*Journal of Software Engineering and Applications*, **2012, 5, 991-1000**
http://dx.doi.org/10.4236/jsea.2012.512114 Published Online December 2012 (http://www.SciRP.org/journal/jsea)

# Treating NFR as First Grade for Its Testability

**Pratima Singh, Anil Kumar Tripathi**

Department of Computer Engineering, Indian Institute of Technology (Banaras Hindu University), Varanasi, India.
Email: pratima.singh.rs.cse@itbhu.ac.in, aktripathi.cse@itbhu.ac.in

Received October 20th, 2012; revised November 22nd, 2012; accepted December 2nd, 2012

## ABSTRACT

Practitioners and researchers in the field of software engineering have realized that Non-Functional Requirements have not received due attention and second grade (or no) treatment has been meted out to Non-Functional Requirements. Many software products/systems are finally not acceptable because of such an approach. This casual approach of treating NFR has moved on to Testing also. Testing of NFR has never been taken seriously. Here in this work, we attempt to understand what needs to be done for proper consideration of NFR, so that they are treated as seriously as the Functional Requirements. In an attempt to treat NFR as seriously as FR we work on the testability of NFR by refining an abstract quality concern into concrete NFR statements. We show that quality concerns needs to be analyzed, for identifying and finally converting them into appropriate and unambiguous NFR. Once a high quality of NFR is ensured then the consequent testing of these NFRs will become as effective as that of Functional Requirement. We finally propose a revised model of Problem Analysis and Requirement Specification. A step wise refinement model for quality concern into testable Non Functional Requirement is also proposed.

**Keywords:** Non-Functional Requirements (NFR); Testability; Requirement Analysis; Goal Refinement; Scenario Based Testing

## 1. Introduction

Although Non-Functional Requirements (NFR) have been present in all software, they have been treated as a second or even third class, type of requirements, frequently hidden inside notes and therefore often neglected or forgotten [1]. Several work [1-14], have indicated the fact that, despite Non-Functional Requirements (NFR), being, among the most expensive and difficult to deal with, even today there are only a few works that focus on NFR as first class requirements. According to Kazmen [15], "It is short sightedness on the part of Development process that Functionality has been given the first seat and Quality concerns have been given the second seat." It is universally accepted fact that NFRs play very dominant role in acceptability of software. Still, it has been treated very off handedly by industry for long, until they realized the fact, that, NFR cannot be neglected any more, because "NFR-not-satisfied" results into low acceptability, which goes against the product because of increasing competitive market, expectations of stakeholders, and failures of various critical systems [16].

There have been few observations on pivotal role of Requirement Engineering on testing a system in literature. Testing expert Dorothy Graham says that "we can save a great deal of money if testers are involved in testing Requirements" [17]. To add t o this crisis of poor testing of NFR, is the fact that [18] "Testing of NFR is in Embryonic stage."

In an effort to give first class treatment to NFR and to focus on agenda of "Requirement Engineering for NFR Testing", we explored the literature for finding, the work done towards testability of NFR. Pratima & Tripathi [14] have identified various issues, challenges and problems in testing of software with NFR. These identified issues are: Identification and classification of NFR, Specification for the testability of NFR *i.e.* Ambiguous specification in software requirement specification (SRS), Handling the Diversity of NFR, Handling Interplays among NFR, Cost and Effort of Testing contributed by NFR, Integrating FR with NFR *i.e.* Problems of scope of FR and NFR, NFRs role in decision making at design level or/ and specify, design and code for Testability of NFR.

If we analyze these issues, challenges and problems at Macro Level, we find that there are some issues which are inherent issues of Requirement Engineering and some are inherent issues of testing. But all facts are explicitly hinting at the fact that, if we are able to concentrate on quality of SRS we have half the game won for system (FR and NFR) testability of software.

As per IEEE format, an SRS should be [15,19,20] 1) Correct 2) Unambiguous 3) Complete 4) Consistent 5) Ranked for importance and/or testability 6) Verifiable 7)

Copyright © 2012 SciRes. 

Modifiable 8) Traceable Non Functional Requirement should specify two attributes [21]:

It should be objective.

It must be testable.

These characteristics of SRS regarding NFR, hold more importance and relevance since they (NFR) arise out of quality concerns of stakeholders, that may inherently be ambiguous and vague. Testing of NFR begins from obtaining measurable and testable requirements. This fact can be well supplemented from the performance testing model [20,22] which has to have a measurable and testable requirement as its first step towards testing.

"Functionality and Quality attributes are orthogonal" [15]. If they were not orthogonal, the choice of function would dictate, the level of usability, performance, availability and security. Clearly thought, it is possible to independently choose a desired level of each. Kazmen [15] has rightly said "had, functionality were the only requirement, the system could exist as single monolithic module with no internal structure at all". Since quality and functionality are absolutely orthogonal, architectural and design decisions play a significant role in bringing down quality concerns of a stake holder, actually implementable into the system. It is though architectural and design decisions that quality concerns are actually brought into the system [23]. Thus exploration of Architectural decision to embrace all the NFR, are a daunting task for Designers and Architects of a system. It should necessarily be explored for the purpose of testability of NFR.

Often quality and NFR as terms have been used interchangeably [12]. This is a misconception, which only reflects, casual treatment of NFR. No doubt Quality concerns are the source of NFR, but they are not NFR themselves. Quality concerns need to be explained for obtaining NFR.

