J. Appl. Environ. Biol. Sci., 6(4S)63-67, 2016

ISSN: 2090-4274
Journal of Applied Environmental and Biological Sciences
www.textroad.com

Collaboration Methodology for Integrating Non-Functional Requirements in Architecture

Shahzad Khan[1], Muhammad Babar[2], Fazlullah Khan[1*], Fahim Arif[2], Muhammad Tahir[1]

[1]Department of Computer Science, Abdul Wali Khan University Mardan, Pakistan
[2]Department of Computer Software Engineering, National University of Sciences & Technology, Islamabad, Pakistan

Received: January7, 2016
Accepted: March 22, 2016

ABSTRACT

Architecture addresses the Non-functional requirements (NFRs) that are the qualities of the system. Functional requirements (FRs) are being taken under consideration at the early stage of software process development while NFRs are not taken into account. Usually, the NFRs are being focused at the end of the project, which does not fulfill the desired qualities. Early design decision is very important to achieve a strong connection between design and requirements, quality of a system and a consistent software product. The position which we put forward, the NFRs should be focused at architectural. Both runtime NFRs (such as performance, security and fault tolerance) and some of those which are not runtime (such as maintainability) should be considered at the architectural level. In this paper, we present a collaboration methodology to integrate the NFRs and Architecture. The proposed methodology focuses on the close collaboration of architect and analyst of the system. Specially, we elaborate this collaboration in form of the involvements of architect into the requirements.

KEYWORDS: Non-Functional Requirements; Architecture; Software Design; Software Requirements.

1. INTRODUCTION

Usually, NFRs are rarely taken under consideration in most of the software development processes. It is rarely considered due to their very high abstraction level, less support of tool and languages, complexity, and informality [1]. NFRs have high abstraction level because they cannot be expressed quantitatively. Such as performance, fault tolerance, availability, maintainability etc. Functional requirements are being incorporated into software architecture at early stages of process, at the end of process all of them are implemented to satisfy the requirements defined at early stages [2]. Most of the methods, techniques, languages and tools are introduced to capture the functional requirements. NFRs are not taken into account at the early stage of the software development process, that effect the system Software architecture decision focuses and pay primary attention to NFRs [3]. The architecture depicts the FRs and NFRs. Early design decision is required to strengthen the connection between requirements and design [4],[8].

The architectural decision taken for an application affects the NFRs. If performance is critical issue, we need to localize critical operations within a small number of subsystems with minimized inter communication. If security is demanded, we need to use a layered architecture. Similarly, if safety is important, we need to deploy small number of subsystems. If the focus of the application is on the maintainability, we need to use fine-grain, self-contained and replaceable components. NFRs cannot be evaluated without looking at the system as a whole because NFRs are described as attributes of the system that contributes to the overall quality of the product. This is the evident of the fact that the NFRs are very complex. On the other hand it is also evident that non-functional requirements are also very important since they contribute to the overall quality of the resulting system [5]. Requirements and architectural design in the early stage of the software process development is the demand of today. It is due to the increasing size, complexity, distribution and heterogeneity [6][9]. If it is dealt altogether at early stage, the result will be satisfactory and fulfilling the customers' demand. FRs, NFRs must be realized through the architecture [10,17,18].

Non-functional requirements compete and conflict each other (such as maintainability and performance). Using large-grain components improves performance but reduces maintainability. Similarly, introducing redundant data improves availability but makes security more difficult. If both performance and maintainability is required, we need a compromised solution. For example to compromise with availability, we need to sacrifice in rush hours and include some down time. A tradeoff is needed to involve finding an optimal solution. The development of large software systems is the need of almost every organization. Because of the involvement of the non-functional requirements, software architecture design has become an important step in large software development [11,14,15,16]. In this paper we define the methodology to have better communication between the architect who build the architecture of the system and the analyst who is responsible for the requirements gathering. We also put forward the issue that NFRs should be focused at architectural level and should be described properly through using some quantitative method

[*] **Corresponding Author:** Fazlullah Khan, Department of Computer Science, Abdul Wali Khan University Mardan, Pakistan
fazlullah@awkum.edu.pk

such as associating some metrics. Our proposed methodology gives an idea to collaborate the architect and analyst. We also elaborate the proper communication between architect and analyst.

