



## International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

# Reduction of Cost while Maintaining High Customer's Satisfaction by Using CBSE Software

Vijay Kumar Sinha<sup>1</sup>, Meenakshi Jaiswal<sup>2</sup>

Research Scholar, IKG PTU, Kapurthala, Punjab, India<sup>1</sup>

Assistant Prof, CGC, Landran Mohali, IKG PTU, Punjab, India<sup>2</sup>

**ABSTRACT:** Cost reduction is a major issue with the industry of software engineering. Cutting the cost enhance the profitability margin while attracting widespread customers due to low cost. Reducing the cost often degrades the quality of software as it cut the cost by reduction of features, quality and performance as well as development time. Maintaining the quality and high satisfaction of client is a major challenge of software industry. We propose a CBSE based software development technique for cost cutting while maintaining the quality. By using the freely available open source ready components; reduction of software development time by rapid development by means of readily available in-house components; reduction of non-compatible software components; customization of client need based components which gives higher satisfaction. We achieve a significant cost reduction (~81.52%) over the other available software development methods.

**KEYWORDS:** CBSE, Non-Participating, System Compatible, Rapid development, Open Source

### I. INTRODUCTION

McIlroy [1968] first suggested techniques Software reuse, at NATO Software Engineering Conference, he predicted that mass-production of components of softwares would end the software crisis [7]. His objective was very clear: to make something only once and to reuse it many times by industry for low cost. Frakes and Terry [1996] – was first person to suggested metric and models on software reusability for low cost and rapid delivery. He recommended that models based on cost benefits, evaluating the maturity level, reclaim library metrics [9]. Kim [2005] takes – on the problem of component based software reusability for low cost production. It discusses the problems in understanding the component based software reuse and to discuss the pre – conditions required to meet before practicing softwares-reclaim on a wide scale in a formal manner [8]. Sharma et. al [2007] debates about managing component – based systems with recyclable components. It converses the reusability thoughts for components based systems and to discover the reusability metrics to calculate reusability direct or indirect ways. [13]. Anas al – badareen [2011] proposed a framework that covers the extraction, acceptance and storing of reusable software components [1]. Gupta and Kumar [2013] showed in a study about reusable software component recovery system. The study debates the techniques for storage and recovery of software components which can be reused by the industry [6]. In case of component based software engineering approach (CBSEA), the stress on components is added while designing software. A component can be considered or assumed as a sovereign system that completes a specific assignment. A component also can be composed of several other components. More specifically software components are prebuilt substances that act as a building block in a software to perform specific functions. These components can connect with each other using standard interface. After the solidification of component based software engineering practices, the notion of software recycle has progressed more significantly. We can say that a component based software engineering is a course that emphasized the designing and building of software using recyclable software components [11]. Reusability focus on programming software to composing software system [11]. There lies several benefits of component reuse like compact development time and cost, the quality and efficiency will also be increased as the pre verified components are being used. But the developers could not harvest much benefit from it. There may be several reasons for that but the lack of proper definition for the components which is to be reused can be the one reason. The non-availability of