Heuristics of measurement and testability, hold a great promise for "testing of NFR". As rightly said by Fenton [20,24]: "you cannot control what you cannot measure [9]". What cannot be measured cannot be managed. Now if we intend to manage testability, we need to measure it. [20,22,25,26]. Similarly IEEE definition of testability [27] explains that anything to be testable should be observable and controllable to make any things testable first it has to be observable then only it is controllable. Thus measurability and testability of NFR are the key points of NFR as well as FR.

Various works [6-8] indicate that there has been revolutionary interest among Software Engineering community to focus on Non Functional Requirement's: specification, analysis, its role in architectural decision making and finally its testing. Various issue, challenges and problems in testing NFR have been highlighted in [14].

With the intension to give first class treatment to NFR, and to enhance its testability, we move on to exploration of following facts:

1) Have quality concerns been objectively identified and mapped to NFRs? How to bring objectivity and refinement in the quality concerns? How to specify NFR, in SRS so as to make it, more testable?

2) What are the efforts made by Requirement Engineers and Architects to imbibe NFR in Software for its testability?

3) How can NFR be treated seriously and be given first class treatment, as well as in case of FR.

The following sections in the paper attempts to Explore and Evaluate NFR treatment for its testability in following dimensions (directions):

Section 2: Explores and evaluates the Limitations of quality concerns crystallization efforts for testability of NFR.

Section 3: Explores and evaluates Scenario Based Requirement Engineering.

Section 4: Propose a model for requirement Process in an attempt to give first class treatment to NFR.

Section 5: A Propose a Scenario based Specification technique for the evaluation of testability.

Section 5b: Derives the relation between Quality factors, Testability and testing effort.

Section 6: Conclusion.

## 2. Exploration and Evaluation of Goal Oriented Requirement Engineering (GORE) for Testability of NFR:

Analyze various efforts to crystallize/refine quality concerns for converting them into NFR statements. Certain significant attempts in this direction have been concerned with identification of quality concerns for the purpose of understanding what else apart from functionality may be needed to be understood by designers. These factors along with some techno-economic and other constrains have been used for estimating cost and working out a useful system, however many of these do not go further for refining quality concerns into NFR as in the case of FR.

In the first attempt of its kind for refinement of quality concern by McCall in 1977, FCM model was proposed [28,29]. This was a pioneer effort for refining the quality concerns of stakeholders which tries to identify quality concerns based on Use, Factor, and Criteria (related to productivity). It has its disadvantage of not being able to be crystallized till absolute atomic level where it is testable. On similar lines, Boehm has identified Quality in terms of characteristics that a software must exhibit [28]. Boehm's view of quality model is refined based on, primary uses, intermediate constructs, primary constructs and metrics. Both (Bohem, Mc Call) have given a clue on quality concerns based on, decomposition of key attributes called quality factors which are high level external

Copyright © 2012 SciRes. 

attributes, decomposed into lower-level attributes called quality criteria. On similar lines Somerville [21] has proposed, a more general classification which distinguishes between product, process and external requirements. Basili's Goal Question Metric [30] refines the quality concerns into NFR from conceptual, operational to quantitative level. This forms the basis of Chung's "NFR Framework" One of the concrete effort to identify NFR in literature of NFR [3,7,8,11,31]. Based on NFR Framework, various CASE tools have been proposed such as "NFR Assistant [8,11,31]", Star UML based on Chung's NFR framework [32]. i*, KAOS, AOURN, AOGRL. RML, GCT [1-6,11,14,33-35] are similar attempts of NFR refinement with vertical and horizontal extension of NFR framework as compared and contrasted in [14] **Figure 1** below shows the example of Goal based refinement of quality concern into a quantifiable *i.e*. measurable and testable NFR [3,7,8,13,31].

"SIG" ("Soft goal Interdependence Graph") for problem refinement and its related "NFR Framework" has its own share of limitations [2,3,7]. As the no. of Soft goal of different Quality concerns increases and the correlation among nodes increase, the evaluation procedure of the complete graph behind the Frame work becomes too complex [32]. Various domain based criteria are not the part of evaluation criteria behind the framework, whose inclusion may add another useful dimension of correlation among nodes.

**Figure 2** Emphasis the fact that while decomposing quality concerns into NFR, correlation among several quality concern complicates the refinement process of NFR. This fact affects the overall testability of NFR.