The rest of the paper is as follow: Section 2 discusses the related work of integrating the NFRs and architecture. In section 3 we introduce our proposed collaboration methodology between architect and analyst. Finally, section 5 concludes and gives future work.

2. RELATED WORK

Software architecture is the first step after the requirements elicitation of software development process. The basic purpose of the architecture is to ensure the productivity of the functional requirements (FRs) at the end of the product. The non-functional requirements (NFRs) are usually focused at the end of the product due to which the qualities of the system are affected. A number of techniques are studied to deal both the architecture and NFRs together in order to achieve the functionalities and qualities at the end of the product. The focus of technique may vary from each other, for example one technique may provide some practical solution, whereas another technique may only argue the issue or provide a framework to cater the issue of integration. Parmenides framework defines how to deal with NFRs within the software development process. The framework describes the NFRs, their refinement, mapping into actual implementation, and integration with functional requirements (FRs). The framework also defines precisely how NFRs are expressed and integrated into an architectural-based development. Furthermore, it defines two languages for describing NFRs, an integration strategy, a set of refinement rules and a mapping strategy [1]. The transactional and non-functional requirements (NFRs) are formally incorporated into dynamic software architecture. An appointment system is also proposed in order to demonstrate how this approach can be utilized in a real application. Furthermore, three NFRs properties have been chosen to focus which are safety, availability and performance [2]. Similarly, use case-based approach to describe NFRs. The approach is based on the concepts of architectural policies. This approach is used to employ use cases and scenarios to describe non-functional requirements [3]. GRL (Goal Oriented Requirement Language) is a language for supporting goal and agent oriented modeling and reasoning of requirements, especially for dealing with NFRs. A UCM (a scenario oriented architectural notation) is also presented. Goals are used in the refinement of non-functional (NFRs) and functional requirements (FRs), exploring alternatives, their operationalization into architectural constructs [4]. CBSP (Component, Bus, System and Property is used to reconcile software requirement and architecture. Furthermore, CBSP minimize the gap between high level requirement and architectural descriptions. CBSP also allows identifying and isolating 'ilities' for the purpose of improving nun-functional properties and allows capturing and maintaining arbitrarily complex relationship between requirement and architectural artifacts [6,12,13]. The framework lies on an XML-based integration core and semantics relation between the models are represented [8,19,20]. The process of integrating FRs, NFRs and architectural options (AO) should be fundamentally based on experience. The authors have presented a comprehensive approach to convert the major issues related to the FRs and NFRs and AOs. The proposed approach supports the elicitation, specification and design activity [9]. The FRs, NFRs and architecture must be focused together, because they constraint each other. In addition, the architecture addresses the NFRs in the early stage of design. The FRs, NFRs and architectural design must be developed in a tightly integrated approach [1].

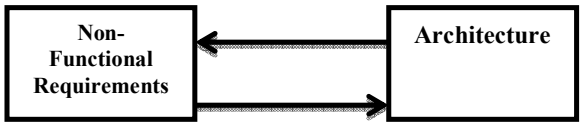


Figure 3.1: Relationship between NFRs and Architecture

3. PROPOSED COLLABORATION METHODOLOGY.

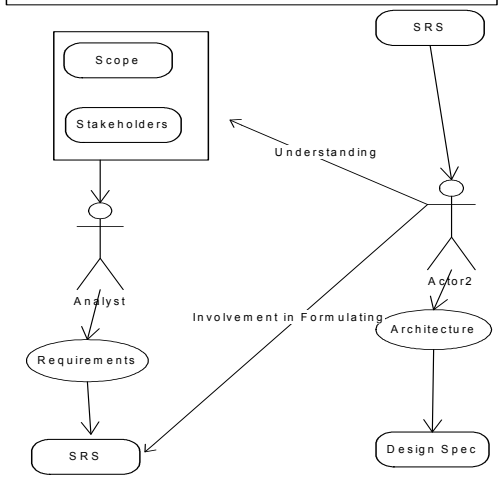
Non-functional requirements reflect the architecture, and in the same way NFRs are reflected by the architecture. Figure 1 depicts this relationship of NFRs and architecture. According to Figure 3.2 if software is desired to achieve certain NF attributes (such as maintainability, performance, reliability), then the architecture needs to be formulated accordingly. Similarly, a specific architecture (which is being formulated) depicts specific non-functional attributes which interm are the qualities of a system. If certain architecture is designed then the specific non-functional attributes of the desired system will be reflected.

