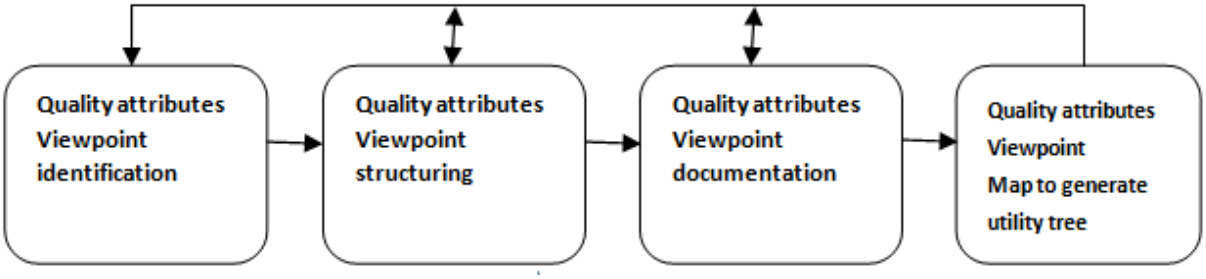
Identify any predominant architectural approaches – for example:

* client-server
* 3-tier
* proxy
* publish-subscribe
* redundant hardware

Generate Utility Tree:

Additional proposed steps to identify the quality attributes:

Generating the utility tree is most important steps in ATAM [2], we may use view point oriented Definition (VORD) [7] to identify most important quality attributes from different viewpoints such as end user view point, developer viewpoint and business view points.



Quality attributes Viewpoint identification:

- Discover viewpoints which receive system services and identify the services provided to each viewpoint

Quality attributes Viewpoint structuring

- Group related viewpoints into a hierarchy. Common services are provided at higher-levels in the hierarchy

Quality attributes Viewpoint documentation

- Refine the description of the identified viewpoints and services

Quality attributes Viewpoint-mapping with utility tree

- Transform the analysis to generate the utility tree

- Identify, prioritize, and refine the most important quality attribute goals by building a *utility tree*.
  - A utility tree is a top-down vehicle for characterizing the "driving" attribute-specific requirements
  - Select the most important quality goals to be the high-level nodes (typically performance, modifiability, security, and availability)
  - Scenarios are the leaves of the utility tree
    - Output: a characterization and a prioritization of specific quality attribute requirements.
  - Scenarios are used to
    - Represent stakeholders' interests
    - Understand quality attribute requirements
    - A good scenario makes clear what the stimulus is that causes it and what responses are of interest.

Sensitivity & Tradeoffs

Stakeholders will join form this step.

Sensitivity:

- A property of a component that is critical to success of system.
- The number of simultaneous database clients will affect the number of transaction a database can process per second. This assignment is a sensitivity point for the performance
- Keeping a backup database affects reliability.

Tradeoff point:

- A property that affects more than one attribute or sensitivity point.
- Keeping the backup database affects performance also so it's a trade-off between reliability and performance.

Risks & Non-Risks:

Risk: The decision to keep back up is a risk if the performance cost is excessive

Non Risk**:** The decision to keep back up is a non-risk if the performance cost is not excessive

Brainstorm & Prioritize Scenarios:

- Stakeholders generate scenarios using a facilitated brainstorming process.
  - Scenarios at the leaves of the utility tree serve as examples to facilitate the step.
  - The new scenarios are added to the utility tree

Present ATAM results:

- Utility tree.
- Set of Scenarios.
- Risks and "non-risks"
- Sensitivity points and tradeoffs.

VI. Conclusion

Risk and non-risk issues will be identified, Utility tree helps to identify and compare the expected quality attributes and present quality attributes. Chance to resolve the risk and change the architectural plan in order to achieve satisfactory performance. It's supporting Waterfall model as we are not moving to the next phase until unless we are not implementing ATAM at System and Software design phase. Gets the change to change software design to support and increase performance of software architecture.

References:

1. ©Ian Sommerville 2004 Software Engineering, 7th edition
2. http://www.sei.cmu.edu/reports/00tr004.pdf
3. Software Engineering Institute - http://www.sei.cmu.edu/reports/00tr004.pdf
4. Evaluating a Software Architecture by Barry Boehm - http://www.pearsonhighered.com/samplechapter/020170482X.pdf
5. Scenario-Based Software Architecture Evaluation Methods: An Overview by Mugurel T. Ionita , Dieter K. Hammer , HenkObbink

http://www.win.tue.nl/oas/architecting/aimes/papers/Scenario-Based%20SWA%20Evaluation%20Methods.pdf

6. Software Architecture Quality Evaluation - Approaches in an Industrial Context by FransMårtensson - ISSN 1650-2140 ISBN 91-7295-082-X - http://www.ide.bth.se/~hgr/Martensson_lic.pdf

7. Requirements engineering with viewpoints - by Gerald Kotonya and Ian Sommeiville - http://www.panda.sys.t.utokyo.ac.jp/kushiro/ReferencePaper/Requirements%20engineering/00487319.pdf