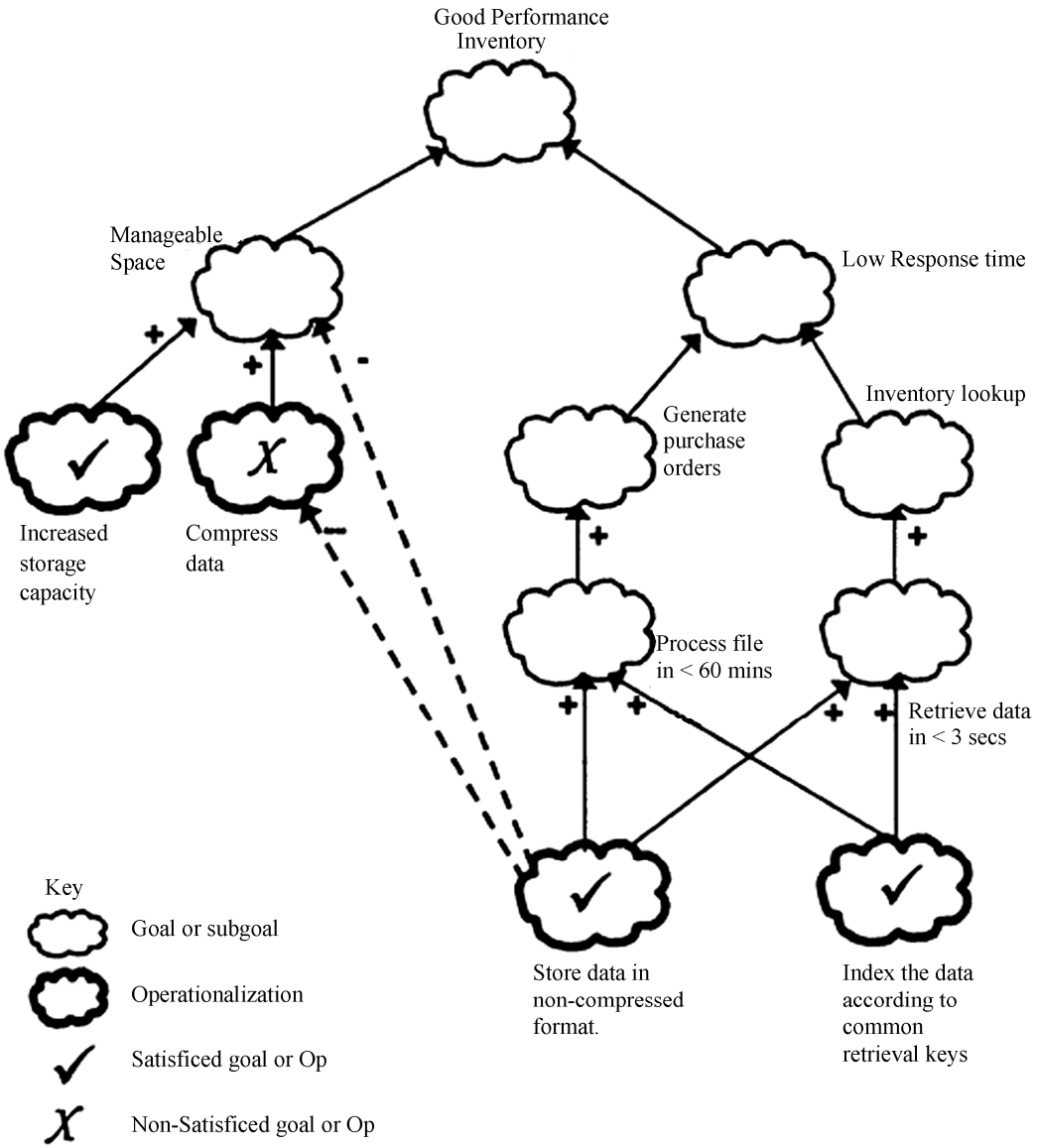


**Figure 1. Adapted from [13]: A soft goal interdependency graph.**

Copyright © 2012 SciRes. 

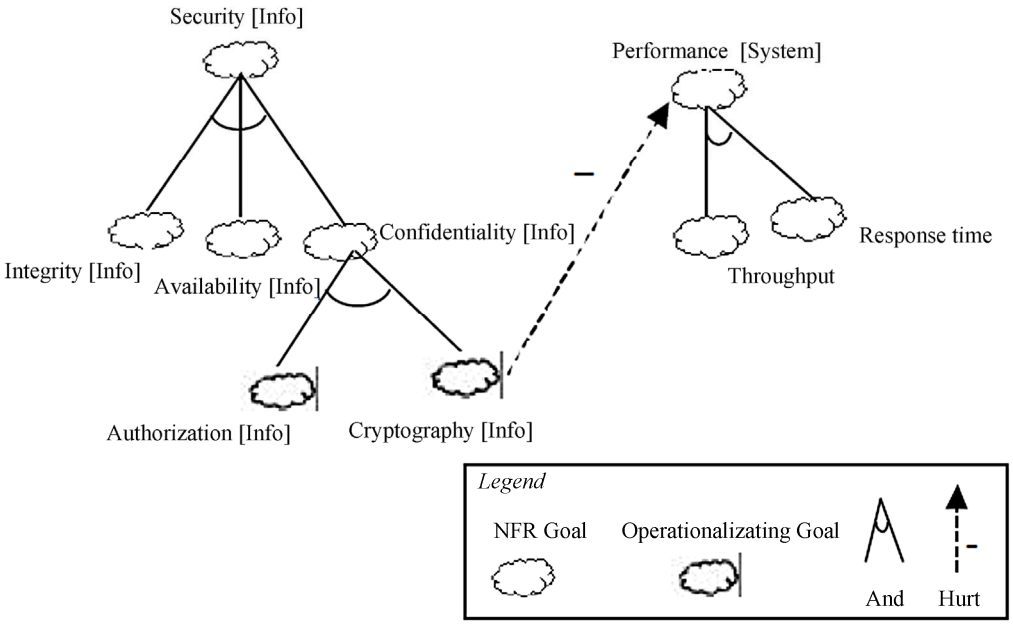


**Figure 2. Adapted from [3]: decomposition of NFR using NFR framework.**

## 3. Exploration and Evaluation of Scenario Based Requirement Engineering for Testability of NFR

Usage of Scenario based Requirement Engineering is advocated for the purpose of analysis because it promotes early discussion over requirement refinement and validation, resulting into conformance document to supplement metrics for quality assement [23,36]. A Scenario is a hypothetical story, used to help a person think through a complex problem or system. This work highlights the characteristics of good Scenarios, which says [36], A Scenario test has Following five characteristics. Scenario is 1) a story that is 2) motivating 3) credible 4) complex and 5) easy to evaluate. It is this unstructured features of Scenario based specification that makes it, an ideal candidate to deal with, NFR testing.

