PREDOMINANT QUALITY ATTRIBUTES IN EVALUATING SOFTWARE ARCHITECTURE AND ADDRESSING SCENARIO COVERAGE PROBLEM

Dr. P. Shanmugapriya, Department of IT, SCSVMV University, Enathur, Tamilnadu, India

N. Kumaran, Department of IT, SCSVMV University, Enathur, Tamilnadu, India

Abstract: *The quality of the software is measured primarily against the degree to which user requirements, such as correctness, reliability and usability are met. The use of a well defined model of the software development process and good analysis, design and implementation techniques are prerequisite to ensuring quality. Software Quality Attributes are the benchmarks that describe system's intended behavior within the environment for which it was built. The quality attributes provide the means for measuring the fitness and suitability of a product. Essential quality attributes need to be characterized in a system-specific way. Most of the Software architecture evaluation methods were not addressing the essential quality attributes. This paper present a report collected from large software firms which gives set quality attributes playing a vital role in getting quality software.*

Keywords: *Software Architecture, Quality Attribute*

1. INTRODUCTION

Developing high quality software is tough, especially when the interpretation of term "quality" is irregular based on the environment in which it is used [1]. In order to know if quality has been achieved, or degraded, it has to be measured, but determining what to measure and how is the difficult part. Measuring quality is challenging since measurement must consider the perspectives of different stakeholders Garvin [2]. The factors that affect quality are termed as quality attributes. There are different categorizations of quality attributes. Quality attributes can be categorized into two broad groups: attributes that can be directly measured (e.g. performance) and attributes that can be indirectly measured (e.g., usability or maintainability). Writing good quality attribute scenarios is hard, but it's an important skill. Quality attribute scenarios are where the desired vague software behaviors are turned into tangible and measurable goals that guide software development. Software architecture has a profound effect on most qualities in one way or another, and software quality attributes affect architecture. Identifying desired system qualities before a system is built allows system designer to mold a solution (starting with its architecture) to match the desired needs of the system within the context of constraints (available resources, interface with legacy systems, etc). When a designer understands the desired qualities before a system are built, and then the likelihood of selecting or creating the right architecture is improved.

2. RELATED WORK

Software architecture evaluation has emerged as an important software quality assurance technique. The principle objective of evaluating architecture is to assess the potential of the chosen architecture to deliver a system capable of fulfilling required quality requirements. Several quality models such as Bohem's Quality Model (1978) [3], McCall's Quality Model (1997) [4], and ISO 9126 Quality Model [5] have been proposed to evaluate the quality related issues at the architecture level. Most architecture evaluation methods conduct evaluation for individual quality attributes first and consolidate the results later. Attribute specific evaluation requires reasoning models and expertise for the quality attribute in focus.

3. ACHIEVING QUALITIES

Scenarios appear to be the most utilized form for acquiring data [6]. Scenarios are used to represent *stakeholders'* interests, understand quality attribute requirements. A good scenario makes clear what stimulus causes it and what responses are of interest. The quality attributes must be described in terms of scenarios, allows an architect to make quantifiable arguments about a system. A scenario defines the source of stimulus, the actual stimulus, the artifact affected, the environment in which it exists, the effect of the action, and the response measure. Writing scenario is only possible when relevant requirements have been identified and an idea of components has been proposed. Writing effective scenarios takes some time to learn. But it's an important skill, as it's in the scenarios where the desired vague software behaviors are turned into tangible and measurable goals. Measurable goals tell you what architectural approaches and tactics to apply as you design the system.

Scenarios help describe the qualities of a system, but they don't describe how they will be achieved. Architectural tactics describe how a given quality can be achieved. For each quality there may be a large set of tactics available to an architect. It is the architect's job to select the right tactic in light of the needs of the system and the environment. Whatever approach is chosen, it must be justified and documented. Quality attribute requirements drive the design of the software architecture. Quality attribute requirements stem from business and mission goals. Scenarios are a powerful way to characterize quality attributes and represent stakeholder views. Software architecture drives software development throughout the life cycle. Software architecture must be central to software development activities. These activities must have an explicit focus on quality attributes. These activities must directly involve stakeholders, not just the architecture team. System qualities can generally be categorized into four:

Runtime System Qualities - Runtime System Qualities can be measured as the system executes.

