J. Appl. Environ. Biol. Sci., 6(4S)77-81, 2016

ISSN: 2090-4274
Journal of Applied Environmental and Biological Sciences
www.textroad.com

Framework for Better Reusability in Component Based Software Engineering

Muhammad Tahir[1], Fazlullah Khan[1*], Muhammad Babar[2], Fahim Arif[2], Shahzad Khan[1]

[1]Department of Computer Science, Abdul Wali Khan University Mardan
[2]Department of Computer Software Engineering, National University of Sciences & Technology, Islamabad, Pakistan

Received: January7, 2016
Accepted: March 22, 2016

ABSTRACT

To develop software from existing component is done to reduce time and cost of the software. Reusable modules and classes reduce implementation time, increase the likelihood that prior testing and use has eliminated bugs and localizes code modifications when a change in implementation is required. Different traditional approaches to achieve reusability in Component Based Software Engineering (CBSE) have been discussed. This paper provides framework to develop component based software. This framework helps to achieve better reusability of the component by providing ease of component selection using domain knowledge and off-the-shelf-components availability. It also helps less skilled developer to develop successful component based software system.

KEYWORDS*:* Reusability, Component based Development, Component Selection, replacement component, Interface with Component.

1. INTRODUCTION

The primary goal of the software system is to achieve quality and efficiency of the desired work. Traditionally software development faces different challenges and fails due to these problems, as they exceed from defined budget and time [1]. Software development is becoming complex with the passage of time. To achieve the basic goal of the software it is not possible to make system from scratch because it will take more time and cost. Solution to this problem was to reuse the functionality. This can be achieved through Component based development (CBD) in which software is divided into small pieces called components. These components have independent existence and desired functionality. These components are tested when they are built. Using these components in new system it saves cost and time and increases quality and demand of the software [1]. The approach to reuse components is not new. Classical designs of complex software systems also initiate with the identification of system parts designated subsystem and blocks. The recent emergence of latest technology and tools has increased the importance to make applications from reusable components. There are certain off the shelf components which provide components for reuse. They only provide an interface which put designers in a difficulty to handle the interactions between the interfaces [2]. At the same time [3] describes the benefits of COTS in reducing time and cost in the market. "Experience has also shown that component based development requires systematic approach and focus on the component aspects of software development" [1]. Goal of component-based software engineering is to provide support to:

- Assemble the components
- Components are developed to assure reusability in future.
- Maintenance and upgrading of systems by providing customization and replacement of components.

The rest of the paper is organized as follows. Section 2 discusses Related Work; Section 3 explains the Reusability in Component based Software Engineering. Section 4 elaborates the Activities of Component based Software Engineering. In Section 5 the Framework for the Component based Software Engineering in explained, and Future work is provided in Section 6, whereas Section 7 concludes the paper.

2. RELATED WORKS

Don Batory and Sean O`Malley has presented a domain independent model for hierarchal software system design, based on the software components which are interchangeable and ensure large scale of reuse. The model is based on the concepts of two independent projects Genesis (domain model for Database management system) and Avoka (domain model for network software systems). Domain modeling formalizes the similarities and differences

[*] **Corresponding Author:** Fazlullah Khan, Department of Computer Science, Abdul Wali Khan University Mardan, Pakistan
fazlullah@awkum.edu.pk

among system of a domain. Authors believe that this "model is a blueprint for achieving software component technologies in many domains" [7].

Colin Adkinson and Dirk Muthig presented a tutorial to represent new method for Component based software engineering known as KobrA. Reuse of different components is done in Component based Software's due to which when integration is done there are certain integration problems which are difficult to handle [2]. These issues are handled by this KobrA approach.

Ed Mancebo and Anneliese Andrew in there paper "A strategy for selecting multiple components" have presented a systematic method to define software architecture and selecting of-the-shelf components (for reuse) simultaneously. The method is based on existing techniques for selecting component and evaluating architecture .Authors have identified architectural decisions as they have large effect on the components which are used at the early stage of analysis and design, alternates are also investigated. This method provides way to do architectural design with components reuse. Limitation of this method is that it depends on the designer skill that how he handles the architecture of the system and the components reuse. Authors have used an example system "Reaction! Presentation software", through out in the paper to explain their view point [5]. Nasib S. Gill in his survey paper discusses the characteristics of component reuse. He discusses the importance of proper characterization of the components that helps developer to reuse the component [4]. David C. Sharp and St. Louis, Missouri have discussed a case study to evaluate the ability of reuse of "operational flight program" and has also described the key aspects of the component based logical architecture used for designing in it. Authors have discussed "In 1995, an initiative was launched at McDonnell Douglas (now merged with The Boeing Company) to assess the potential for reuse of operational flight program (OFP) software across multiple fighter aircraft platforms, and to define and demonstrate a supporting system architecture based upon open commercial hardware, software, standards and practices.' This initiative produced a set of reusable object oriented navigation software, which was flight tested on several different aircraft platforms hosted on different hardware configurations (both MIPS@ and PowerPCO based)" [1].