A. Gregoriades, A. G. Sutcliffe [37] have learned from MC Call's and Bohem's QOC (Question, Option, Criteria) notation of quality model for the refinement of NFR for the purpose of generating Scenarios based template. In this work finally a tool is proposed based on Bayesian Network. This tool named "System Requirements Analyzer (SRA)" is used to validate system. In SRA scenarios are transformed into sequences of task steps and the reliability of human agents performing tasks with computerized technology is assessed using Bayesian Belief Network (BN) models. Their attempts may be sufficient for requirement analysis but still far from generating a testable NFR.

While discussing Scenario based testing for NFR we cannot afford to miss the contribution of Architectural Community. Testing community can learn a lot from Architectural Community [38,39] for finding a predictor (stimulator or inhibitor)for testing of NFR. Kazmen [23] has very beautifully presented a template for generating a quality attribute Scenario and used it for taking design decision. ATAM (architectural trade off analysis method): an ADD (Attribute driven design), in which key scenarios are identified, evaluated for tradeoff and decision making to make specific design decision. Similar learning may be picked up for finding predictors of testability of NFR from ATAM. The format or the template for the quality attribute Scenarios is demonstrated in **Figure 3**. It consist of six parts: A quality attribute Scenario is a quality-attribute-specific requirement [23].

l) Source of stimulus. This is some entity (a human, a computer system, or any other actor) that generated the stimulus.

2) Stimulus. The stimulus is a condition that needs to be considered when it arrives at a system.

3) Environment. The stimulus occurs within certain conditions. The system may be in an overload condition or may be running when the stimulus occurs, or some other condition may be true.

4) Artifact. Some artifact is stimulated. This may be the whole system or some pieces of it.

5) Response. The response is the activity undertaken after the arrival of the stimulus.

6) Response measure. When the response occurs, it should be measurable in some fashion so that the requirement can be tested.

They distinguish general quality attribute Scenarios (general Scenarios)—those that are system independent and can, potentially, pertain to any system—from concrete quality attribute Scenarios (concrete Scenarios)—those

Copyright © 2012 SciRes. 

that are specific.

Scen Scenario Based Testing seems to hold great promise for testability of NFR [36,40]. As can be seen from **Figure 3** that Scenario based evaluation is more structured way of refinement of quality concerns than normal used case diagram because of semi structured form *i.e.* soft form of both NFR and Scenario in its identification of boundaries. The boundary to define an NFR is very thin and unstructured. There is need to structure (or map) the quality concerns of stakeholders into more testable Non Function Requirement statements. Scenario based analysis improves the testability of NFR because scenario is as abstract in its conceptualization as an NFR in its identification.

From the learning gained from different Software Engineering community Scenario is an effective means of capturing and evaluating users concerns in early stage of development [37]. We advocate the usage of Scenario based requirement analysis due to its strength of mapping the quality concerns into measurable, observable and controllable units through a template in the following sections. Thus the testability of NFR, can be enhanced by structuring the NFR, by mapping it into a testable templates, which refines an NFR for its testability.

## 4. Proposed Model of Requirement Analysis

Based on the learning from above literature from usage of Scenario for Non Functional Requirement analysis, specification, validation for the purpose of Requirement Engineering community and architectural design decision maker. we propose following modal of Requirement analysis process. This effort is motivated by the fact that

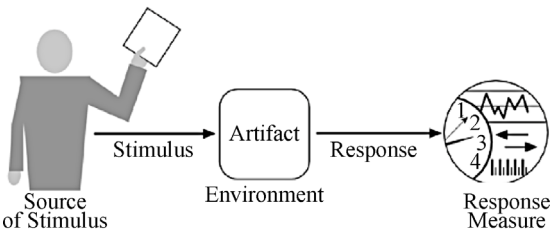


**Figure 3. Adapted from [23]: Scenario based testing components.**

NFR are being given second hand treatment which needs to be introspected into, by each community ranging from Requirement Engineering, design and Testing Community. As concluded from [14], the most important issue related to testing of NFR is the "second grade treatment for NFR", or non serious treatment of NFR. In an effort to normalize this biasness we compare and contrast the requirement analysis method of Functional Requirement and Non-Functional Requirements and finally propose a model, which tries to treat quality concerns as seriously as Functional Requirement. **Table 1** below discusses the difference between Problem analysis done for Functional Requirements to Non Functional Requirement. The difference in their approaches highlights the fact that where FR is concrete in its conceptualization that it focuses on data and object, NFR Analysis through GQM focuses on relationship of why, how and why not. Thus all together different mindset is required during analysis of the two different types of requirements.

The above model in **Figure 4** is motivated by the fact that what cannot be measured cannot be managed [24]. So if NFR specification is measurable, manageable, it is testable too (observable, controllable). The above approach of difference of treatment of FR and NFR for analysis can be minimized by identifying a relationship between External and Internal attributes. Thus we bridge the gap of treatment of FR with respect to NFR specification for its testability using following Scenario based template for Testability. It is note worthy at this point that not all quality concerns are refine able to achieve a measurable metric, there may be few quality concerns like usability or security concerns which enhances the testability of few quality concerns even by its unambiguous specification (without being reduced to metric for its measurement).

## 5. Proposed Scenario Based Refinement Template for Testability of NFR

Our proposed method has to follow two steps for refinement of NFR till testable point:

Step 1: Identify the vertical and horizontal (cross cutting) delimiters of a quality concern.

