IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

AN EFFICIENT FRAMEWORK TO IDENTIFY QUALITY ATTRIBUTES AND TO PROVIDE FLEXIBILITY IN DEVELOPMENT

Shimontini Majumder, Suchandra Nath Chakraborty, Anirban Bhar[*]

B. Tech student, Department of Information Technology, Narula Institute of Technology, Kolkata, India
B. Tech student, Department of Information Technology, Narula Institute of Technology, Kolkata, India
[*]Assistant Professor, Department of Information Technology, Narula Institute of Technology, Kolkata, India

DOI: 10.5281/zenodo.56901

ABSTRACT

Non Functional Requirements (NFRs) are gaining importance for high quality software development and also gaining interests in the competitive market. To enhance the quality of a software project along with its quantity, a framework is necessary to identify the correct NFRs and implement them in the appropriate way to the architectural models. This work provides a framework to identify and measure the appropriate NFRs thoroughly; those are successfully carried out on a case study to validate them. Effective measurement techniques are used to identify and implement NFRs appropriately, that will not only provide flexibility in the development cycles but also increases the cost effectiveness of the project.

KEYWORDS: Non Functional Requirement (NFR), Quality Attributes (QA), Development Life Cycle, Architectural Model, Decision Table.

INTRODUCTION

A system's utility is determined by both of its functional as well as non-functional characteristics. In most of the developed softwares,due to quickly having a running system with the basic necessity, quantitative or functional requirements are taken in count but the quality factors are mostly overlooked the requirement phase. To ascertain their quality of measure, they are to be examinedand analyzed properly. Non Functional Requirements (NFRs) are those requirements to state the quality factors of a software and are going to be one of the major requirement to software modeling in near future. They are also termed as the Quality Attributes (QA). Few among the QAs are Accessibility, Reliability, Modifiability, Security.

Although the Quality Attributes (QAs) are an important part of the requirement analysis for any model, they gain the minimum focus. The reason behind this is people are more concerned about the present requirements to fulfill their needs. They want a model that will solve their immediate problems but does not care about what may happen after a few years or may be a few months! If the model doesn't work properly after a period of time then the whole effort is wastage as well as causes cost inefficiency that is not at all desirable in the competitive market.

RELATED WORK

Quality Attributes do affect the lives of people at some point of time. So they need to be taken care of by rightly examiningand analyzing them and properly [4][8] implementing them into the models. There have been a lot of research before on this identification process [5][6][9][10], those were mainly based on some assumptions, but not measurements. These works make the use of methods to make sure design decisions on the inclusion or exclusion of requirements which will have effect on the architectural design of the software. Unlike the metrics approaches, these latter approaches are concerned about making NFRs a relevant and important part of the software development process. Boehm and In propose a knowledge base [6] where NFRs or QAs are given the priority through an Associates' perspectives, dealing with NFRs at a high level of abstraction. The NFR Framework views NFRs as goals that might conflict among each other and must be represented as soft goals to be satisfied [11]. The soft goal concept was introduced to cope with the abstract and informal nature of NFRs. Each soft goal will be decomposed into sub-goals represented by a graph structure inspired by the and trees used in problem solving. This process continues until the requirements engineer considers the soft goal satisfied The other methods used widely are Scenario-based analysis [13].

THE MODEL

This model will help to understand the importance, identify as well as implement the QAs in the development of a model on a measurement basis [12]. It consists of mainly two Phases: The Identification Phase of QA and The Implementation Phase of QA.

The Identification Phase of QA

The main objective of the developing or developed model should be that it must provide the required services to its users. As an added part comes its validation. Then comes how certain can the services be, rather the uncertainty of the services. And lastly comes how it can be measured.

a) Validation of QAs- The following statements show how the QA can be validated:
- Each requirement of the QA must have a source.
- The QA mentioned must not be fictitious, rather must be attainable.
- They must be confined to testing when implemented.
- The requirements should be independent, if dependent they must not be in conflict to each other.
- They must cause an overall gain to the system.
- There must be clarity in the QAs being suggested, no scope for misconceive.

b) Uncertainity of QAs- A decision table is to be constructed to find the uncertainties that should not be possessed by the Qas. Table 1 represents a sample decision table.