# International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

arecognisedmeaning for a component can also create misperception and misconstruction among the developers. It can be explained with an example: If one purchase a stereo music system and bring it at house. We can fit various components similar to speakers, earphones etc. into it. Each of these components has been intended to fit a definite architectural style, the influences between components are homogenous, and communication protocols have been pre-established. Assembly of such components is cool than to build a system of thousands of discrete parts. We need to achieve this indeed [11]. A component essentially be deliberate in such a way that it should be able to integrate and communicate with other components in the same system easily. It also integrate and communicate with other components outside the system but within same domain. The CBSE integrate and communicate with applications outside the domain too. There are several queries about component reusability which are quiet unanswered, For example it can a large system be built by combining the reusable components, will it be possible to find the components that already exists, how the library of components can be created and made available [11], how the inducements will flow to the designer of a component. There is a must requirements for new methods and tool support that favours and supports the design and development and maintenance of reusable software components. The wide spread literate found in the component based software engineering and its various optimization techniques. Gaoyan[1] described about the verification technique for CBSE through formal analysis as well as traditional S/W engg techniques. Summarizing most of the principle research done in this field is: Gaoyan demonstrated an Automata-Theoretic approach for model checking. In this paper ,the Model-checking Black-box Testing Algorithms or program for Systems with Unspecified Components: Here the he presented both LTL (linear time temporal logic) and CTL (computation tree logic) model-testing algorithms for the systems with unspecified software components. The LTL (resp. CTL) formula about the system, directly deduce the condition in terms of the communication graphs. The another approach suggested by Egon et al. [2] He says that the without verifying components and interaction it was nearly impossible to robust systems. The Testing of such systems required combination of unit and integration tests, and must deal with the verifying contracts that enabled interaction of components. Fevzi Belli and Christof J. Budnik [3] proposed that it was widely accepted that traditional test methods were not necessarily appropriate for testing of component based software (CBS). As a result it also conventional test tools because similar drawbacks for the test automation of CBS based on their graphical user interfaces (GUI), because for any level of user-focused testing domain knowledge and knowledge about the implementation of the CBS are essential to run tests. Sami Beeydeda and Friedhofstr [4] proposed that it is very beneficial to use of CBSE based softwares for the gigantic software systems as it surely have benefits for cost cutting as well as faster delivery . However, CBSE complexity remains an issue in software engineering. The user of a component was generally found problems with the information's that is necessary is not available in general as it is black box and having different vendor make. In absence of adequate information distinguishes testing of components from other software entities that is called as non-component-based development. The author gave an overview of testable bean approach, built in testing approach and STECC approaches to testing component. This method described can simplified as component user's test insofar that the component user might not need to create test cases for testing. In another development Chengying Mao [5] suggested that some features of component, such as high resolvability, limited access support and implementation transparent, bring a great confront for testing the systems built by externally-given software components. This is applicable specially for regression testing. The Built-in test design was a highly effective way to improve component's testing. He presented a built-in regression testing techniques to legalize the change and its blow on component-based software, which desirable the mutual teamwork between the component users & component developer's team. Through employing preliminary experiments on some medium scale systems, their regression testing method based on built in test design has been proven to be feasible and practical. Although their method indicated the same precision as Orso et al.'s method at statement level, it needs less exchanged information [i.e., meta-data] and test scripts, so it was more cost-effective. Miao et al. [6] suggested to borrowing the belief of component communication. He developed the idea of Logic Component [LC] .In this paper a Web application was splitter into LCs which was mapped into the real physical components. They assumed that every component and LC was a black box that is well tested. The Web applications may be regarded as a collection of mutually interactive software components. It mainly focuses on testing the interactions of components of software. To model each component automation was used. The compositions of automata were used to model the component based interaction. For each component-test series, a new automaton can be used by composition of automata. Theoretical test cases can be created from the newer automaton. By the way of mapping the actions to the actual operations and adding the statistics of test space, the component test cases were



# International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

created. Zheng et al. [7] demonstrated that software yield was often configured with commercial-off-the-shelf (COTS) components. Whenever the new releases of these components were made accessible for incorporation and testing, source code was frequently not specified. There are wide regression test selections processes are developed and have been exposed to be cost efficient. However, the majority of these test selection techniques depends upon the access to source code for change detection. Based on their earlier work, it was studied the solution to regression testing COTS-based applications that include the mechanism of dynamic link library (DLL) files. They developed the Integrated - Black- box Approach for Component Change Identification (I- BACCI) technique that aims regression tests for component based application programs based upon static binary code analysis. The possibility case study was conducted at ABB on the CBSE software products written in some C/C++ language to demonstrate the efficiencies of the I-BACCI. As a results of the case study specify this process could cut the requisite number of regression tests by as much as 100 percentages. They propose that software crop were often configured with commercial-off- the-shelf (COTS) components. When new releases of these components were made available for incorporation and testing, source code was frequently not given. Different regression test selection techniques have been developed that is cost effective. However, the most of these test selection methods rely on access to source code for transform recognition. The whole work, were studying the solution to regression testing COTS-based programs that integrate components of dynamic link library (DLL) files.

