International Journal of New Media Studies **ISSN: 2394-4331**
Volume 4 Issue 2, Julyy-December, 2017

Built-in Testing for Component-Based Software Development

Hitali Shah

Institute of Technology, Nirma University

ABSTRACT

When Component based software development process is employed, there is a need to test the prefabricated components after integrating them with the existing software system. The components available are already tested but the interfaces connecting them are not. This paper focuses on how different test cases and test interfaces can be embedded within components so that when they are integrated they can be easily tested. The benefits for using built-in testing for component based software development process is discussed. Built-in contract testing is studied further and its various terminologies are mentioned.

Keywords—Built-in Testing, testing, COTS, Component-based software development

INTRODUCTION

Component Based software development is used in majority when it comes to excel the software development process. The implicit assumption when using COTS is that time required to deploy and test the software system would less time consuming compared to the traditional software development process. There may occur a condition when the component may not work as expected. In such cases a lot of time may be consumed in acceptance and integration testing which is not preferable. Moreover, the COTS are tested on different servers and when they are deployed on client server there can be run-time errors. While integrating COTS with the existing software system could also generate defects. Testing these components fabricated by third-party companies may take a lot of time and energy.

Built-in testing is method of testing in which code is added within the component that is used at run-time to test that component. The developer adds test cases and test interfaces while developing the component. They basically work on the approach of “plug-and-play” such that when the COTS are deployed the built-in tests could validate the environment and work as they are expected to. Built-in testing thereby would reduce the time and cost required for integration and regression testing.

This paper discusses the reasons for using built-in testing in component-based software development. Following sections would discuss the various methods of using built-in testing in integration and regression testing. Moreover, the use of built-in testing in maintenance of software system is also discussed.

Challenges In Component-Based Software Testing

Different testing methods are used for component based software testing. Some of the approaches are mentioned in [3] . One of the approach is using testable beans which is mentioned in [1].Testable beans use Enterprise JavaBeans framework. These testable beans can be extended to provide support for Built-in Testing. When deploying built-in testing within prefabricated components there are various research questions that arise. Here some of the questions are addressed.

Why to use Built-in testing?

When using component based software development the major challenge is to perform integration and regression testing thereby followed by acceptance testing and maintenance after the software is being deployed. The prefabricated components may not behave the same way on the client’s server as expected to due to various different parameter changes. Testing such components would increase testing time thereby increasing the usage of resources and cost of the software development.

So in these cases it would be preferable to use built-in testing where the component developers provide developed test suites within the component and when they are plugged into the system the testing is automated. This is one of the main reason why built-in testing is employed in component based software development system.

How can Built-in Testing facilitate verifiability?

Verification and Validation requires lot of human intervention in standard testing procedures. When Component based software system is used to ensure the benefits of plug-and-play there is a need not only to perform integration and regression testing but also verification. Built-in Testing provides a way through which the prefabricated components could automate their testing by using already defined test cases and test interfaces within the component. To provide verifiability such that component can provide reactive notifications in case of defects, an extension to built-in testing has been proposed in [2] termed as MORABIT. 3) How can Built-in Testing facilitate maintainability?

To facilitate maintainability mechanisms built-in testing operates in basically two modes as normal mode and maintenance mode as proposed in [6].

In normal mode it works same as normal calling member functions procedure as

```
Class    class_name {
    //class   interface
    Declarations
    Constructer  declaration Destructor
        declaration
     Test    cases     //built −in    tests
    //implementations
    Constructor Destructor
     Test   interfaces    //built −in   tests
    }
```

In normal mode, the object are called as normal functions. In maintenance mode, the test cases are recalled and the output is obtained as follows:
T e s t r e s u l t 1 = BIT1 OK

How Built-in testing facilitates regression testing?
To facilitate regression testing within built-in testing basically two perspectives needs to be checked. Firstly the component developer would develop test interfaces and place them within the component to check the relation between specific input and changed point. Secondly, the component user would call only those test cases whose test interfaces can be used for regression testing. The method for performing regression testing through built-in testing has been mentioned in [5] along with the test selection criteria.

Will built-in testing shorten software total development time?
In most cases, it would definitely decrease total development time as the test cases and test interfaces are embedded within the component itself and they provide integration and regression testing. But the time required in testing the requirements and analysis phase would be the same as any other software development system. Hence considerable differences cannot be observed as the component user to perform acceptance and regression testing will need to select specific test cases from already available test suites.

The above questions address most of the issues in component based software development through built-in testing approach. In the following section the built-in testing approach as proposed in [4] is discussed.

Built-in Contract Testing
The built-in testing addresses the following two places where test suites should be located in the component based software development.

1) To embed test cases and test interfaces within the component so that servers at client side can be tested and see if the component works as expected.
2) At client side to check the component works according to the semantics mentioned in the contract.

The following figure illustrates the above two points.