Table 1. ***For determining the ascertain QAs***

	QA1	**QA2**	**QA3**	**QA4….**	**QAn**
Task 1	X			X	
Task 2		X			X
Task 3	X				
….					
Task n					X

From Table 1 those QAs that have no 'X' marks at all, they are the uncertain QAs and they are to be put aside from being implemented. Whereas all the QAs that have at least one 'X' mark are the ascertain QAs, which are to be implemented in our process.

c) Measurement of QAs- For the measurement of the QAs we are using the formula suggested by Davis [3] for the quality measurement of the model. The formula is given by

$$\text{MCR \%} = (N_C / N_T) * 100$$

Where N_C = number of ascertain QAs

N_T = total number of QAs (ascertain QAs + uncertain QAs)

The Implementation Phase of QA

After successful identification of the Quality Attributes, they are implemented on any of the Development Life Cycles [1][2][11][14] for developing a model. This mainly involves two sub-phases:

a) Testing Flowchart-

This is basically a sub-process which is used as a part of every revised model. This consists of a very simple flowchart, which helps in determining the steps that leads to the implementation of the correctly identified QAs into the model. It is a step by step process where each step is followed only after completion of the previous step. Here all the steps are dependent on each other, so has an impact on each other as well. The steps involvedin the flowchart are as follows:

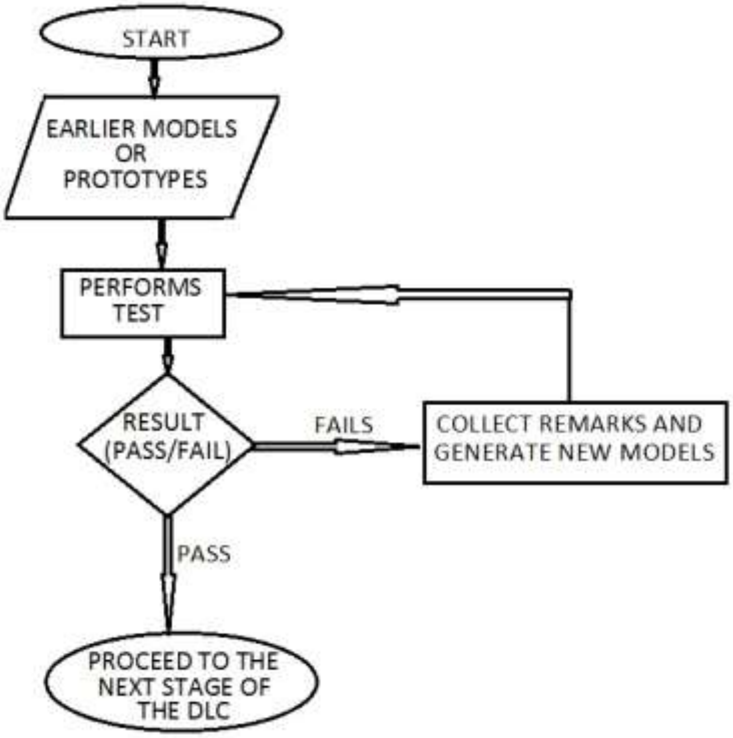

Figure 1. ***The Testing Flowchart***

All the prepared models performs the test of going through this flowchart to proceed to the next stage of its Life Cycle. It is allowed to pass on to the next stage if and only if it passes the test. Otherwise it is being reconstructed again and is made to perform the test again.

b) Revised Model-

This a process of revision of the prepared models or prototypes, that is followed by the flowchart testing, being again followed by this phase itself. It becomes a cycle of the two previously mentioned phases- the Testing Flowchart and the Revised Model one after another unless the model is completely ready for usage. Figure 2 represents a diagrammatic view of this stage to help to understand it in a better way.