Navneet et al. [8] introduced that rapid and quick development of software's can be made possible by means of CBSE. In CBSE, the software product was built by combining different techniques of on hand software from diverse suppliers or vendors. By means of this technique, cost and time of the software package reduced significantly. However in the testing stage there are many challenges for a software tester, due to limited access to the source code of the reusable component of the software product. The component meta-data could be used to join extra information with the components to facilitating of CBSE based software testing. The Black box testing was used as in this method the code of the component was not available. Generally, a component has a concealed interface and a tester are not able to put input values in it until its interface was not finished. The challenges in component based testing by use of metadata method for black box testing would be used when component's interface not exists. Here they demonstrated the techniques that how the metadata could be used in black box testing. Jiang et al. [9] again demonstrate in their paper that the adequate testing of black-box software components was a great basis before they reuse it in the concept of component based software development. The test-data production and test sufficiency ensuring were difficult issues for the unavailability of the source code of black-box components. They extended component interface specification model was proposed to support the component understanding, testing and reuse. Then the function of different kinds of specification elements in testing was defined. Based on the syntactic and semantic specifications, the proposed test-data generation method could produce test suite meeting a certain mutation score, which was viewed as a kind of effective test adequacy criterion. Last but not the least, many experiments were carried out and their results have shown that the diverse kinds of requirement could support the testing of black-box components.

## II. PROBLEM STATEMENT

Cost reduction is a major issue with the industry of software engineering. Cutting the cost enhance the profitability margin while attracting widespread customers due to low cost. Reducing the cost often degrades the quality of software as it cut the cost by reduction of features, quality and performance as well as development time. Maintaining the quality and high satisfaction of client is a major challenge of software industry. We propose a CBSE based software development technique for cost cutting while maintaining the quality. By using the freely available open source ready components; reduction of software development time by rapid development by means of readily available in-house components; reduction of non-compatible software components; customization of client need based components which gives higher satisfaction.

## III. OBJECTIVE OF RESEARCH

The principal objective of this research includes:

- a) To Study and examine the existing software development techniques and their development cost.
- b) Development of a method for reduction of cost while maintain the high customer satisfaction.



# International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

- c) Comparison of existing cost reduction method and the proposed system for the cost as well as the customer satisfaction,

## IV. RESEARCH METHODOLOGY: CASE STUDY

The component based softwares are an excellent solution for low cost softwares . By customization of CBSE we succeed to maintain the customer's satisfaction too.

- 1) Reduction of software development cost :

For software cost reduction we can use the freely available open source software components for developing the software. We take IFW Campus ERP component software as case study.

About IFW Campus ERP:

EduCamp is intranet/internet based web application that assists faculty, parents, students, & management employees to use the data of EduTech, access related reports and handle day to day processes from anywhere in the campus or anywhere in the world. It lets the users to interact with basic operation & information of EduTech as it is an inherited web tool for EduTech. EduCamp is a complete secured system as it picks up only the relevant entries for each student, faculty, parents & administrator area and shows them individually to the users with complete security. Also, EduCamp is an information sharing system which provides easy accessibility to all students, parents, faculty and administration to share information, notices, assignments etc.

### Components of IFW Campus ERP Modules:

Table -I

IFW Campus ERP Solutions Major Software Components				
	Administrative Login Components :	Student Login Software Components :	Parent Login Components :	Employee Module Components
1	Student listing	Fees module	Campus news	Subject specialization report
2	Faculty listing	Due fees report	Due fees report	D.R.R. (daily report register)
3	Faculty + student logs	Fees invoice	Fees module	Milestone management
4	Download manager	Fees deposited	Fees invoice	Access exam time table
5	D.R.R. (daily report register)	Attendance record	Fees deposited	Attendance of students
6	Academic calendar	Academic time table	Student guidelines	AttendanceReport
7	Milestone management	Exam time table	Attendance record	Book submission reminder
8	Milestone status of faculties	Exam sitting arrangement	Book submission reminder	Access salary information
9		Performance in exam	Academic time table	Student guidelines
10		Library book searching & reservation	Performance in exam	Campus news



# International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

11		Teacher evaluation test	Exam sitting arrangement	Input exam date
12		Student guidelines	Training & placement	Access time table
13		Book submission reminder	Exam time table	Downloads
14	Library book searching & reservation	Downloads	Academic calendar	Self-attendance
15	Training & placement management	Academic calendar	News manager	Training & placement
16		Training & placement	Downloads	Library book searching & reservation
		Campus news		Academic calendar

(Here Red marked components are must have components whereas Blue marked components are optional components and blue marked components are optional.)