**Table 1. FR analysis vs NFR analysis.**

| Difference between FR Analysis and NFR Analysis | |
|---|---|
| Problem analysis results in DFD, Data Dictionary and Object Diagram [41,42]. | Problem Analysis of NFR by GQM resulting into testable Metrics. |
| Focus is on data and object [42]. | Focus is on Relationship of why, how and why not. |
| Key work area is Identification of Class/Object, Statement, Attributes, Associations and defined Services [41]. | Key work area is Identification of Purpose, Issue, Object (Process) and Viewpoints. |
| Requirement Specification results into Use Cases [41]. | Requirement Specification results into Scenarios. |
| Testable SRS can be made on basis of Internal Attributes. | Testable SRS for NFR can be made based on mapping external attribute to Internal Attributes. |

Copyright © 2012 SciRes. 

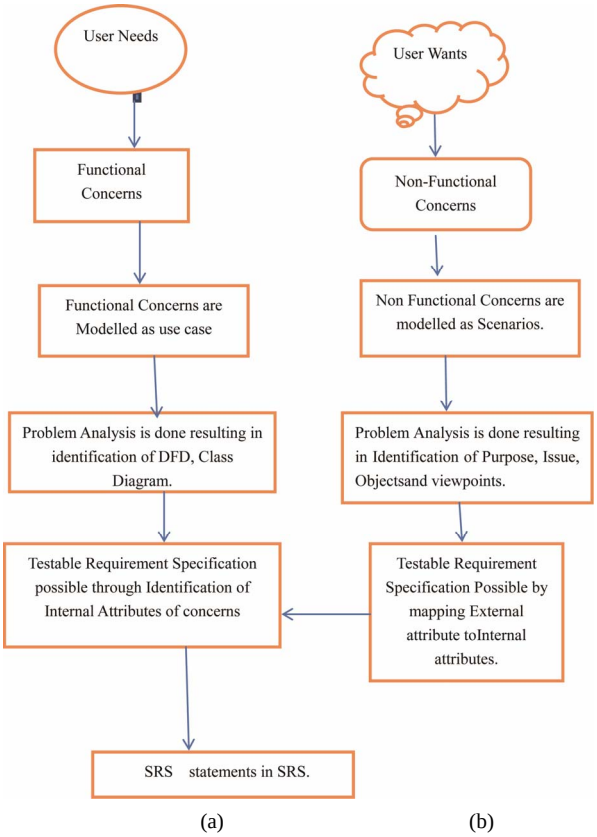


**Figure 4. Requirement analysis of FR (a) vs NFR (b).**

Step 2: Provide a Pyramid cal template for refinement of NFR till it is measurable or at least unambiguously identified, after identification of delimiters. We observe in the following example that few quality concerns like performance and reliability may be refined till metric where an allowable range may be set. But few like security or usability concerns may not be refined to the limit where an allowable value can be set. Still template enforces the identification of such quality concerns till unambiguous, observable, controllable and testable limit.

The Scenario based requirement analysis model discussed below, attempts to enhance the testability of NFR by refining the abstract (lofty) quality requirements of stakeholders into measurable values where ever possible so that it is testable too. Achieving measurable values for all quality concerns may not be possible, but refinement till achievement of unambiguous specification enhances the testability of NFR. It may be observable from above model that cross cutting concerns which spread across several modules also plays a significant role in deciding the predictors of, efforts for testability of NFR. Literature of Quality Concern has Focused a lot on Vertical *i.e.* Top-Down, refinement of Quality concern into NFR. This can be perceived as vertical dissection of the problem of refinement. Crosscutting concern view associated with every Quality concerns, may be called as Horizontal dissection if the problem of refinement. The spread of the crosscutting concerns across can give some predictors/indicators of testability of requirement. No doubt the indictors which we get are good means to achieve refined values from the abstract quality requirements. It can be seen as a rich source of external attributes of software which if efficiently mapped to Internal attribute can bridge the gap between FR and NFR treatment as discussed in above **Figure 4** in above section.

We are depicting the refinement stages through a pyramidical representation where in the top signifies the raw quality concern and as we move towards the bottom through the refinement steps we obtain the refined NFR. We have proposed our concept with three quality concerns only. They are Performance, Security and Reliability. Similar template can be applied for other concerns too: The Importance of Pyramid cal representation model is need to be appreciate the step wise refinement process of an abstract quality concerns into single point Requirement Statement, present in the lowest sections of the pyramid, The Requirement engineers are expected to come together the idea of quality concern into a single point statement. The field in the base of the pyramid is expected to be specifies in terms of measurable values. Its true that all quality concern may not be able to be decomposable till exact metric unit but if it can be speci-

Copyright © 2012 SciRes. 

fied till utmost unambiguous level, it contributes towards the testability of NFR.

**Table 2** below identifies the delimiters and cross cutting concerns related to Performance of a system.

**Figure 5** below shows that a quality concern such as Performance can be decomposed, step wise towards easily observable metric. Initial delimiters are very abstract or raw resulting into basic time and space characteristics of a performance. At middle level we collect related delimiters through crosscutting concerns. After identifying the cross cutting concerns in terms of caching, state mgt, and session management. Which affects the entire applications diagonally, we move down the pyramid getting refined delimiters. They have been achieved after vertical and horizontal flow of requirements of system. The template enforces the analysis to give the range of acceptable values to delimiters of performance refinement pyramid.

**Table 3** below identifies the delimiters and cross cutting concerns related to Security of a system.

Security Refinement Pyramid in **Figure 6** below also refines the security concerns but since all quality concerns are not decomposable till measurable unit, in the given template, macro view of delimiters do not result into a unit which crystal clear delimiters whose values can be set within a range. But if refinement is done till these points, NFRs testability can be increased.

**Table 4** below identifies the delimiters and cross cutting concerns related to Reliability of a system.

In the above **Figure 7** the top of the pyramid shows quality concern which is refined down the pyramid cal structure for the purpose of deciding the allowed values of each delimiters.