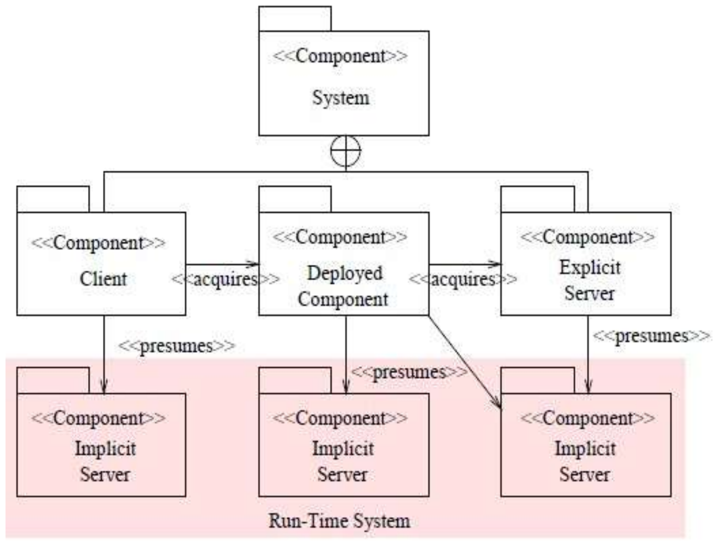

Figure 1.Component Deployment in System

Above two points can be categorized into two components namely tester component and testable component.

Tester Components
Instead of embedding test suites within prefabricated component, the tester components are those which are embedded with the software system to test the entire system. These tester components are required to check the servers at the client side to ensure that the component behaves as expected. The tester component includes one or more built-in tests and is executed after assembling the component within the system software.

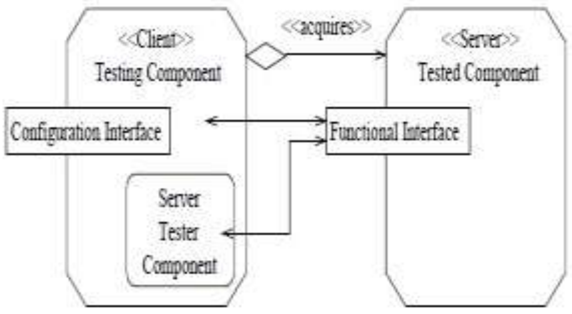

Figure 2.Testing component with server component

Testable components
The testable components are the traditional built-in testing components which are embedded with test cases and test suites. These components not only include the code to traditional components but also test suites. They exhibit the properties of encapsulation so the component users do not have any idea of the functionalities and test suites embed within the component by the component developer. As shown in figure the testable component includes a testing interface along with a functional interface. The testing interface is just like another java-like interface exhibiting testing contract.

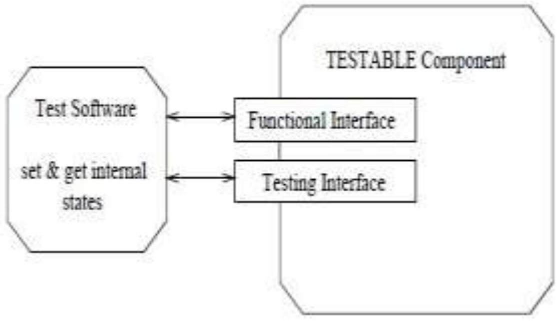

Figure 3 Testable component

CONCLUSION

Testing of component based software development is one of the recent research area and not many research has been performed in this area. Though using built-in testing in component based software development provides many benefits but there is trade-off here. With more added functionality comes more complexity in the code of the component, which creates pretty difficult for the component developer to code such a component. Built-in testing if deployed correctly within the component would provide all the testing benefits as it provides mechanisms for integration and regression testing along with maintainability and verifiability features.

REFERENCES

[1]. Sami Beydeda. “Research in testing COTS components built-in testing approaches”. In: Computer Systems and Applications, 2005. The 3rd ACS/IEEE International Conference on. IEEE. 2005, p. 101.
[2]. Daniel Brenner et al. “Reducing verification effort in component-based software engineering through

built-in testing". In: Information Systems Frontiers 9.2-3 (2007), pp. 151–162.

[3]. Sravan Kumar Pala, "Synthesis, characterization and wound healing imitation of Fe3O4 magnetic nanoparticle grafted by natural products", Texas A&M University - Kingsville ProQuest Dissertations Publishing, 2014. 1572860. Available online at: https://www.proquest.com/openview/636d984c6e4a07d16be2960caa1f30c2/1?pq-origsite=gscholar&cbl=18750

[4]. Jerry Gao et al. "On building testable software components". In: COTS-Based Software Systems. Springer, 2002, pp. 108–121.

[5]. Hans-Gerhard Groß. "Built-in Contract Testing in Component-based Application Engineering". In:

[6]. CologNet Joint Workshop on Component-based Software Development and Implementation Technology for Computational Logic. 2002.

[7]. Sravan Kumar Pala, "Advance Analytics for Reporting and Creating Dashboards with Tools like SSIS, Visual Analytics and Tableau", *IJOPE*, vol. 5, no. 2, pp. 34–39, Jul. 2017. Available: https://ijope.com/index.php/home/article/view/109

[8]. Chengying Mao. "Built-in regression testing for component-based software systems". In: Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International. Vol. 2. IEEE. 2007, pp. 723–728.

[9]. Yingxu Wang, Graham King, and Hakan Wickburg. "A method for built-in tests in component-based software maintenance". In: Software Maintenance and Reengineering, 1999. Proceedings of the Third European Conference on. IEEE. 1999, pp. 186–189.