IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 3, Ver. IV (May – Jun. 2015), PP 53-57
www.iosrjournals.org

A Review on Quality Assurance of Component- Based Software System

Parminder Kaur[1], Navdeep Batolar[2]
[1]*Professor, Computer Science Department, Guru Nanak Dev University, Amritsar, India*
[2] *Student, Computer Science Department, Guru Nanak Dev University, Amritsar, India*

Abstract: *Among the various challenges to industry, the major challenge is to offer products with high level of quality andfunctionality at inexpensive value and short time and energy to market. Component-based software development(CBSD) approach have become quite popular from the point of view of quality assurance. The most captivating reason for embracing CBSD approach is the conjecture of reuse. The goal of this study is to comprehend categorize and inspect prevailing research in CBSD field from quality point of view. The main focus is given onproposals that are accustomed to assess the quality of component- based software system (CBSS). This paper represents the methodology through which the quality of component-based software system can be assured. Quality assurance is taken as a vital research term. In this paper, two main approaches for assuring quality are taken under consideration: encapsulation and composition rules. Functional and non functional properties are encapsulated as an individual unit to be able to fulfill the product quality demands. In this paper, focus is given on two main quality attributes: predictability and reusability. This paper analyze the prior work being prepared for quality assurance and compare the work on the foundation of their research methodology. No matter what but it is very difficult to develop a software system by fulfilling all the quality demands and because of this reason a more determined attempt is required to meet an improved assessment approach in the future.*

Keywords: *Component-based software systems, quality assurance, encapsulation, composition rules*

I. Introduction

Software quality is defined as conformance to standards and requirements. Quality assurance should go hand in hand while developing the software system because without an appropriate level of quality , the component usage will provide ruinous results. A CBSS is a system in which component is used as a basic unit. A software component can be considered as a unit of composition with specified interfaces and context dependencies. [4] The main aim of component-based development approach is to develop the software system by using already existing components and hence there is no need to start from the scratch.

The main benefits of CBSS is reusability as software system is developed by using already existing components, reduced time to market and hence reduced software production cost.[10] The heart of component-based development (CBD) technology is its software component model which defines the various standards for component implementation ,naming, customization, interoperability, deployment and composition [4]. The main use of component model is for composition of components. Composition defines the following key points:

- Specifies the particular components used to develop the system.
- What type of interfaces these components have.
- What are the pattern of their execution.

The reason for doing this review is to present the advantages of using component- based approach for improving the quality of software system. CBD is an important emerging topic which can provide concepts which provides us with more sophisticated methods as compared to traditional techniques. This paper surveys the previous work being done for assuring the quality of CBSS and it also reveals the strength and weakness of these approaches for assuring quality. The two main quality attributes that are taken into consideration are predictability and reusability.

This paper is organized as follows. In section II , the previous work being done for assuring the quality of CBSS is explained briefly. In section III the research method for assuring the quality is described and various research questions are identified and the observations are explained . In section IV conclusions are derived.

II. Related Work

There are many research works being done for enhancing the quality of CBSS and quality assurance is considered as a key research content in our review work. The literature review of various papers is being explained below in much detail.

Seceleanu et al.(2013) [15] has discussed the method by which the system design can be simplified. Component- based design technique is used for developing software. It is a technique in which a software system is developed by using commercial off the shelf (COTS) components and hence increases the predictability and reuse of the software system.. According to this paper a software design can be simplified if we encapsulate both functional and extra functional attributes (quality assurance). This approach has several challenges like this method is quite complex and in many case not achievable. Abdellatief et al.(2013) [1] has classified and studied existing metrics for component- based software system (CBSS) and identified the elements for evaluating the quality of CBSS. Two metrics discussed are component consumer and component producer. According to this paper the true benefits of component- based software system can be achieved if the components are evaluated by using appropriate metrics.