## Deriving Relation among Factors of Quality, Testability and its Testing Efforts

In the **Table 5** below we attempt to identify various horizontal and vertical factors which either directly or inversely related to Testability and Testing effort of a system. The table lists the correlation rules which relate impact of increase in no. of factors on testability and testing effort of NFR.

This **Table 5** may predict the testability of NFR based on these identified factors.

**Table 2. Performance delimiters.**

| Quality Concern | Performance |
|---|---|
| Macro view of Delimiters | Time, space |
| Micro view of Delimiters | Latency, throughput, Processing time, Jitter, miss rate, data loss. |
| Cross Cutting concern | Caching, State management, Session Management. |

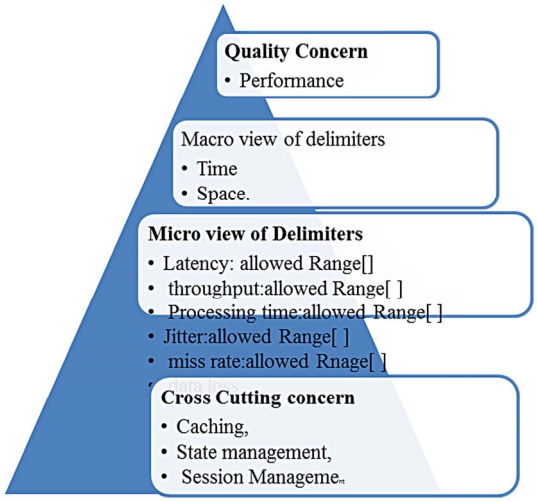


**Figure 5. Performancr pyramid.**

**Table 3. Security delimiters.**

| Quality Concern | Security |
|---|---|
| Macro view of Delimiters | Confidentiality, Integrity, Availability. |
| Micro view of Delimiters | Identifying trust boundaries for spoofing user. Repudiation of users action, difficulty of protection against various attacks like SQL injection, cross site, scripting, DOS attack, data tempering. |
| Cross Cutting concern | Authentication, authorization, Configuration Mgt, exception management, auditing, session management, Logging, encryption. |

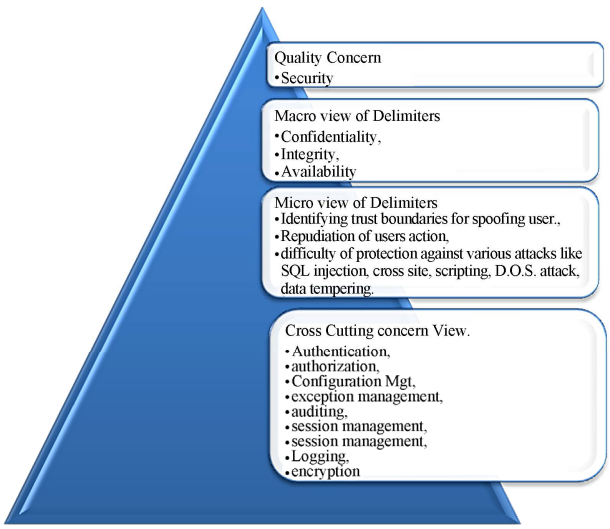


**Figure 6. Security refinement pyramid.**

## 6. Future Research Directions:

1) This Problem Analysis model can be refined for Domain Based Analysis, where refinements of Goal can be done better by putting Domain based constrains on the refinement process.

Copyright © 2012 SciRes. 

**Table 4. Reliability delimiters.**

| Quality Concern | Reliability |
|---|---|
| Macro view of Delimiters | Availability Robustness. |
| Micro view of Delimiters | MTTF (mean time to failure), MTTR, Downtime Probability, Fault Rate, Time to Recover, |
| Cross Cutting concern | Authentication, Authorization, Exception mgt, validation |

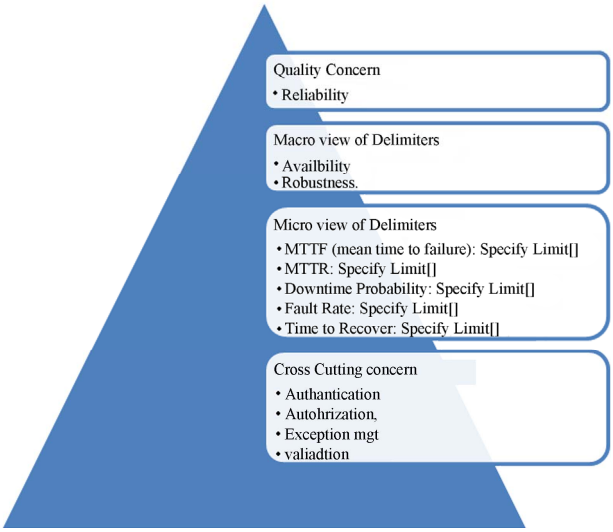


**Figure 7. Reliability refinement pyramid.**

2) Quality concerns are noting but stakeholders concerns, observations has to be done, as to how these concerns can be a part of the software and how these aspects can be tested through Test Aspects.

3) The same concept Scenario based refinement can be modeled/mapped on any Deterministic mathematical model, for early predictability of Testability of software with NFR.

## 7. Conclusions

There are several discussion on Requirement engineering playing dominant role testing and testability of Software [43,44]. Even Performance engineering community believes the key to testable.