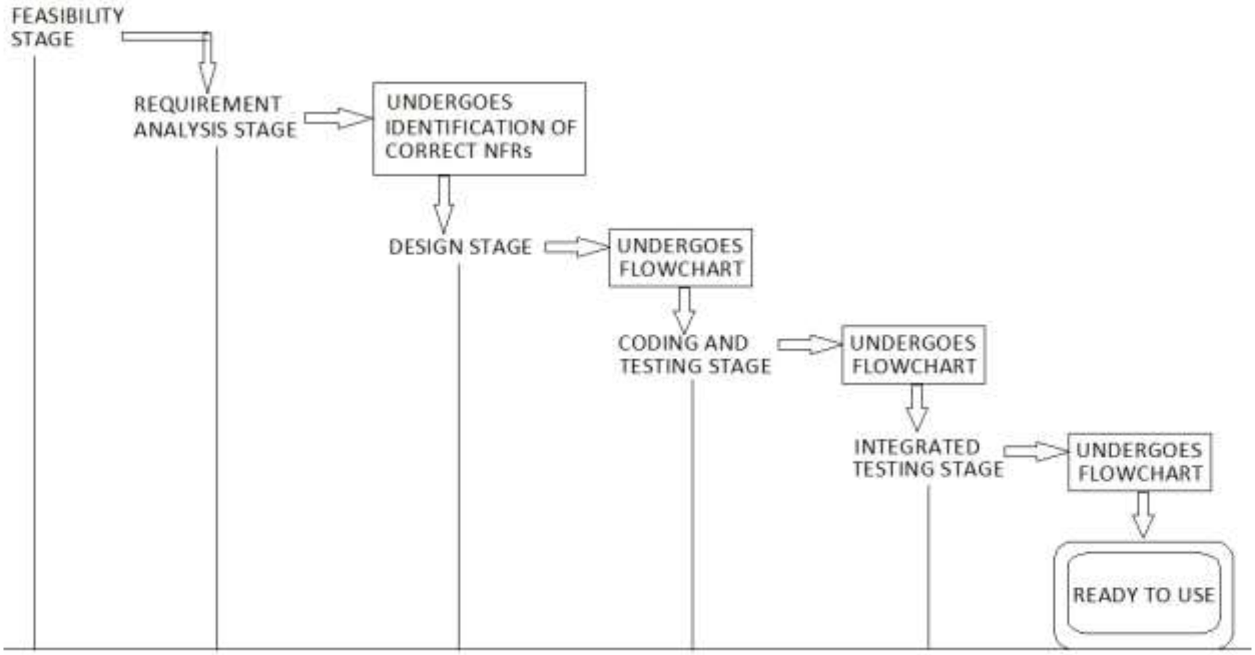

Figure 2. ***The Identification method and the Testing Flowchart in the developmental life cycle of a model***

This phase has six stages such as:

- Feasibility stage
- Requirement analysis stage
- Design stage
- Coding and Testing stage
- Integrated testing stage
- Ready to use

The above six stages are a common part of every Development Life Cycle, the only difference being, here each of the stages are followed by a particular method. The feasibility stage being similar to all the other life cycle stages, the requirementanalysis phase is followed by the identification of the correct NFRs or QAs method.

The Design stage, the Coding and Testing stage, and the Integrated testing stage, each of the three stages are followed by the Testing Flowchart method mentioned above. This enables the identification of the QAs at the right stage, and their implementation as well as checking after each stage completion, whether the Quality Attributes are retained in the model or not. If retained they are carried on to the next stage until the last stage. If not they are reclaimed and reconstructed to go through the implementation stages correctly. The model we get as an output from this whole process is believed to have retained all the NFRs or Quality Attributes as a part of the software and they can be used along in the future.

A RESTAURANT SYSTEM THAT GIVES THE SERVICES FOR BILLING AND ACCEPTING ORDERS: A CASE STUDY FOR THE IDENTIFICATION PHASE OF QA

It is a software model that has three main Associates or operators namely the Admin, the Customer and the Staff. The model enables a customer to perform three tasks that is he can place his required order, can cancel the already placed order, and can check the status for his order.