Nikolik et al.(June 2012) [14] has discussed about the economic metrics. The main aspects taken into consideration for assuring quality are artifact defect and artifact test cases. The concept of test cases is used which is a procedure applied to artifact on order to obtain actual results. The cost and the value of test cases are considered as important economic variables of quality assurance economics. Defects are handled by using three techniques prevention, defect removal, defect avoidance. The test case cost value is used to calculate ROI which is a performance measure used for evaluating efficiency. Case study is performed in this paper in order to determine the change in test case with respect to changes in artifact. Based on the case study some guidelines are introduced for maximizing the test case value and ROI and minimizing the test case cost. In order to get more practical knowledge about the metrics experimentation is needed on large industrial projects. In order to make defect avoidance possible more experimentation is needed. Experimentation on economic release criteria is also needed. Li et al.(2012) [11] pointed out that the true benefit of dynamic configuration can only be achieved if it causes minimum disruption to the ongoing application. The main issue on which this paper concentrates are the properties of configuration framework which include dynamic version control (DVM), reconfiguration timing control, stateless equivalent and controllability overhead. Various methods are used to preserve the features of quality of service (QoS) assurance. In order to control the overheads the reconfiguration mechanism is divided into three phases: installation, transformation, removal. The measure challenge for implementing the concepts discussed above is that these concepts can only be applied to local process, hence component state migration is the biggest challenge. Esposito et al.(June 2011) [6] has given an appropriate method for assessing the quality of each component in a CBSS and selecting that component which fits better according to the system requirements. A framework is formed which can achieve the above mentioned objectives. A customized quality model is used which describes the quality attributes and will properly evaluate off the shelf products. A personalized and stratified definition of quality model is being provided. The approach used in this paper is very flexible so that changes can be easily made according to current state of art and the approach is mainly used for critical software. Crnkovic et al.(2010) [4] focused on component models and gives a brief description about currently available software component models. This paper defines the component models as set of standards which are used for implementing, customizing, composing and deploying of components. The research methodology used in this paper is based on an empirical approach which follows three main steps: observations and analysis, classification, validation. This paper has given a brief description about basic characteristics and principles used for component models which includes lifecycle, construction and extra functional properties.. The main aim of this paper is to enhance the understanding of component- based approach.

Lau et al.(2007) [10] gives a brief description about software components models. The three main quality attributes described in this paper which can be achieved by using component- based approach are reusability, productivity and reduced time to market. In this paper currently available component models are analyzed and classified into a taxonomy suitable for component based development .This paper has given description about 13 software component model which include JavaBeans, EJB,COM, .NET, CCM, Web services, Koala, Kobra, Acme like ADLs, UML2.0 , PECOS, SOFA and fractal. After the selection of appropriate component model their syntax, semantics and composition is explained. The taxonomy explains the characteristics of the existing component models. An ideal model is still under survey that would allow composition on both deployment and design phase together with the use of a repository. The ideal model should have the key characteristics of encapsulation and compositionality. Kenett et. al(2007) [8] explores the quality concepts. This paper describes an extended quality conceptual framework which represents an extension of software quality framework. Two fundamental concepts discussed in this paper are assuring the quality and testing the product. The main aim is to place quality in proper prospective in relation to acquisition and development of computer software. Various activities are performed to assure quality which includes establishing requirements and controlling the changes, establishing method of implementation and achieving specified product quality and finally evaluating process and product quality. These three concepts are explored in detail. In order to characterize the extended quality framework a set of definitions and related concepts are first specified and explained in detail. The product quality is specified by using product attributes. The

definitions forms the basis for establishing quality requirements, methods to help satisfy these requirements, and quality evaluation. Liangli et al.(June 2006) [12] describes the method for improving the testability of CBSS and quality assurance is taken as a key research content. In this paper eight types of dependencies have been summarized between two components in a component- based system. Two types of graphs are used component dependency graph (CDG) and component direct dependency graph (CDDG) and then dependency relationship matrix and direct dependency relationship matrix are defined. Two types of approach code- based and specification- based approach are used to find the dependency. Matrix-based approach is used to test a component when it is integrated to a component-based system.