Requirement is specification only [45] According to Heuristics of testability [46] and James Bach "What you see is what can be tested". All such attempts which makes the respective artifacts visible makes it testable. Scenario Based representation is apt for NFR analysis, due to its inherent benefit of encouraging rigorous exploration and validation at the early stage of NFR. Scenarios are very closed to real life situation, so are best suited for analysis of such an abstract, unstructured entity like NFR. NFR testing is different from FR testing as discussed in [14]. There are large no. of combination of situation possible, so to handle such large set of combinations, decision tree is ideal for its analysis. In the Future research direction we come up with tool support for

**Table 5. Relation among factors of quality, testability.**

| Correlation Rules | Factor | Consequences | |
|---|---|---|---|
| 1: | Increase in number of quality concern (QC) in an application | **Results in:** 1) Decreased Testability of application | **Testability $\alpha$ 1/No. of QC** |
| | | 2) Increased Testing Effort | **Testing effort. $\alpha$ No. of QC** |
| 2: | Increase in number of crosscutting concern (CC) | **Results in:** 1) Decreased Testability | **Testability $\alpha$ 1/No. Of CC** |
| | | 2) Increase Testing Effort | **Testing Effort $\alpha$ No. of CC** |
| 3: | Increase in number of Macro view Delimiters (**MaVD**) | **Results in:** 1) Increased Testability | **Testability $\alpha$ No. of MaVD** |
| | | 2) Increase Testing Effort | **Testing Effort $\alpha$ No. of MaVD** |
| 4: | Increase in Number of micro view Delimiters (**MiVD**) | **Results in:** 1) Increased Testability | **Testability $\alpha$ No. of MiVD** |
| | | 2) Increase Testing Effort | **Testing Effort $\alpha$ No. of MiVD** |
| 5: | Increase in correlations among micro view of different quality concern. (**COR**) | **Results in:** 1) Decreased Testability | **Testability. $\alpha$ 1/ No. of COR** |
| | | 2) Increase Testing Effort | **Testing effort $\alpha$ No. of COR** |

Copyright © 2012 SciRes. 

analyzing testability of software based on such analysis of the following mentioned correlation rule. We found that Scenario Based representation is most appropriate for NFR analysis, due to its inherent benefit of encouraging rigorous exploration and validation at the early stage of NFR. Scenarios are very closed to real life situation, so are best suited for analysis of such an abstract, unstructured entity like NFR. We finally proposed a template based on vertical and horizontal dissection of analysis of the quality concern. The delimiters identified at that stage becomes a good basis for identifying the testability and testing effort of the system.

## REFERENCES


[1] L. M. Cysneiros, J. C. S. P. Leite and J. S. M. Neto, "A Framework for Integrating Non-Functional Requirements into Conceptual Models," *Requirements Engineering Journal*, Vol. 6, No. 2, 2001, pp. 97-115. doi:10.1007/s007660170008

[2] L. Chung and J. C. S. do Prado, T. Borgida, *et al*., "On Non-Functional Requirements in Software Engineering," Mylopoulos Festschrift, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 363-379.

[3] S. Ullah, M. Iqbal and A. M. Khan, "A Survey on Issues in Non-Functional Requirements Elicitation," 2011 *International Conference on Computer Networks and Information Technology* (*ICCNIT*), Islamabad, 11-13 July 2011, pp. 333-340.

[4] P. Singh and A. Tripathi, "Exploring Problems and Solutions in Estimating Testing Effort for Non-Functional Requirement," *International Journal of Computers* & *Technology*, 3 October 2012.

[5] A. Matoussi and R. Laleau, "A Survey of Non-Functional Requirements in Software Development Process," October 2008 Laboratory of Algorithmics, Complexity and Logic (LACL) University, Paris, 2008.

[6] L. M. Cysneiros and J. C. S. Leite, "Using UML to Reflect Non-Functional Requirements," *Cascon*, 2001, p. 2.

[7] L. Chung, B. Nixon, E. Yu and J. Mylopoulos, "Non-Functional Requirements in Software Engineering," Boston Kluwer Academic Publishers, Boston, 2000.

[8] L. Chung and B. Nixon, "Dealing with Non-Functional Requirements: Three Experimental Studies of a Process-Oriented Approach," *Proceedings of the* 17*th International Conference on Software Engineering*, Seattle, Washington, April 1995, pp. 24-28.

[9] J. Mylopoulos, M. Pistore and P. Traverso, "Model Checking Early Requirements Specifications in Tropos," *Proceedings of the* 5*th IEEE International Symposium on Requirements Engineering*, *IEEE Computer Society*, 2001, p. 174.

[10] M. Glinz, "On Non-Functional Requirements," *The* 15*th IEEE International*, 15-19 October, 2007, pp. 21-26.

[11] A. K. Bharadwaj and T. R. Gopalakrishnan-Nair, "Mapping General System Characteristics to Non-Functional Requirements," *IEEE International Advance Computing Conference*, Patiala, 6-7 March 2009, pp. 1634-1638.

[12] S. Bode and M. Riebisch, "Tracing the Implementation of Non-Functional Requirements," IGI Global, Copying or Distributing in Print or Electronic Forms without Written Permission of IGI Global is Prohibited, 2011.

[13] L. Chung and N. Subramanian, "Process-Oriented Metrics for Software Architecture Adaptability, Requirements Engineering, 2001," *Proceedings of the* 5*th IEEE International Symposium*, Toronto, 27-31 August 2001, pp. 310-311.

[14] P. Singh and A. K. Tripathi, "Issues in Testing of Software with NFR," *International Journal of Software Engineering* & *Applications*, Vol. 3, No. 4, 2012, p. 61.

[15] L. Bass and P. Clements, "Software Architecture in Practice," 2nd Edition, Pearson, London, 2003.