3. **Reusability in Component based Software Engineering.**

To build complex systems software developer always built from smaller components to avoid risk in the system. These smaller components are integrated to make the whole system. New system is built from using existing components in component based software. There are two engineering techniques in the development of component based systems which are as follows:

- **Reuse Technique:** To create more complex systems by using existing components.

Evaluation Technique: Software that can be maintained easily by making it highly componentized.

In a software life cycle changes in requirement occurs during development and after it is deployed. System that is well designed provides ease to cater the changes these are local to the system which have little or no effect on other components. These changes can be achieved with replacing components with some other component which are licensed and can be used easily. There is also a component model that can support components placement and their interaction. There must be some process and architecture that support component based development [6].

4**. Activities in Component based Software Engineering.**

Framework provided in Figure 1 shows the framework for CBSE. It contains certain additional activities from software engineering processes and these are explained below:

4.1 Component Selection. Component selection has always been a problem in the Component based software engineering. Traditional approaches have discussed Component based software engineering but
didn't provided a framework to select the component, which leads less skilled developer in the difficulty. Component selection is the most important activity in the component based software engineering, it not only leads to the success of the component as it saves time and cost of the development but also provides customer satisfaction. In Figure 1 we have proposed the better way of component selection. Software engineer selects the components from the "Available components in the Domain". A component which exists in the domain are testable and follows certain standards, hence they are compared with the desired functionality we want to achieve from the component. If desired functionality is achieved then we will match the cost with the component selected. If cost to buy the component is less then cost to develop the component we will buy that component. If the price of component to buy is higher, then negotiate with the customer to pay more, if customer doesn't show agreement then check some other alternative component. If no alternate component is available check customized component as they also help in reducing time and cost. If customized component is available then buy it else make or create your own component.

4.2 Interface with Component. Once the component is selected then it is used in the design and during implementation phase these components are merged with other components. Interfacing is a problem when the components developed are not compatible with the software developed. Component of the shelf and customized components require certain implementation code to provide interface with the system.

4.3 Replacement. Maintenance is an ongoing process once the software has been built and delivered. In Component-based Software Engineering there are certain requirements that needs to be changed or certain additional functionality needs to be added in the Software. For that replacement has been done.

5. Framework for Component based Software Engineering.

Component based software engineering helps in reducing cost and time. Framework shown in Figure 2 helps less skilled developer to develop a successful system. Difference between other process models and the Component based software engineering is that later requires component selection to reduce the time and cost. This framework shows all the activities which other process models possesses.

