Journal of Engineering and Applied Sciences 13 (16): 6622-6625, 2018
ISSN: 1816-949X

Classifying Non-Functional Properties of Software Systems According to their use in Some Application Areas

Abdulkarim Bello, Abu Bakar Md Sultan, Abdul Azim Abdul Ghani and Hazura Zulzalil
Faculty of Computer Science and Information Technology,
University Putra Malaysia, Malaysia

Abstract: Dealing with non-functional properties has a great challenge to software engineers for many years. For many years, software engineers are using this properties provide high quality to the software products they developed. This make it possible for engineers to divide their focus of testing non-functional properties according to, the services the software product renders. Although, non-functional properties of software systems are being described as the most important contributors to the success of software systems, study to date revealed that there is no general consensus on the notion of certain classification of NFPs. This study presents results of a study on the investigation conducted on the different classification of Non-Functional Properties (NFP) of software systems.

Key words: Non-functional properties, non-functional requirements, non-functional requirements, NFPs, NPRs, Malaysia

INTRODUCTION

Software system's utility is determined both by its functional as well as non-functional properties (Chung and Leite, 2009). Dealing with non-functional properties of software systems poses a lot of challenges to software engineering communities. Over the years, different methods and techniques are being proposed to improve on their documentation and elicitation (Ameller and Franch, 2010). Search-base software testing being a sub-area of search based software engineering (Afzal *et al*., 2009; Harman *et al*., 2015) an area of research that attracted much attention in recent years as part of the general interest in search-based software engineering approaches (Harman, 2007). The growing number in non-functional properties consideration can be attributed correlation of growing number of automation in software development community and software testing, since, it is known that exhaustive testing is infeasible and the fact that software test data generation is consider NP-hard (McMinn, 2004). A lot of researchers have written a survey on search-based software testing including McMinn (2004) which shows the application of search-based techniques for white-box testing, black-box testing, grey-box testing and for the verification of non-functional properties (Afzal *et al*., 2008). Since, non-functional properties of software systems described the way in which the systems operate, rather than actions the system performs (Patrick and Jia, 2015) makes it so much important to optimize.

Though you may find two software systems that might be functionally the same but may differ in the way they perform their actions and also, differ in their execution time, response time, memory usage and/or power consumption. Therefore, there is need to have software systems with its non-functional properties thoroughly optimized. The needs to develop high quality software that finds solution amongst all the available solutions is a cause of concerned to software systems developed considering the role of non-functional properties, being these properties that measure the quality software systems, also, derives a lot of challenges to software engineers due to nature posed by these properties. The aim of this study is to categorically classify all the non-functional properties of software system that are identified by this study and also mention the properties that have no definition or have definition but have no attributes (Mairiza *et al*., 2010) as is reported by various literatures.

Literature review: The success of a particular software system or the safety of the people using the system are dependent upon underlying quality concerns in many software systems (Mirakhorli and Clelanf-Hung, 2012). This section outlined some of the related work of software

Corresponding Author: Abdulkarim Bello, Faculty of Computer Science and Information Technology,
University Putra Malaysia, Malaysia