Requirements are gathered and analyzed by analyst. He is responsible for thoroughly eliciting the requirements. He is also responsible for the categorization of functional requirements and non-functional requirements. Analyst thoroughly does the analyses of the NFRs. After properly handling NFRs, finally the analyst specifies the requirements in a form of a document. On the other hand, architect performs the design activities of the software system.

The design starts from the high level design which is also called architecture. He gets start form the requirements document. The problem we face, is that the architect does not have any sort of involvement in formulating this document. In order

to integrate the NFRS and architecture, a close collaboration among architect and analyst is needed. This close collaboration provides a strong connection of NFRs and architecture.

Figure 3.2 Overall Collaboration Methodology



Our methodology of close collaboration helps the architect to get understanding of the requirements. It promotes architect to have a thorough comprehension of the stakeholders and their needs. Furthermore, it tells about the importance of stakeholders and scope understating for the architect. The proposed methodology of collaboration is achieved through involvement of architect into the analysis activities.

3.1 Architects Involvement in Requirement Document

Architect should collaborate with the analyst in formulating the final draft of requirements. Software requirement specification is the final draft of requirements. SRS includes the overall requirements captured and analyzed by the analyst. Therefore a strong involvement of the architect is needed. This scenario is depicted in figure 3.2.

3.2 Architect Awareness of Stakeholders

In order to have a strong connection between architecture and the non-functional requirements, the architect must be aware of the major stake holders and their needs. If architect is aware of the stakeholders and their needs, then he would design the architecture accordingly. In figure 3.2 the arrow towards the stakeholders shows that the architect gets involved into the stake, and the requirements and goals of the stakeholders.

3.3 Architect Awareness of the Scope Statement

The overall objective and the acceptance criteria is mentioned in the scope statement. It also depicts some of the NF attributes. The architect must be aware of this scope in order to build a good architecture that reflects the desired NFRS accordingly. Arrow towards scope statement in the figure 3.2 depicts the importance of the understanding of the scope statement in order to have a sophisticated architecture.

4. Conclusion and Future Work

Software architecture is widely used to address the functional requirements of the application. A number of techniques, methodologies, languages, tools, and processes are introduced to cater the functional requirements of the software. An important consideration is that, the software architecture addresses the non-functional requirements as well. Specially, runtime qualities of the software system (such as performance, security, availability and fault tolerance) are thoroughly addressed by the architecture. In this paper, we proposed a methodology to collaborate between architect and analyst. This collaboration is from the architect side mostly. We conclude that non-functional requirements should also take into account at architectural level. Functional requirements and nun-functional requirements must be deal together in order to achieve better interconnection between design and requirements.

We also highlight some of the future intentions. The test and inspection of NFRs is very important. Different people might have different expectations from NFRs, so they are needed to be tested and expected properly. This testing fan inspection can only be possible, if we describe the NFRs properly. NFRs could be described using the proper quantitative method which is indented for the future work.

REFERENCES

[1] C Lopez, H Astudillo, Use-case and Scenario-based Approach to Represent NFRs and Architectural Policies, 2nd International Workshop on Use Cases, 2005, Citeseer

[2] Nelson S. Rosa , George R. R. Justo , Paulo R. F. Cunha, A framework for building non-functional software architectures, Proceedings of the 2001 ACM symposium on Applied computing, p.141-147, March 2001, Las Vegas, Nevada, United States

[3] Nelson S. Rosa, George R.R. Justo and Paulo R.F. Cunha, Incorporating Non-functional Requirements into Software Architectures, Springer Berlin / Heidelberg, Saturday, January 01, 2000,

[4] Burge, JE and Brown, DC: 2002, NFRs: Fact or Fiction?, Technical Report WPI-CS-TR-02-01, Computer Science Department, WPI.