Non-Runtime System Qualities - Non-Runtime System Qualities cannot be measured as the system executes.

Business Qualities - Non-Software System Qualities that influence other quality attributes.

Architecture and Domain Specific Qualities - Quality attributes specific to the architecture itself and Quality attributes found in specific business domains.

A quality attribute scenario consists of six parts:

1. stimulus - the condition that affects the system
2. response - the activity that results from the stimulus
3. source of stimulus - the entity that generated the stimulus
4. environment - the condition under which the stimulus occurred
5. artifact stimulated - the artifact that was stimulated
6. response measure - the measure by which the system's response will be evaluated

The design time = static / development time / code time

Scenario Name:		
Scenario(s):	The system provides failover on the Servlet Container	
Business Goal(s):	fluently handle general failures	
Relevant Quality Attributes:	Availability	
Scenario Components	Stimulus:	Servlet container failure
	Stimulus Source:	Internal (unspecified)
	Environment:	Normal operating environment (runtime)
	Artifact (If Known):	A Servlet container failure
	Response:	Detect a fault, perform provide failover, and recover fully
	Response Measure:	Detect a fault within 1 second, provide failover within 15 seconds, and fully recover from failure within 2 minutes
Questions / Risks / Issues:		

Fig 1: Sample Scenario

4. EMPIRICAL STUDY

At the outset a theoretical survey was done and these widely used standard Quality attributes are identified. These Quality attributes are constant or varies according to the environment. Software Quality can be improved by considering the best possible parameters in Architecture Review [7]. To fine tune the process and to get the realistic status of the essential parameters in evaluating Software architecture, the study was conducted in twenty five large software firms, to identify the essential and high priority quality attributes in evaluating software architecture and preferences of quality attributes were captured from software architects. The table-1 shows the opinion poll results.

Table-1 Quality Attributes

Quality Attribute	**Its Role**
Dependability	It is a measure of confidence that the module is free from errors.
Security	It is a measure of the ability of the module to resist an intrusion.
Adaptability	It is a measure of the ability of the module to tolerate changes in resources and stakeholder's requirements.
Maintainability	It is a measure of the ease with which a software system can be maintained.
Portability	It is a measure of the ease with which a module can be migrated to a new environment.
Throughput	It indicates the efficiency or speed of a module.
Capacity	It indicates the maximum number of concurrent requests a module can serve.
Turn-around Time	It is a measure of the time taken by the module to return the result.
Parallelism Constraints	It indicates whether a module can support synchronous or asynchronous invocations.
Availability	It indicates the duration when a module is available to offer a particular service.
Ordering Constraints	It indicates the order of returned results and its significance.
Evolvability	It indicates how easily a module can evolve over a span of time.
Result	Indicates the quality of the results returned.
Achievability	It indicates whether the module can provide a higher degree of service than promised.
Priority	It indicates if a component is capable of providing prioritized service.
Presentation	It indicates the quality of presentation of the results returned by the module.

Architectural decisions directly impact system qualities, and often a decision in favor of one quality has an impact on another. Quality goals can primarily be achieved if software architecture is evaluated with respect to its specific quality requirements before its implementation [8].

The table - 2 shows the respondents' opinion about the quality attributes in software architecture evaluation. Though many quality attributes are considered to maintain the quality of the software, these twelve attributes are found widely used in practice in most of the software developing firms. Table- 3 shows the mean weight and the rank of the quality attributes in software architects view. From the table it is clear that security is the one important quality attribute which holds the first rank among the attributes. 64% of the respondents express security is the very highly required quality parameter.

