International Journal of Management, IT & Engineering
Vol. 7 Issue 4, April 2017,
ISSN: 2249-0558 Impact Factor: 7.119
Journal Homepage: http://www.ijmra.us, Email: editorijmie@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gage as well as in Cabell's Directories of Publishing Opportunities, U.S.A

COMPONENTS BASED SOFTWARE TESTING STRATEGIES TO DEVELOP GOOD SOFTWARE PRODUCT

Ahmed Mateen[*]

Hina Zahid[**]

Abstract

Component based software development is used for building a new software artifact rapidly by using less resource. Different components are collected and integrated collectively to form new product therefore the feature of new software product depends upon these components. To ensure value of overall product testing of each component is vital. But problems arise during testing when tester has inadequate access to the components. Testing is the procedure of exercising a program with the specific intent of decision errors before to delivery to the end user. This paper discusses the value of components testing and comparison of these testing techniques. Some important techniques are discussed like Unit testing, System testing, Integration testing Module testing. Alpha testing beta testing and also proposed better testing technique to ensure that the proposed testing techniques are more efficient and beneficial a survey study conducted. Survey based on questionnaires and results calculated by applying statistics on data gathered from survey. At the end some suggestions to improve the process of testing are also given in this paper.

Keywords:Component testing Component based software engineering Testing strategies Software testing

[*]**Department of Computer Science, University of Agriculture Faisalabad**

1. Introduction

Component based software engineering is used to develop new software application by reusing already build components. It also improves the quality and reduces the maintenance cost of the product.

From the last few decades in the field of software engineering demands from the users end are increasing and they want their software products within time[1].

If such problem arises then it would be difficult to fix faults in the product after its deployment, this will increase the maintenance cost and even could cause product to face the failure. In component based testing the main issue arises due to limited access to the source code. Also less information is provided by component providers about design document of the reusable components. The main reason is that component providers do not want to expose complete information about component and do not provide awareness about the behavior of component in other environment. To overcome such problems testing team need to apply strict testing criteria so the components that are being reused are tested earlier for their required product environment. Lot of efforts has been done to solve the problems in component testing because testing has significant role for the development of any quality software product. As component based software is developed by integrating different components therefore the overall quality of the product depends upon the quality of components used. Also effective component testing process helps to select suitable components for new product[2].

Software professionals agree that software testing is as significant as software development.Development and testing teams work together to deliver a quality product with in right time. Testing team helps to reduce the ambiguities of the product by detecting bugs and by the interaction with the developer these bugs can be removed in initial stage of software development. This methodology helps to produce a bug free quality product within time for customer satisfaction and also reduce the overall cost of the product[3].

There are many testing strategies that are helpful in component based testing because these are useful to generate information about the components and help to evaluate the validity of the component for the new software product. These are some of the testing strategies that will be discussed in this paper.

A. UML based test model for Component
B. Built-in-tests in components (BIT)
C. Component Interaction Testing (CIT)

A) UML Based Test Model

This technique uses UML diagrams to find the problems that might reside in the component interfaces. It is used to specify and construct the artifacts of the software product. By using collaboration diagrams and sequence diagrams interaction among different objects in a component is represented. Then this model uses these diagrams to link the development process of UML to the test process to perform testing among the component as shown in figure [4].

B) Built-In Testing Components

Built-in contract testing use the technique of describing test functions in the source code as the member function of the product. This feature enhances the maintainability of the software product. After merging these functions the software product that is proposed can be used in two different modes. The one in which software has behavior as conventional system is the normal mode. The other in which the member functions of BITs are called is the maintenance mode in this mode these functions are activated and after their execution testing results are generated as shown in figure 2[5]. These are the steps that are involved in this strategy.

1) Built-in test in component are included in the software as the member function.
2) Normal mode is used to work the product behavior as conventional system.
3) Then maintenance mode is used to activate the BITs which result generating the test report after the execution of the testing function.

C) Component Interaction Testing

This technique involves the assumption about each component with the other component that tells how they react with each other. Test requirements and formal models of components are used to generate test cases for integration testing and can be reused when more components are integrated or when components are changed. The components that are being tested are not integrated in fix order that make this testing strategy more flexible. When it is possible to create assumption between how component react with each other with the help of requirements and set forward model[6].