Alvaro et al.(2005) [2] discusses the advantages of using component based approach for software development. Reliability quality feature is described in much detail as how reliability is ensured by using a component based development approach. According to this paper as COTS components are reused on various occasions , hence they are likely to be more reliable as compared to the components developed from scratch. This paper discuses the issues emerged while implementing component based approach and also provide appropriate directions for resolving these issues. Three main questions which this paper raises are what to evaluate, how to evaluate and who will evaluate. Various quality models are discussed which are based on component technology and software quality experience of the researchers. Apart from reliability other quality attributes like functionality, usability, efficiency, portability, maintainability are also discussed. However the quality models discussed are not evaluated into academic or industrial scenario due to which the real efficiency to evaluate software components using these models remain unknown. Lau et al.(2005) [9] represents the taxonomy of current component models. The purpose of representing this taxonomy is to find out the similarities and differences between the currently available models according to commonly accepted criteria. A reference framework for software component model is represented. The syntax, semantics and composition of the of the component models are explained in detail. Composition is explained in two phases design phase and development phase. Component models are classified according to various categories based on component syntax which include JavaBeans, EJB(the components are implemented in java) COM, CCM (these use IDLs to define generic interfaces) Kobra, UML 2.0 (components are explained by using architectural design languages). This taxonomy clearly reveals the strength and weakness of currently available component models. As a future work we need to find a model that supports predictable assembly which forms the cornerstone for component-based development approach. Muccini et al.(June 2005) [13] represents the dependability level in a component-based software system. Two main issues taken into consideration while determining the dependability are quality assurance of reusable software components and quality assurance of the assembled component-based system. This paper mainly evaluates software architecture-based regression testing methods that warranties reusability. This paper mainly concentrates on the factors by which the testing efforts can be reduced and quality can be increased. As a future work a more diligent method will be used to meet the above objectives. Xia et al.(2001) [16] discuss about component- based software engineering(CBSE) approach. The main area of interest of this paper is to study as how quality assurance can be made possible by using CBSE. A risk analyzer tool ARMOR is also studied. The main focus is given on system architecture which is a layered and modular architecture. The three main component technologies discussed in this paper are CORBA, COM and DCOM, Sun's javabeans and enterprise javabeans. Quality assurance technologies are also studied which include reliability analysis model and component- based approach to software engineering. The quality assurance model should address to both process of component and process of overall system. The ARMOR tool measures and test the quality and risk for software programs. As future work the ARMOR tool is made to evaluate and analyze the quality and risk of components and component- based software system. Dias et al.(June 2000) [5] has pointed out that the component- based software development depends mainly upon the quality of the components as well as their layout. This paper represents an approach to analyze the architecture and component-based development on the basis of statechart semantics. In order to assure the quality , behavior of the specified components is also taken into consideration. Both static and dynamic techniques are used for analysis process. A gap is found between the state –of-the- art and state-of-practice which is reduced by using two approaches : bringing art-to-practice and bringing practice-to-art. If an integrated set of capabilities are used for both architecture and component-based development then the quality of the software system can be enhanced.

III. Research Method

3.1 Research Questions

Various research questions have been tackled and recognition of these research questions is the first step for consistent literature review.

RQ1. What are the proposed definition of CBSS and why CBSS become quite popular?
Motivation: To understand about CBSS.
A CBSS is a system in which component is used as a basic unit. A software component can be considered as a unit of composition with specified interfaces and context dependencies.[4]. A CBSS is a system which is being developed by combining components that have been deployed independently.[1]. CBSS are becoming quite popular because of the following advantages. Firstly, the reusability feature as software system is developed by using already existing components. Secondly, reduced time to market and hence reduced software production cost [10]. Thirdly, it provides us with shorter software life cycles.

RQ2. Is quality assurance is taken as a key research content?
Motivation: To understand the importance of quality during software design.
Software quality is defined as conformance to standards and requirements. Quality assurance should go hand in hand while developing the software system because without an appropriate level of quality , the component usage will provide ruinous results.