Cost Reduction Process:

1. By Use of Open Source Freely Available Software Components :

Cost-Quality Table			
	Components	Average Development Cost from Scratch (inRs. )	Open Source Components integration cost (inRs. )
2	Front End Elements	25000	Nil
3	Back End elements	35000	Nil
4	E-Commerce	55000	Nil
5	Shopping Cart	655000	Nil
6	Security	76000	Nil
7	Multi-media	45600	Nil
8	Downloadable files	12000	Nil
9	Password protected sections	9000	Nil
10	Online databases	15000	Nil
11	Site Search	12300	Nil
12		76000	Nil
Component Integration + Customization Cost			185000/-
			1015900/-
Savings = Rs. 830900/ = <b>81.789%</b>			Rs. 185000/
Saving through Component Reuse			
	Components	Average Development Cost from Scratch (inRs. )	Component Reuse inclusive of Customization cost
2	Front End Elements	25000	5000
3	Back End elements	35000	7000
4	E-Commerce	55000	11000
5	Shopping Cart	655000	13300
6	Security	76000	15500
7	Multi-media	45600	9400



# International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

8	Downloadable files	12000	2500
9	Password protected sections	9000	2100
10	Online databases	15000	3100
11	Site Search	12300	3200
12	Database Management	76000	15600
Additional integration cost			100000
		1015900/-	Rs. 87700/
Savings = Rs. 828200/ = <b>81.52%</b>			

The above two table shows that the component reuse incurred almost 81% cost reduction and by using of open source software it again reduced by almost 81% of total cost. By use of joint strategy we can almost (81.52+81.78) / 2 = ~ 81.65% cost can be reduced which is a great success.

## V. CONCLUSION AND FUTURE SCOPE

The cost reduction or cost cutting challenges are very important for software industry in present days. There are many ways to achieve this goal and make the s/w industry more efficient and meaningful by delivering low cost and high quality softwares. Astonishingly the cost increases for the softwares are mainly not because of the production cost or time for evolution or complexity of applications and competence of people to get it executed. Generally it occurs because of some non-technical mismanagement. So a small attention to these issues can have significant cost reduction impact on the CBSE softwares. Component based softwares have the advantage of rapid delivery by adding or removing components. By use of freely available components we reduce the software cost significantly. The rapid development and software reuse also reduce the software development cost. Easily Adding and removing the software components and customization as per client's requirements keep the satisfaction level high. We successfully achieved 81.65% of cost reduction by using joint strategies. This research can be further developed as reduction of software maintenance cost by providing client's self-maintaining features for normal maintenance while online maintenance system for complex updates. Self-maintenance user's manual will also a great help in cutting the maintenance cost reduction.