The following steps are followed in this approach to test the components.

1) Create a showing how the components must be used.

2) Then test requirements are specified.

3) After this test cases are generated by combining the test requirement and set forward model.

4) Last step is of testing interaction between these components.

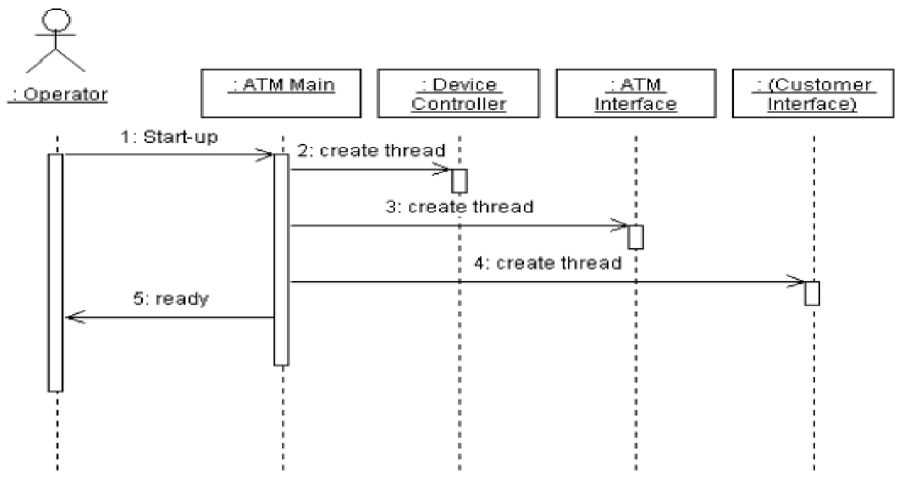


Figure 1. UML Test sequence diagram [4]

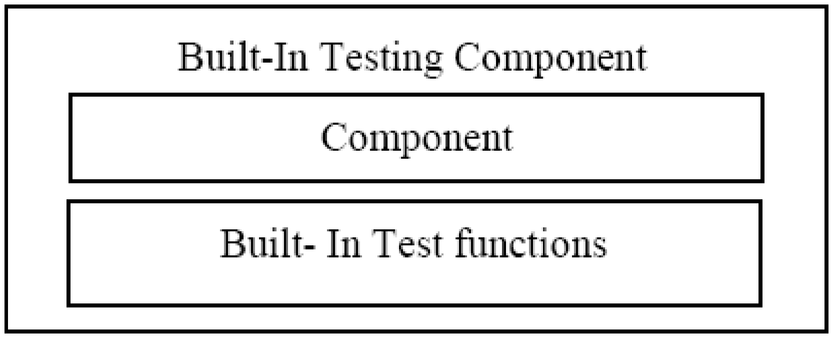


Figure: 2 Built-in testing[5]

Table1. Strategies and their methods of testing

[1] Strategy	[2] Testing Criteria
[3] UML based test model	[4] Use sequence and collaboration diagrams to extract faults.
[5] Built-in-tests in components	[6] Built-in-tests as member function in source code.
[7] Component Interaction Testing	[8] Capturing assumptions as formal test requirements.

2. Previous Work

The re-enables developers to effectively create a reuse component improves the overall quality of the software, reducing the cost of software, and provide software with fewer errors aspects of software design are described in some detail for both direct solution algorithms to produce a respectable inconsistency in software development[7]t. Conversion allows developers to create professional processing software engineering to improve the overall software superiority, reduce expenditures software, and download the software using a small amount of error. Process allows programmers to build a professional software system, such as presentation software artifact slightly from systems construction software such as scraping[8]. The demonstration software design and analysis, become an active area of research. However, the approaches and structural analysis software design informally. The term "architecture" can be accurately defined as a semantic compatibility between related companies can be properly investigated. software developers use different descriptions of the development process, including analysis, requirements, design, code, manuals, test plans, change requests, and asking style, our process diagrams and models. These descriptions have been built and maintained by software engineers often through their development.the development of capabilities for software development life cycle gambling reflection[9]. Using software project goal is to create software at a high level. Construction will use the software, which are liberation activities. Account application software exhibition activities and development processes of software using the software in the rules of how the administration. Effective representation, query, and special combinations of abstract touch in the database, because it is clear that the titular frame assembly called vague spatial algebra[10].