RQ3. Can we assure quality in a CBSS ?
Motivation: Develop an appropriate method for developing a quality software.
The latest resonance in the hardware development encouraged the developers to create a software which is simple in its design but contains an appropriate level of quality .In order to develop a system which contains a well defined quality level, the developers must satisfy challenges that go afar from pure functionality. In order to meet the above objectives an appropriate methodology is to be developed developed which contains the following key features:

- Simplifying the software design by using functional and extra functional attributes.
- Study of metrics that is used for evaluating quality of Component- based software system.
- Validation of metrics by developing component-based software system with functional as well as extra functional attributes.

RQ4. How can we evaluate the available component-based software metrics?
Motivation: The main purpose of this question is to tackle the currently available component-based software metrics and identify their advantages and disadvantages.

The metrics are mainly used to identify the CBSS attributes. CBSS metrics have been viewed as a perspective of consumer and producer. The metrics can be evaluated by first gathering all the relevant information about currently available metrics and then comparing these metrics and identifying their pros and cons according to the gathered information.

Table-1: Answers evaluation criteria

The answers	Original scale of the answers
The answers can be correlated to the related work	Yes
The answers can be mostly surmised from related work	Mostly
The answers can be somewhat surmised from related work	Somewhat
The answers are undetectable or unknown	No

Table-2: Assessment of related work

Research paper	RQ1	RQ2	RQ3	RQ4
Seceleanu et al.(2013)	Somewhat	Mostly	Mostly	No
Abdellatief et al.(2013)	Yes	Somewhat	Somewhat	Mostly
Nikoli)k et al.(June 2012	No	Somewhat	Somewhat	Somewhat
Li et al.(2012)	Somewhat	Mostly	Somewhat	No
Carlson et al.(2012)	Somewhat	Yes	Mostly	No
George et al.(June 2012)	No	Yes	Somewhat	No
Esposito et al.(June 2011)	Yes	Yes	Mostly	No
Crnkovic et al.(2010)	Somewhat	No	Somewhat	No
Lau et al.(2007)	Somewhat	Somewhat	Mostly	No
Kenett et. al(2007)	No	Yes	Somewhat	No
Liangli et al.(June 2006)	Somewhat	Yes	Somewhat	No
Alvaro et al.(2005)	Yes	Somewhat	Mostly	No
Lau et al.(2005)	Somewhat	Somewhat	Somewhat	No
Muccini et al.(June 2005)	Mostly	No	Somewhat	No
Xie et al.(June 2004)	Somewhat	No	No	No
Xia et al.(2001	Yes	No	Somewhat	No
Dias et al.(June 2000)	Somewhat	No	Somewhat	No

3.2 Observations

Software quality is an important factor which simplifies a system design and increases the trust in the correct functioning of the software. Various analysis and prediction techniques are developed to form a sound software development methodology that will produce quality software. Component- based software engineering has become well known as it offers reduced development cost, theory of reuse and shorter life cycles also act as a motivational factor. All the above mentioned features has improved the product quality and make this approach very attractive. In order to simplify a system's design while maintaining the quality feature, the concept of component binding is being studied which is a mechanism to connect the component in such a way that one component's interface is connected to other's. There are two types of component binding[15]:

- **Horizontal binding:** It represents the connection of a component's provided interface with a subsequent component's required interface. This assembly does not necessarily constitute a new component; it is just an assembly of interacting components, and the resulting composition is called a horizontal composition.
- **Vertical Binding:** It is an assembly that constitutes a new composite component that complies with the model's interface; the new composite component can be connected to other components in the same way as any other component complying with its model.
  In order to simplify the design of software system, an appropriate method is to encapsulate both functional and extra functional properties. [15]
- Functional properties describe the relationship between component and system variables and constrain the values associated with system operations or state changes .
- Extra functional properties also known as nonfunctional properties mainly includes the quality attributes such as efficiency, effectiveness. These properties has higher trustworthiness, due to its ability to uncover potential trouble spots before actual system implementation.