## REFERENCES

- [1] Al – Badareen A., Selamat M.H., Jabar M.A., “Reusable Software Component Lifecycle”, International Journal of Computers, 5(2), pp. 191 – 199, 2011.
- [2] Gupta S., Kumar A., “Reusable Software Component Retrieval System”, International Journal of Application or Innovation in Engineering and Management, 2[1], pp. 187 – 194, 2013.
- [3] Imeri F.; Antovski L., “An Analytical View on the Software Reuse”, ICT Innovations 2012, Web Proceedings of the 4<sup>th</sup> ICT – ACT Conference, Ohrid - Macedonia, ISSN: 1857 – 7288, pp. 213 – 222, 2012.
- [4] Kim W., “On Issues with Component – Based Software Reuse”, Journal of Object Technology, 4[7], pp. 45 – 50, 2005.
- [5] McLroy M.D., “Mass Produced Software Components”, Software Engineering: Report on a Conference by the NATO Science Committee, Brussels, pp. 138 – 155, 1968.
- [6] Pressman R.S., *Software Engineering – A Practitioner’s Approach*, 5th ed. New York: McGraw-Hill Inc., ISBN: 0073655783, 2001.
- [7] Sharma A., Kumar R., Grover P., “Managing Component – Based Systems with Reusable Components”, International Journal of Computer Science and Security, 1[2], pp. 60 – 65, 2007.
- [8] Gaoyan Xie. —Decompositional Verification of Component-based Systems—A Hybrid Approach. *Proceedings of the 19th International Conference on Automated Software Engineering [ASE’04], IEEE, 2004*, pp. 414-417.
- [9] Egon Valentini, Gerhard Fliess and Edmund Haselwarter. —A Framework for Efficient Contract-based Testing of Software Components. *Proceedings of the 29th Annual International Computer Software and Applications Conference [COMPSAC’05], IEEE, 2005*, p.219-222.
- [10] Fevzi Belli and Christof J. Budnik. - Towards Self-Testing of Component- Based Software. *Proceedings of the 29th Annual International Computer Software and Applications Conference [COMPSAC’05], IEEE, 2005*, pp.205- 210.
- [11] Sami Beydeda. —Research in Testing COTS Components - Built-in Testing Approaches. 3rd ACS/IEEE Conference on Computer Systems and Applications, 2005 IEEE, 2005, p.101.
- [12] Chengying Mao. —Built-in Regression Testing for Component-based Software System. 31st Annual international computer software and Applications Conference [COMPSAC 2007], IEEE, 2007, p.723-728.
- [13] Huaikou Miao, Shengbo Chen, Huanzhou Liu and Zhongsheng Qian. An Approach to Generating Test Cases for Testing Component-based Web Applications. Workshop on Intelligent Information Technology Application. IEEE, 2007, p.264-269.





# International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

- [14] JiangZheng, Laurie Williams, Brian Robinson and Karen Smiley. —Regression Test Selection for Black-box Dynamic Link Library Components. Second International Workshop on Incorporating COTS Software into Software Systems: Tools and Techniques [IWICSS'07]. IEEE, 2007, pp.9.
- [15] Navneet Kaur, Ashima Singh, "Generating More Reusable Components while Development: A Technique, International Journal of Innovative Technology and Exploring Engineering , Volume-2, Issue-3, February 2013.
- [16] W. B. Frakes and K. C. Kang, "Software reuse research: status and future," IEEE Transactions on Software Engineering, vol. 31, pp. 529-536, 2005.
- [17] M. Morisio , "Success and Failure Factors in Software Reuse," IEEE Transactions on Software Engineering, vol. 28, pp. 340-357, 2002.
- [18] DanLaurențiuJișa , "Component Based Development Techniques – comparison ," International Conference on Computer Systems and Technologies – CompSysTech,2004.
- [19] GillandGrover , "Reusable software components," Advances in computers, vol. 33, pp. 1-65, 1991.
- [13] SandeepKhimta, ParvinderS.Sandhu,andAmanpreetSinghBrar,"AComplexityMeasure forJavaBeanbasedSoftwareComponents",WorldAcademy ofScience,Engineeringand Technology, Volume 42, 2008.
- [20] Richa Mittal , ashima Singh "Refining Complexity Metrics of Component Based Softwares by takingAverage Use Factor in Black Box Testing" Vol 10 no. 2 , pp 1881, 2014

## BIOGRPHY

	Vijay Kumar Sinha received his M.Tech. degree in Computer Science and Engineering from Punjabi University , Patiala. Currently he is persuing his Ph.d. degree in Computer Science & Engineering from IKG Punjab Technical University , Kapurthala (Punjab) India. His research interests include Image Processing , Software engineering , Software components, Software Reuse, Software architecture and software metrics.
	Ms. Meenakshi Jaiswal is working as Assistant Professor at Chandigarh Engineering college , Landran , Mohali (Punjab) , under IKG-PTU . Her area of research interest includes software engineering , Software reuse , Green Computing and big data.