A) OBJECTIVE

The objective of this research is to proposed better testing technique by comparison of different techniques in components based software products and also discusses which technique is good and efficient for that SPSS software will be used to performed to prove result.So that the quality of products will be improved.

3. Research Methodology

A)Components based software engineering prosess

B) Phases of Testing Strategy

This strategy consists of different phases that will support testing on components involved in the product. These are the following phases of the testing strategy.

i) Interaction Representation

In this phase components are represented in pictorial form. Since components can interact with each other through their interfaces. A component send request to other component which provide service to request sending component.component A is connected to component B depending upon request and service. Component B sends request x to component A which provide service against that request. Same way service y is provided by component B. The service provided by component is executed by its corresponding function[11].

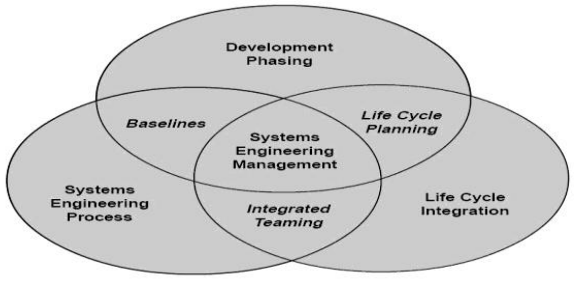


Figure 3. components based software engineering process[14]

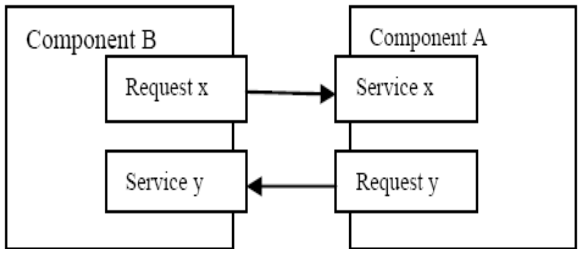
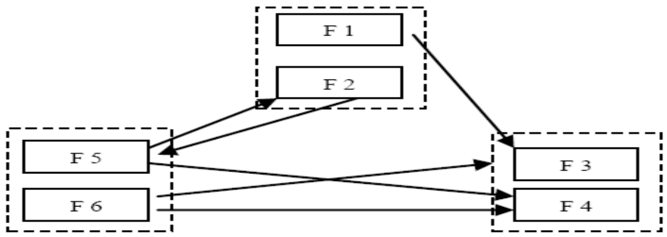


Figure 4. Interaction Representation of components [11]



$$\chi^2 = \sum \frac{(O - E)^2}{E}$$

χ^2 = the test statistic $\sum$ = the sum of

O = Observed frequencies E = Expected frequencies

Figure 5. Function Dependancy Graph [12] Equation of Chi square [13].

ii) Function Dependency Graph

After getting information about components and function specifications a function dependency graph is made that explains the function dependency. In Fig 5 function dependency graph is made to show the dependency of different functions of component on other functions [12].

B) Statistical Technique

A chi square statistic is a measurement of how expectations compare to results. The data used in calculating a chi square statistic must be random, raw, mutually exclusive, drawn from independent variables and drawn from a large enough sample [12].

C) Components based testing strategy – proposed strategy

Interface based component testing strategy has been proposed as a feasible method for testing a component based software. This strategy can lead to the production of a quality product as each component will be tested according to the requirements of the user. The main characteristics of this strategy are to give importance to testing during component software development and analyze dependency between components.