engineering researchers that focused on the identification and utilization of non-functional properties of software systems. Work such as that by Harman *et al.* (2015) undertook a review in the area of search-based software testing, its origin, trends in publications and open problem. The review uses data from Search-Based Software Engineering repository, indicates how the area continues to grow with respect to the increase in publication, how well the area of non-functional properties is getting the attention of researchers with some cause of concern to some areas. The concluded by introducing a future tool that automatically finds bugs, fix the bugs and verifies what it fixes. Ameller and Franch (2010) conducted an empirical study to answer the question on how software architects consider non-functional properties/requirements of a software system. The study uses software architects from twelve different companies to answer some interview questions through a semi-structured interviews. The study identified and grouped non-functional properties for the study into: most important and non-technical NFPs. The most important NFPs consisted of thirteen non-functional properties which include performance, usability, security, availability, interoperability, maintenance, accuracy, fault tolerance, reusability, scalability, modularity and portability. While the non-technical NFPs are licensing issues, technological policy, cost, external regulations, availability of support and organizational policy. After (Chung and Leite, 2009) finds out that the attention of software engineering researchers concentrate mostly towards the identification and optimization of functional characteristics of software systems conducted a study that explores the non-functional properties of software systems. The study also finds a plethora of definitions to non-functional properties in software engineering and grouped the identified non-functional properties based on the article presented by Paech and Kerkow (2004). Roohullah *et al.* conducted a survey that emphasizes the benefits on integration of non-functional properties of software system at architectural level of system development. The survey identified some techniques through which non-functional properties can be integrated into system design phase of software development. The considered non-functional properties for their research are performance, maintainability, security and fault tolerance. There is also a study by Mylopoulos *et al.* (1992) that uses process-oriented approach to represent non-functional properties of software system. The study considers design architecture as the basis for evaluating how well a particular property suit certain architectural design in which thirteen non-functional properties were identified and grouped into three phases of acquisition, performance, design and adaptation.

Table 1: Sources of information considered for the study

Database	Address
IEEE xplore	http://www.ieeexplore.com/
Springer link	https://link.springer.com/
Science direct	http://www.sciencedirect.com/
ACM digital library	http://dl.acm.org/
Google scholar	https://scholar.google.com/

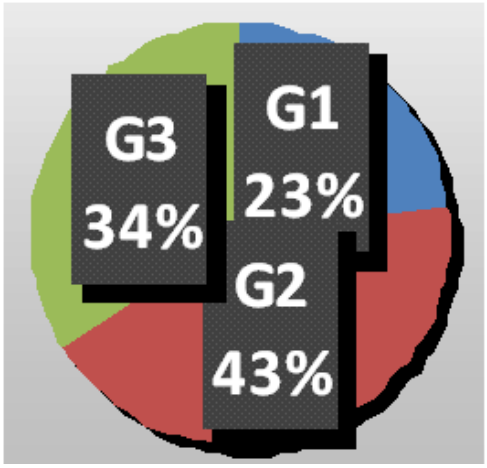

Fig. 1: Usage based categorization of non-functional properties

MATERIALS AND METHODS

The study was conducted from different sources of information published in academic resources within the area of software engineering such as journal article, conference proceedings and ISO/ISEC standards and some few industrial reports. All these selected articles cover various issues that are concerning the non-functional properties in software engineering as can be seen in Fig. 1 as I reported by Chung and Leite (2009). Table 1 shows the details of the databases and composition of articles considered form each database for the study. All the articles were then analyzed systematically using content analysis technique (Cho and Lee, 2014). The method of content analysis was chosen because it enables the researcher to systematically identifying its properties through locating the more important structures of its contents. Such information are then categorized to provide meaningful reading contents which are then extracted for further usage.

RESULTS AND DISCUSSION

We used the information extracted from the selected primary studies to answer the research question. This section used the results obtained by the study to answer the research questions defined at the beginning of the study.

Table 2: ISO/ISEC25010-2011 standard classification of non-functional properties

NFPs	Sub categories of NFPs					
Functional Stability	Functional Appropriateness	Functional Completeness	Functional Correctness			
Performance Efficiency	Capacity	Time Behaviou	Resource rutilization			
Compatibility	Coexistence	interoperability				
Usability	Appropriateness Recognisability	Learnability	Operability	User error protection	User interface aesthetics	Accessibility
Reliability	Maturity	Availability	Fault Tolerance	Recoverability		
Security	Confidentiality	Integrity	Non repudiation	Authenticity	Accountability	
Maintainability	Modularity	Reusability	Analysability	Modifiability	Testability	
Portability	Adaptability	Installability	Replaceability			

Table 3: Categories of non-functional properties as classifies by Mairiza *et al.* (2010)