However it remains a challenge to simplify system's design while maintaining the quality feature. For this reason much effort is needed to develop an evaluation approach that will enhance the quality of component-based software system.

IV. Conclusions

Today's software system is characterized by its quality. A component – based approach that maintains the quality to an appropriate level while simplifying the system's design is must. The work represented here introduces various techniques by which the quality of component-based software system can be improved. More sophisticated experimentation and evaluation methods are required so that quality assurance and simplification of system's design should go hand in hand and in future work we will try to perform such experiments.

References

[1]. Abdellatief, M., Sultan, A. B. M., Ghani, A. A. A., & Jabar, M. A. (2013). A mapping study to investigate component-based software system metrics.Journal of systems and software, 86(3), 587-603.

[2]. Alvaro, A., Almeida, E. S., & Meira, S. L. (2005). Quality attributes for a component quality model. 10th WCOP/19th ECCOP, Glasgow, Scotland.

[3]. Carlson, J. (2012, June). Timing analysis of component-based embedded systems. In Proceedings of the 15th ACM SIGSOFT symposium on Component Based Software Engineering (pp. 151-156). ACM.

[4]. Crnkovic, I., Sentilles, S., Vulgarakis, A., & Chaudron, M. R. (2011). A classification framework for software component models. Software Engineering, IEEE Transactions on, 37(5), 593-615.

[5]. Dias, M. S., & Vieira, M. E. (2000). Software architecture analysis based on statechart semantics. In Software Specification and Design, 2000. Tenth International Workshop on (pp. 133-137). IEEE.

[6]. Esposito, C., Cotroneo, D., Barbosa, R., & Silva, N. (2011, April). Qualification and Selection of Off-The-Shelf components for Safety Critical Systems: a Systematic Approach. In Dependable Computing Workshops (LADCW), 2011 Fifth Latin-American Symposium on (pp. 52-57). IEEE.

[7]. George, R.; Samuel, P. (2012, November) Improving design quality by automatic verification of activity diagram syntax. Intelligent Systems Design and Applications (ISDA), 2012 12th International Conference on , vol., no., pp.303,308.

[8]. Kenett, R. S., & Baker, E. (2010). Process Improvement and CMMI® for Systems and Software. CRC Press.

[9]. Lau, K. K., & Wang, Z. (2005, August). A taxonomy of software component models. In SoftwareEngineering and Advanced Applications, 2005. 31st EUROMICRO Conference on (pp. 88-95). IEEE.

[10]. Lau, K. K., & Wang, Z. (2007). Software component models. Software Engineering, IEEE Transactions on, 33(10), 709-724.

[11]. Li, W. (2012). Qos assurance for dynamic reconfiguration of component-based software systems. Software Engineering, IEEE Transactions on, 38(3), 658-676.

[12]. Liangli, M., Houxiang, W., & Yansheng, L. (2006, October). The Design of Dependency Relationships Matrix to improve the testability of Component-based Software. In Quality Software, 2006. QSIC 2006. Sixth International Conference on (pp. 93-98). IEEE.

[13]. Muccini, H., Dias, M., & Richardson, D. J. (2005, July). Reasoning about software architecture-based regression testing through a case study. InComputer Software and Applications Conference, 2005. COMPSAC 2005. 29th Annual International (Vol. 2, pp. 189-195). IEEE.

[14]. Nikolik, B. (2012). Software quality assurance economics. Information and Software Technology, 54(11), 1229-1238.

[15]. Seceleanu, C.; Crnkovic, I. (2013, November) Component Models for Reasoning. Computer , vol.46, no.11, pp.40,47.

[16]. Xia, C., & Fu, A. (2001) Component-Based Software Engineering: Technologies, Quality Assurance Schemes, and Risk Analysis Tools. In Seventh Asia-Pacific Software Engineering Conference.

[17]. Xie, G. (2004, September). Decompositional verification of component-based systems-a hybrid approach. In Automated Software Engineering, 2004.Proceedings. 19th International Conference on (pp. 414-417). IEEE.