[5] Liao, L. From Requirements to Architecture: The State of the Art in Software Architecture Design, http://www.cs.washington.edu/homes/liaolin/Courses/architecture02.pdf

[6] P. Grunbacher, A. Egyed, N. Medvidovic, "Reconciling Software Requirements and Architectures: The CBSP Approach", Proceedings of the 5th International Symposium on Requirements Engineering (RE'01), pp.202-211, IEEE CS Press, 2001.

[7] Paech, B., von Knethen, A., Doerr, J., Bayer, J., Kerkow, An experience based approach for integrating architecture and requirements engineering ", accepted for ICSEworkshop STRAW 2003

[8] Software Engineering Body of Knowledge (SEBOK) 2004.

[9] V. Cortellessa, A. Marco, P. Inverardi, F. Macinelli, and P. Pelliccione. A framework for the integration of functional and non-functional analysis of software architectures. In TACoS, 2004.

[10] L. Liu and E. Yu, From requirements to architectural design –Using goals and scenarios, in: From Software Requirements to Architectures Workshop (STRAW 2001), Toronto, Canada, May 2001.

[11] B. Paech, A. Dutoit, D. Kerkow, A. von Knethen, Functional requirements, non-functional requirements and architecture specification cannot be separated – A position paper", REFSQ 2002.

[12] M. A. Jan, P. Nanda and X. He, "Energy Evaluation Model for an Improved Centralized Clustering Hierarchical Algorithm in WSN" in Wired/Wireless Internet Communication, Lecture Notes in Computer Science, pp. 154–167, Springer, Berlin, Germany, 2013.

[13] M. A. Jan, P. Nanda, X. He and R. P. Liu, "Enhancing lifetime and quality of data in cluster-based hierarchical routing protocol for wireless sensor network", 2013 IEEE International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing (HPCC & EUC), pp. 1400-1407, 2013.

[14] F. Khan, K. Nakagawa, Faisal Bashir, "Dual Head Clustering Scheme in Wireless Sensor Networks" in the ICET , 2012.

[15] F. Khan, K. Nakagawa, "Comparative Study of Spectrum Sensing Techniques in Cognitive Radio Networks" in World Congress on Computer and Information Technology, 2013, pp.1-8

[16] M. A. Jan, P. Nanda, X. He and R. P. Liu, "PASCCC: Priority-based application-specific congestion control clustering protocol" Computer Networks, Vol. 74, PP-92-102, 2014.

[17] Mian Ahmad Jan and Muhammad Khan,, "A Survey of Cluster-based Hierarchical Routing Protocols", in IRACST– International Journal of Computer Networks and Wireless Communications (IJCNWC), Vol.3, pp.138-143, April. 2013.

[18] Mian Ahmad Jan and Muhammad Khan, "Denial of Service Attacks and Their Countermeasures in WSN", in IRACST– International Journal of Computer Networks and Wireless Communications (IJCNWC), Vol.3, April. 2013.

[19] M. A. Jan, P. Nanda, X. He and R. P. Liu, "A Sybil Attack Detection Scheme for a Centralized Clustering-based Hierarchical Network" in Trustcom/BigDataSE/ISPA, Vol.1, PP-318-325, 2015, IEEE.

[20] M. A. Jan, P. Nanda, X. He, Z. Tan and R. P. Liu, "A robust authentication scheme for observing resources in the internet of things environment" in 13th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 205-211, 2014, IEEE.

[21] F. Khan, S.A. Kamal, F. Arif, "Fairness Improvement in long-chain Multi-hop Wireless Ad hoc Networks" in IEEE ICCVE 2013, Las Vegas, USA 2-6 December, 2013

[22] F. Khan "Secure Communication and Routing Architecture in Wireless Sensor Networks" in the 3rd Global Conference on Consumer Electronics (GCCE) Tokyo, Japan: IEEE Tokyo, 2014.

[23] F. Khan "Throughput & Fairness Improvement in Mobile Adhoc Networks" in the 27th Annual Canadian Conference on Electrical and Computer Engineering, Toronto, Canada: IEEE Toronto, 2014.

[24] S. Khan, F. Khan. "Delay and Throughput Improvement in Wireless Sensor and Actor Networks" in the 5th National Symposium on Information Technology: Towards New Smart World (NSITNSW) Riyadh: IEEE Riyad Chapter, 2015.