This model enables a staff to perform two tasks that is he can check the status of the bill or order placed by any customer, and can serve or redeem the order placed by any customer.

This model also enables the Admin that is the owner of the model to look after each and every task performed by a staff or a customer and can control them from anywhere he wants, accordingto his demand or wish. He can also keep track of the records of his restaurant and monetary affairs. The identification process is as follows-

The Associates-

- Admin
- A customer
- A staff

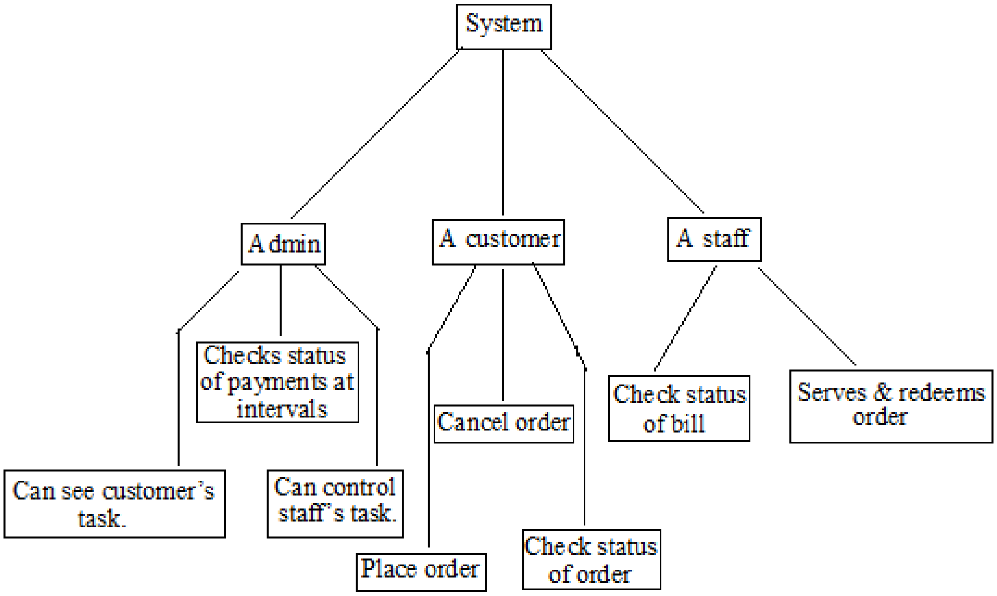

Figure 3. ***A Tree structure that displays the levels of a system***

The required QAs:

- Admin-
  - Can see customer's task (Privacy)
  - Checks status of payments at intervals (Portability, Scalability)
  - Can control staff's task(Visibility)

- A Customer-
  - Place order (Security)
  - Cancel order (Reuseability)
  - Check status of order (Privacy)

- A Staff-

- Check status of bill (Stability)
- Serves & redeems order (Robustness)

Table 2. ***Different divisions and tasks of the system***

System	A Restaurant System.
Associates	Admin, A Customer, A Staff.
Tasks	1.Can see customer's task, 2.Checks status of payments at intervals, 3.Can control staff's task, 4.Place order ,5.Cancel order, 6.Check status of order ,7.Check status of bill ,8.Serves & redeems order
Quality Attributes	Privacy,Portability, Scalability,Visibility,Security,Reuseability, Stability, Robustness.

Table 3. ***Decision table for determining the possible QAs***

	Privacy	Portability	Scalability	Visibility	Security	Reuseability	Stability	Robustness
Task 1	X							
Task 2		X	X					
Task 3				X				
Task 4					X			
Task 5						X		
Task 6	X							
Task 7							X	
Task 8								X

CALCULATION AND DISCUSSION

Formulae:

MCR %= (N_C/ N_T) *100

Where N_C = number of ascertain QAs

N_T = total number of QAs (ascertain QAs + uncertain QAs)

So, MCR% = (8/8)*100
=100%

The Quality Attributes are checked against each sub-divided task of the associates, and all of them are found to be valid. So in this case particularly the possible identified QAs are:-
Privacy, Portability, Scalability, Visibility, Security, Reuseability, Stability, Robustness.