Table-2 Software architects opinion on Quality Attributes

Attributes	Very High		High		Medium		Low		Very Low	
	Frequency	Percent	Frequency	Percent	Frequency	Percent	Frequency	Percent	Frequency	Percent
Dependability	6	24.0	15	60.0	4	16.0	0	0	0	0
Security	16	64.0	7	28.0	2	8.0	0	0	0	0
Adaptability	8	32.0	10	40.0	7	28.0	0	0	0	0
Portability	0	0	23	92.0	1	4.0	0	0	1	4.0
Parallelism constraints	3	12.0	7	28.0	12	48.0	3	12.0	0	0
Ability	6	24.0	16	64.0	3	12.0	0	0	0	0
Ordering constraints	2	8.0	11	44.0	9	36.0	2	8.0	1	4.0
Evolvablity	2	8.0	12	48.0	11	44.0	0	0	0	0
Achievability	6	24.0	16	64.0	2	8.0	1	4.0	0	0
Priority	8	32.0	16	64.0	1	4.0	0	0	0	0
Presentation	4	16.0	18	72.0	2	8.0	1	4.0	0	0
Learnability	5	20.0	14	56.0	5	20.0	0	0	1	4.0

Table-3 Weighted Value and rank of Quality Attributes

Attributes	Very High			High			Medium			Low			Very Low			Mean Weight	RANK
	Frequency	Weight	Percent	Frequency	Weight	Percent	Frequency	Weight	Percent	Frequency	Weight	Percent	Frequency	Weight	Percent		
Dependability	6	30	24	15	60	60	4	12	16	0	0	0	0	0	0	4.08	4
Security	16	80	64	7	28	28	2	6	8	0	0	0	0	0	0	4.56	1
Adaptability	8	40	32	10	40	40	7	21	28	0	0	0	0	0	0	4.04	5
Portability	0	0	0	23	92	92	1	3	4	0	0	0	1	1	4	3.84	8
Parallelism constraints	3	15	12	7	28	28	12	36	48	3	6	12	0	0	0	3.4	11
Ability	6	30	24	16	64	64	3	9	12	0	0	0	0	0	0	4.12	3
Ordering constraints	2	10	8	11	44	44	9	27	36	2	4	8	1	1	4	3.44	10
Evolvablity	2	10	8	12	48	48	11	33	44	0	0	0	0	0	0	3.64	9
Achievability	6	30	24	16	64	64	2	6	8	1	2	4	0	0	0	4.08	4
Priority	8	40	32	16	64	64	1	3	4	0	0	0	0	0	0	4.28	2
Presentation	4	20	16	18	72	72	2	6	8	1	2	4	0	0	0	4	6
Learnability	5	25	20	14	56	56	5	15	20	0	0	0	1	1	4	3.88	7

The quality attribute, priority is in second highest place and the ability takes position three. The rank is calculated for the quality attributes based on its mean weight. In this way the quality attributes are ranked and helpful in fixing the quality of the software product and it is highly recommended by the software architects to use these quality attributes in early architecting stage of software development. The results of the study show that the systematic use of quality attributes in software architecture evaluation as a part of the software design process will improve the quality of outcome. The main aim associated with this study is that, to find the essential and key quality attributes in architects view. The rank column of table-2 shows priority of quality attributes in practice.

5. CONCLUSION

In this paper, the importance of quality attributes in evaluating software architecture evaluation is presented. To get the optimum results in identifying quality attributes, a review have been conducted in large software firms and it is presented in the table. The results of this study may serve as a roadmap to the software architects in helping them to select the right quality attributes in evaluating software architectures.

6. REFERENCES

1. Recommended practice for architectural description. IEEE Standard P1471, 2000.
2. D. Garvin. What Does Product Quality" Really Mean. Sloan Management Review, pp. 25-45, 1984.
3. B. W. Boehm, J. R. Brown, H. Kaspar, M. Lipow, G. McLeod, and M. Merritt. Characteristics of Software *Quality*, North Holland, 1978.
4. J. McCall, P. Richards, and G. Walters. *Factors in software quality*. Vol I-III, Rome Aid Defence Centre, Italy, 1997.
5. ISO, International Organization for Standardization, *ISO 9126-1:2001, Software engineering - Product quality, Part 1: Quality model*, 2001.
6. 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.
7. P.Shanmugapriya , R.M.Suresh, The Impact of Knowledge Based Conceptual Model in Evaluating Software Architecture, International Journal of Computer Science and Communication, Volume-2, Number-1, 2012. ISSN: 0973-7391.
8. P. Clements and R. K. Kazman, M. Klein. Evaluating Software Architectures: Methods and Case Studies. Addison-Wesley Professional; 2002. ISBN 0-201-70482X