D) Suggestions to improve Component testing

Testing have very important role in any software development process. There is lot of work need to be done in component based testing area. As much improved testing method would make product more usable and there would be less chances of failure. These are some of the suggestions to improve the component testing.

i) Good testing automation should bring into practice because it can help to evaluate large number of test cases that would be more effective to detect faults. Although there are some tools but with limitations so there need improvement.

ii) There should be more information sharing between consumers and producers in case of third party component usage. This may cause lesser problems to test components.

iii) Limitations to create generic test suites should be removed because it would be more helpful for testing components with the similar behavior.

iv) Component change information should be provided to the consumer so that they have complete information if there are changes made in the components.

v) For testing detailed design document and information about these components and changes with respect to all of its previous versions should also be provided the components.

vi) Complete documentation including user manuals and interface specification should be provided by the component producers along with the components.

vii) Testing should be made on the reusable components under different scenarios so that reusable components have vast scope of usability under different conditions.

viii) Testing results should be stored so that in future these results can be used to perform testing faster and in efficient manner.

ix) Testing should be very strict to fulfill the requirements of the product. As components are combined requirements should be checked on each step of new integrated component.

4. Results

Majority of the respondants were UML based test model UML testing model is produced with the help of Automatic Message Function (generated by separating association and sequence diagrams) and Node (representing integration targets). So the UML based model is very valuable model in softwere testing and the respondants ratio were 79%.

In this perspectives responses was in four categories agree and disagree somewhat agree strongly agree.the maximum results were agree because risk was rsduced by software testing.2% were disagree thay may be felt that risk was not reduced by software testing.

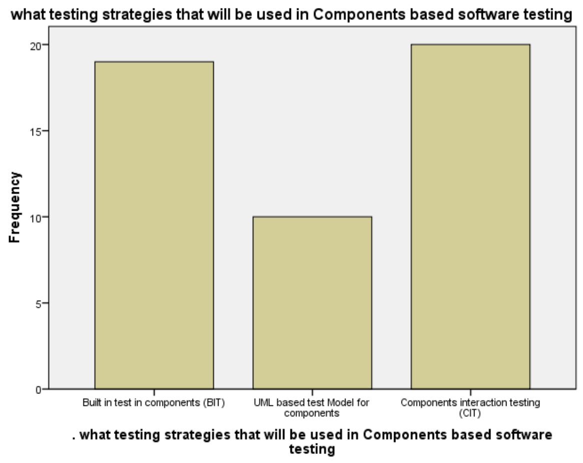


Figure 6. Testing strategies. This figure shows the results of testing stretegies 40% respondants were CIT stretegies.10% was BIT and 19% were UML based test model for components

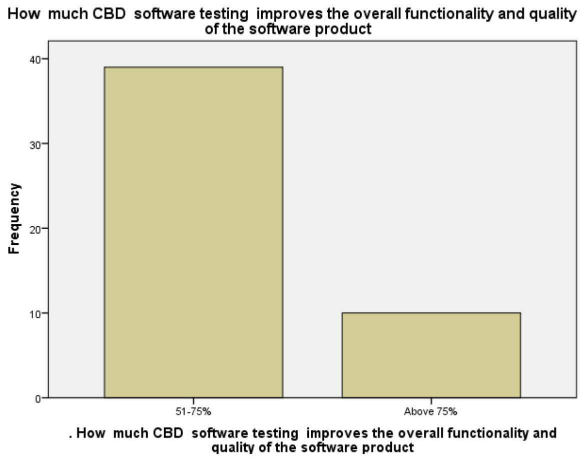


Figure 7. Improve software quality. In this perspectives79 responses was in between 51 - 75% and other 20 was above 75%

Table2. Testing strategies

		Frequency	Percent	Valid Percent	Cumulative Percent
	Built in test in components (BIT)	19	38.8	38.8	38.8
	UML based test Model for components	10	20.4	20.4	59.2
	Components interaction testing (CIT)	20	40.8	40.8	100.0
	Total	**49**	**100.0**	**100.0**	

Table-3 Improve software quality

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	51-75%	39	79.6	79.6	79.6
	Above 75%	10	20.4	20.4	100.0
	Total	**49**	**100.0**	**100.0**	

How much CB software testing improves the overall functionality and quality of the software product

Table4. Risk reduced by software testing

		Frequency	Percent	Valid Percent	Cumulative Percent
	Agree	29	59.2	59.2	59.2
	Disagree	2	4.1	4.1	63.3
	Somewhat Agree	10	20.4	20.4	83.7
	Strongly Agree	8	16.3	16.3	100.0
	Total	**49**	**100.0**	**100.0**	