CONCLUSION AND FUTURE WORK

As a concluding part for this paper we must say that , all these processes performed step by step will help one to locate all the possible NFRs , will increase the efficiency of the model, will be cost effective , and can be used with any of the development processes which provides a flexibility to the developers.

As a future scope we want more research work to be done on this topic as for example , how can such a framework be made much more cost effective, as well as time effective. The measurements can be made more accurate based on the technical aspects of the model to be developed. Measurable NFRs can provide the proof to achieve the goals of application oriented software systems. We have plan to concentrate more on the above mentioned matters and will produce some more frameworks, which will have experienced inputs and an overall optimization of the whole developmental cycle.

REFERENCES

[1] Pressman, R. S. Software Engineering: A Practitioner's Approach (6th ed.). New York:-McGraw-Hill Publication, 2005.

[2] Anirban Bhar, Narula Institute of Technology & Dr. Sabnam Sengupta, B.P.P.I.M.T, Kolkata, “Proto-Spiral: A Hybrid SDLC Model for Measuring Scalability Early in Development Using a Probabilistic Approach”, ETCC 2014.

[3] A. Davis, S. Overmyer, K. Jordan, J. Caruso, F. Dandashi, A. Dinh, G. Kincaid, G. Ledeboer, P. Reynolds, P. Sitaram, A. Ta, M. Theofanos “Identifying and measuring quality in a software requirements specification”, First International Software Metrics Symposium, 1993.

[4] Pratima Singh and Anil Kumar Tripathi, “Treating NFR as First Grade for Its Testability”. Journal of Software Engineering and Applications, 2012, 5, 991-1000.

[5] Chung L., Nixon B. “Dealing with Non-Functional Requirements: Three Experimental Studies of a Process-Oriented Approach” Proc. 17th Int. Con. on Software Eng. Seatle, Washington, April pp: 24-28, 1995.

[6] Boehm, Barry and In, Hoh. “Identifying Quality-Requirement Conflicts”. IEEE Software, March 1996, pp. 25-35.

[7] Breitman, K. K, Leite J.C.S.P. and Finkelstein Anthony, “The World's Stage: A Survey on Requirements Engineering Using a Real-Life Case Study”, Journal of the Brazilian Computer Society No 1 Vol. 6 Jul. 1999 pp:13:37.

[8] Chung L., “Representing and Using Non-Functional Requirements: A Process Oriented Approach”, Ph.D. Thesis, Dept. of Comp.. Science. University of Toronto, June 1993. Also tech. Rep. DKBS-TR-91-1.

[9] Lindstrom, D.R. “Five Ways to Destroy a Development Project”, IEEE Software, September 1993, pp. 55-58.

[10] Cysneiros L.M., Leite, J.C.S.P. and Neto, J.S.M. “A Framework for Integrating Non- Functional Requirements into Conceptual Models” Requirements Engineering Journal — Vol 6 , Issue 2 Apr. 2001, pp:97-115.

[11] Anirban Bhar, Narula Institute of Technology & Dr. Sabnam Sengupta, B.P.P.I.M.T, Kolkata, “Soft Testing Life Cycle: A guideline to model, analysis and verify NFR”, International Journal of Latest Trends in Engineering and Technology (IJLTET), Volume 6, Issue 3, pp. 271-276 in January 2016.

[12] Gopichand Merugu and Anandarao.Akepogu, “Four Layered Approach to Non-Functional Requirements Analysis”, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 6, No 2, November 2011, pp. 371-379.

[13] Leite J.C.S.P. et.al. “Enhancing a Requirements Baseline with Scenarios”, Requirements Engineering Journal, 2(4):184-198, 1997.

[14] R.S Pressman , “Software Engineering: A Practitioner's Approach”, McGraw-Hill; 5th Edition edition, 2000.