Has definition and attributes	Has definition	Without definition and attributes
Accessibility, adaptability, availability, efficiency, fault tolerance, functionality, integratability, integrity, maintainability, maintainability, modifiability, performance, portability, privacy, readability, reliability, reusability, robustness, safety, scalability, security, testability, understandability, usability,	Accuracy, analysability, Attractiveness, changeability, complexity, composability, confidentiality, consistency, correctness, defensibility, dependability, evolvability, extendibility, flexibility, immunity, installability, interoperability, learnability, likeability, localizability, maturity, operability, quality of service recoverability, replaceability, stability, suitability, survivability,	Accountability, additivity, Adjustability, affordability, agility, anonymity, atomicity, auditability, augmentability, certainty, compatibility, comprehensibility, conciseness, configurability, conformance, controllability, customizability, debuggability, decomposability, demonstrability, distributivity, durability, effectiveness, enhenceability, expandability, expressiveness, extensibility, feasibility, formality, generality, legibility, manageability, measurability, mobility, nomadicity, observability, predictability, provability, reconfigurability, repeatability, replicability, self-descriptiveness, simplicity, standardizability, strcturedeness, supportability, susceptibility, sustainability, tailorability, traceability, trainability, transferability, trustability, uniformity, variability, verifiability, versatility, viability, visibility, wrappability,

The number of non-functional properties used in software engineering: To answer the first question of the study (i.e., RQ1) this study itemized the different number of non-functional properties in software engineering as is presented in different contents of the primary studies encountered by this review.

ISO/ISEC25010-2011 standard quality characteristics: The software quality characteristics defined by ISO. (2011) are identified twenty eight non-functional properties grouped into eight different characteristics that includes: functional Suitability, performance Efficiency, Compatibility, Usability, Security, Maintainability and portability (ISO., 2011). Under each category subcategories are define. Table 2 outlined the non-functional properties as defined by ISO. (2011).

Definition/attributes-based classification: The definition/attributes-based classification of non-functional properties was proposed by Mairiza *et al.* (2010). In this classification, over hundred non-functional properties are grouped into three categories based on definition and attributes. The classification includes those with definition and attributes, those with definition and those without definition and attributes (Table 3 and 4).

Table 4: Application area based classification

Application area	Relevant NFPs
Banking and finance (Business)	Accuracy, conformity, reliability, confidentiality, performance, security, usability verifiability, privacy, verifiability
Education	Inter-operability, reliability, correctness performance, scalability, security, usability,
Energy resources	Availability, performance, reliability,safety,
Government and defence	Accuracy, confidentiality, performance privacy, reusability, verifiability, viability
Health care	Communicativeness, confidentiality, integrity, performance, privacy, reliability, safety, security, traceability, usability
Telecommunication	Compatibility, conformance, dependability, performance, portability, accessibility
Transportation	Accuracy, accessibility, availability, completeness, dependability, integrity, safety, security, verifiability

Frequency-based usage of non-functional properties: To answer research Question two (RQ2), NFPs are software systems are grouped in three as can be seen in Fig. 1 below. The study grouped the identified NFPs into three categories, G1-G3.

G1 (1997-2007): This group all non-functional properties that are identified from different work of SBST researchers that fall between the years of 1996 and 2007 and are used in optimizing software system (Afzal *et al.*, 2008, 2009).

G2 (1997-2015): This group contains non-functional properties that get were used by software engineering researchers within the period range of 1996-2015 for optimizing software system.

G3 (1997-2015): This group contains the non-functional properties that this study has not come across any study that used them to optimized software systems. This group falls into the categories of non-functional properties that without definition and attributes (Mairiza *et al*., 2010). Classification base on application domain.

CONCLUSION

Non-functional properties are the means through which software systems can be given qualitative feature and make it operate in it most optimized form. This study was able to enumerate the available non-functional properties of software systems and group them into different classifications, the ISO/ISEC (Mairiza *et al*., 2010) and according to, the usage intensity by software testers for the optimization of software systems. The focus of the study is to classify the available non-functional properties that are identified by software engineers from various work they published and software engineering standard body such as IEEE, ISO/ISEC, etc., it does not report the procedure the researchers used in coming up with the non-functional properties and/or the various method they used while optimizing software system through the mentioned non-functional properties. The study, also, does include not the metric measurements of non-functional properties and their fitness evaluation.