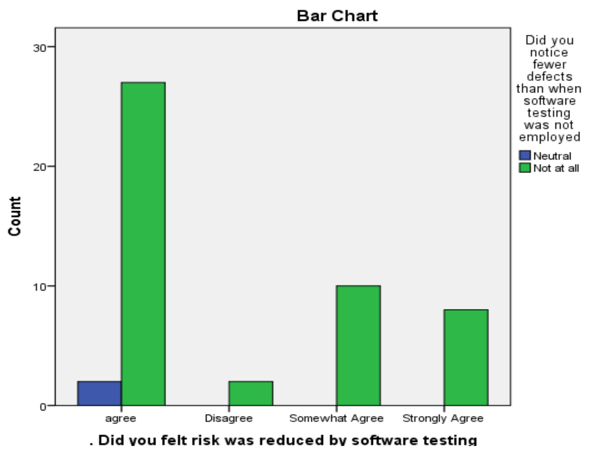


Figure 8. Risk reduced by software testing

5. Conclusions

In the component based software development different already build components are used. To ensure the quality of such product component testing is very essential. But problems may arise for the tester in component testing phase due to the limited access of the component. In this paper component based testing strategies are discussed and using a good strategy according to situation quality of the product can be improved. If best suitable practices are applied on new component based software, it become more reliable and chances of failure become very less. Testing should be performed on every component and whole software product before delivering it.

6. References

[1] Conrad, J., Dengler, P. Francis, B. Glynn, J. Harvey, B. Holllis, B. Ramachandran, R. J. Schenken, S. Short and C. Ullman. *Introducing .NET, Wrox Press.USA*, 1(1):506-512,2012.

[2] Dogru, A. H. and Tanik. M. M.“A process model for component-oriented software engineering”. *IEEE Transations, USA*. 20(2): 34-41, 2011.

[3] Henry, E., Faller. B.“Large Scale Industrial Reuse to Reduce Cost and Cycle Time”. *IEEE Software,USA*. 12(5): 47-53.2013.

[4] Rizwan J. “Reuse and Component Based Development”. *Internationl Conference. Software Engineering Research & Practice, Las Vegas, USA*, 1(1): 146-150,2006.

[5] Jerry, J and Jingsha. “Testing Coverage Analysis for Software Component Validation. 29th Annual International Computer Software and Applications Conference *IEEE*.13(5): 47-53,2012.

[6] Shalom, C. and Haan. U.“A Software System Development life cycle model for improved stakeholders communication and collaboration”. *International journal of computers, communications and control*, 5(1): 20-41,2010.

[7] Shireesha, P. and Sharma. S.S.V.N. “Building Reusable Software Component”.*International Journal of Software Engineering & Applications*, 1 (3): 38-46, 2010.

[8] Shaker, M. and Ibrahim. H. “Information Extraction from Hypertext Mark-Up Language web Pages”. *Journal of Computer Science*, 5(8): 596-607,2009.

[9] Wallnau, K. C and Stafford. J. “Abstractions for A New Class of Design Problem”, *In Proceedings of the 27th Euromicro Conference, 2001. D Edition, McGraw Hill, 2008., Journal of Business Research*, 59(10):1072-1078,2008.

[10] Postmus, D. and Meijler. T. D. “Aligning the economic modeling of software reuse with reuse practices”. *Journal ofInformation and Software Technology*. 50(7): 753-762,2008.

[11] Gilchrist, I. “Software Quality Engineering: Testing, Quality Assurance, and Quantifiable Improvement”. *Journal of Software Testing, Verification and Reliability*. 16(2): 124-128, 2006.

[12] Delgado, A. A., Troyano J. A. E. and Estipa. R. “Testing UI elements with Model-Based User Interface Development”. *Journal of Human Computer Studies*, 86(1):48-62,2015.

[13] Do, Q., S. Cook and M. Lay. “An Investigation of MBSE Practices across the Contractual Boundary”. *Journal of Procedia Computer Science*, 28(1): 697-701, 2014.

[14] Gao, J and Ming. C.,“A component testability model for verification and measurement” 29th Annual International Computer Software and Applications Conference, IEEE,2012.