[16] J. Lee and N.-L. Xue, "Analyzing User Requirements by Use Cases: A Goal-Driven Approach," *National Central University*, *IEEE Software*, Vol. 16, No. 4, 1999, pp. 92-101.

[17] C. U. Smith and L. G. Williams, "Software Performance Engineering for Object-Oriented Systems: A Use Case Approach," 1998, in Press.

[18] T. Romania, "Software Testing—State of the Art and Current Research Challenges," *The* 5*th International Symposium on Applied Computational Intelligence and Informatics*, 28-29 May 2009.

[19] R. S. Pressman, "Software Engineering: A Practitioner's Approach," 6th Edition, McGraw-Hill Publication, New York, 2005.

[20] S. Desikan and G. Ramesh, "Software Testing: Principles and Practices," Pearson, London, 2006.

[21] I. Sommerville, "Software Engineering," 2nd Edition, 2012.

[22] R. Black, "Foundations of Software Testing, Cengage Learning,"

[23] P. Clements, L. Bass, R. Kazman and G. Abowd, "Predicting Software Quality by Architecture-Level Evaluation," *The* 5*th International Conference on Software Quality*, Austin, October 1995.

[24] L. M. Cysneiros and J. C. Sampaio do Prado Leite, "Non-functional Requirements: From Elicitation to Conceptual Models," *IEEE Transactions on Software Engineering*, Vol. 30, No. 5, 2004.

[25] G. J. Myaers, "The Art of Software Testing," 2nd Edition, John & Willey Inc., 2004.

[26] N. E. Fenton and S. L. Pfleeger, "Software Metrics: A Rigorous and Practical Approach," 2nd Edition, International Thomson Computer Press, 1997.

[27] IEEE Standard Glossary of Software Engineering Terminology, 1990.

[28] http://www.bth.se/com/besq.nsf/(WebFiles)/CF1C3230DB425EDCC125706900317C44/$FILE/chapter_1.pdfMc

[29] M. Guessi, L. B. R. Oliveira and E. Y. Nakagawa, "Extensions of UML to Model Aspect-Oriented Software Systems, *Clei Electronic Journal*, Vol. 14, No. 1, 2011.

[30] L. Layman, V. R. Basili, M. V. Zelkowitz and K. L. Fisher, "A Case Study of Measuring Process Risk for Early


Copyright © 2012 SciRes. 

Insights into Software Safety," *The* 33*rd International Conference on Software Engineering*, Waikiki, Hawaii, May 2011, pp. 623-632.

[31] http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf

[32] S. Supakkul and L. Chung, "Integrating FRs and NFRs: A Use Case and Goal Driven Approach," *Proceedings of* 2*nd International Conference on Software Engineering Research & Applications*, Los Angeles, 5-7 May 2004, pp. 30-37.

[33] E. Kavakli and P. Loucopoulos, "Goal Modelling in Requirements Engineering: Analysis and Critique of Current Methods," *Information Modeling Methods and Methodologies*, pp. 102-124. doi:10.4018/978-1-59140-375-3.ch006

[34] S. Greenspan J. Mylopoulos and A. Borgida, "On Formal Requirements Modeling Languages: RML Revisited," 16*th International Conference on Software Engineering* (*ICSE*-94), Sorrento, 16-21 May 1994.

[35] U. R. N. Tropos, J. Castro, M. Kolp and J. Mylopoulos, "Towards Requirements-Driven Information Systems Engineering: The Tropos Project," *Information Systems*, Vol. 27, No. 6, 2002, pp. 365-389. doi:10.1016/S0306-4379(02)00012-1

[36] J. D. Cem Kaner, "An Introduction to Scenario Testing Cem Kaner," Florida Tech, June 2003.

[37] A. Gregoriades, A. G. Sutcliffe and H. Karanikas, "Evaluation of the SRA Tool Using Data Mining Techniques," *Proceedings of the* 15*th International Conference on Advanced Information Systems Engineering, CAiSE* 2003, Klagenfurt, 16-18 June 2003.

[38] M. R. Barbacci, M. H. Klein, T. Longstaff and C. Weinstock, "Quality Attributes," Technical Report CMU/SEI-95-TR-021, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, 1995.

[39] X. Franch, P. Botella, X. Burgués and J. M. Ribó, "Putting Non-Functional Requirements into Software Architecture," *Proceedings of* 9*th Software Engineering and Knowledge Engineering Conference* (*SEKE*), Madrid, 18-20 June 1997.

[40] A. Gregoriades, J. E. Shin and A. G. Sutcliffe, "Human-Centred Requirements Engineering," 12*th IEEE Proceedings of the Requirements Engineering Conference*, Washington, 6-10 September 2004, pp. 154-163.

[41] R. Mall, "Fundamentals of Software Engineering," 3rd Edition, PHI Publication, New Delhi.

[42] G. Booch, "UML Modelling Book: Unified Modeling Language User Guide," Addison Wesley, Boston, 1998.

[43] B. Lawrence, K. Wiegers and C. Ebert "The Top Risks of Requirements Engineering," *IEEE Software*, Vol. 18, No. 6, 2001, pp. 62-63.

[44] D. Graham, "Requirements and Testing: Seven Missing-Link Myths," *IEEE Software*, Vol. 19, No. 5, 2002, pp. 15-17. doi:10.1109/MS.2002.1032845

[45] J. D. Cem Kaner, "The Ongoing Revolution in Software Testing," *Software Test & Performance Conference*, 8 December 2004.

[46] B. Pettichord, "Design for Testability," *Pacific Northwest Software Quality Conference*, Portland, 13-14 October 2002.

Copyright © 2012 SciRes. 