REFERENCES

Afzal, W., R. Torkar and R. Feldt, 2008. A systematic mapping study on non-functional search-based software testing. Proceedings of the 20th International Conference on Software Engineering and Knowledge Engineering, July 1-3, 2008, San Francisco, CA, USA., pp: 488-493.

Afzal, W., R. Torkar and R. Feldt, 2009. A systematic review of search-based testing for non-functional system properties. Inform. Software Technol., 51: 957-976.

Ameller, D. and X. Franch, 2010. How do software architects consider non-functional requirements: A survey. Proceedings of the 16th International Working Conference on Requirements Engineering: Foundation for Software Quality, June 30-July 2, 2010, Springer, Essen, Germany, ISBN:978-3-642-14191-1, pp: 276-277.

Cho, J.Y. and E.H. Lee, 2014. Reducing confusion about grounded theory and qualitative content analysis: Similarities and differences. Qual. Rep., 19: 1-20.

Chung, L. and J.C.S.D.P. Leite, 2009. On Non-Functional Requirements in Software Engineering. In: Conceptual Modeling: Foundations and Applications, Borgida, A.T., V.K. Chaudhri, P.Giorgini and E.S. Yu (Eds.). Springer, Berlin, Germany, ISBN:978-3-642-02462-7, pp: 363-379.

Harman, M., 2007. The current state and future of search based software engineering. Proceedings of the 2007 International Conference on Future of Software Engineering, May 23-25, 2007, IEEE, Washington, DC, USA., ISBN:0-7695-2829-5, pp: 342-357.

Harman, M., Y. Jia and Y. Zhang, 2015. Achievements, open problems and challenges for search based software testing. Proceedings of the 2015 IEEE 8th International Conference on Software Testing, Verification and Validation (ICST), April 13-17, 2015, IEEE, Graz, Austria, ISBN:978-1-4799-7125-1, pp: 1-12.

ISO., 2011. IEC25010: 2011 Systems and software engineering-systems and software quality requirements and evaluation (SQuaRE)-system and software quality models. International Organization for Standardization, Geneva, Switzerland.

Mairiza, D., D. Zowghi and N. Nurmuliani, 2010. An investigation into the notion of non-functional requirements. Proceedings of the 2010 ACM International Symposium on Applied Computing, March 22-26, 2010, ACM, New York, USA., ISBN:978-1-60558-639-7, pp: 311-317.

McMinn, P., 2004. Search-based software test data generation: A survey. Software Test. Verification Reliab., 14: 105-156.

Mirakhorli, M. and J. Cleland-Huang, 2012. Tracing Non-Functional Requirements. In: Software and Systems Traceability, Cleland-Huang, J., O. Gotel and A. Zisman (Eds.). Springer, London, UK., ISBN:978-1-4471-2238-8, pp: 299-320.

Mylopoulos, J., L. Chung and B. Nixon, 1992. Representing and using nonfunctional requirements: A process-oriented approach. IEEE Trans. Software Eng., 18: 483-497.

Paech, B. and D. Kerkow, 2004. Non-functional requirements engineering-quality is essential. Proceedings of the 10th Anniversary International Workshop on Requirments Engineering Foundation for Software Quality, June 7-8, 2004, Fraunhofer IESE, Kaiserslautern, Germany, pp: 237-250.

Patrick, M. and Y. Jia, 2015. Exploring the landscape of non-functional program properties using spatial analysis. Proceedings of the 7th International Symposium on Search Based Software Engineering (SSBSE'15), September 5-7, 2015, Springer, Bergamo, Italy, ISBN:978-3-319-22182-3, pp: 332-338.