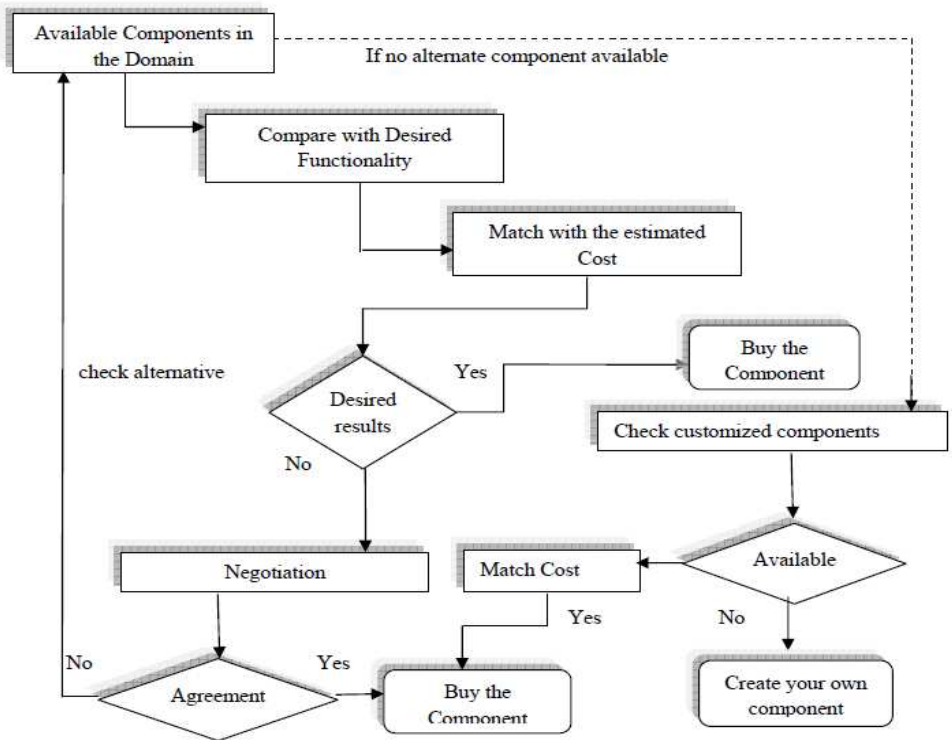

Figure 1: Component Selection for Component based Software Engineering

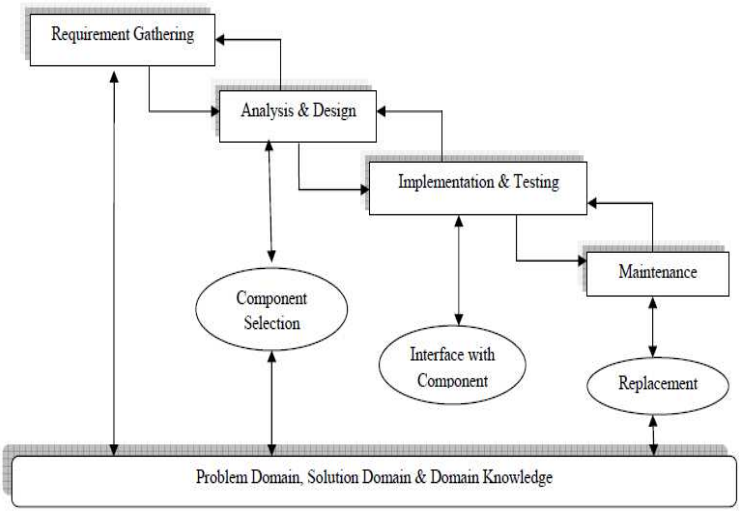

Figure 2: Framework for the Component-based Software Engineering

Requirement gathering is the activity in which we gather the requirements of the software that needs to be built. This has been done on the basis of Domain Knowledge and the problem/solution domain. Domain knowledge is very important for the success of any project. After gathering requirements analysis has been done to verify that the requirements are correct and feasible. This will lead in making design of the system. During design system has been divided into components. If components are available in the market and in the domain of the system, they are selected as explained in section 4 on the basis of domain knowledge. These components will reduce time and cost of making component. If the component exist they are used in the design, otherwise you need to create your own component. During implementation phase we are implementing the code of the component which is not available in the market and in the domain, and then interfacing with the components which have been selected for the software. At the same time we are doing unit testing of the component which has been created by the team and integration testing when merge the new component with the other system. After the testing software has been deployed and tested at user end this is the maintenance phase. In this phase there is certain component which might create problems when they are deployed, these components will be checked and corrected but if these components are those which have been purchased they need to be replaced. After delivering the system there might some requirement which might change later and for that we need some component which provides us additional features again we have to replace the component or buy the new one. This replacement of the component will also be done using the knowledge of domain as shown in the Figure 2.

6. **Future Work.**

Framework for the Component based software engineering has been presented but the implementation of this framework will be done in the future to demonstrate that how this model helps in achieving the better reusability. Furthermore, it can be extended to various communication networks for better reusability and implementations such as Wireless Sensor Networks [8, 9, 10, 11, 12, 13, 14, 15] and Internet of Things [16-20].

7. **Conclusion.**

In this paper we have presented a framework for Component based software engineering and for the selection of the component provided a model which will lead less skilled person to select the component from the domain. This framework helps in achieving the reusability of the component and in enhancing the knowledge of less skilled developer.

REFERENCES

[1] David C. Sharp, The Boeing Company, St. Louis, Missouri. Reducing Avionics Software Cost through Component Based Product Line Development. 15" AIMIEEE Digital Avionics Systems Conference Proceedings, October, 1996, page. 401 - 406

[2] Colin Atkinson and Dirk Muthig. Component Based Product Line Engineering using UML. C.Grecek(Ed): ICSR7, LNCS 2319, pp. 343-344, 2002.

[3] Clemens Szyperski, Jan Bosch, and Wolfgang Weck. Component-Oriented Programming. A. Moreira and S. Demeyer (Eds.): ECOOP'99 Workshops, LNCS 1743, pp. 184{192, 1999.

[4]Nasib S. Gill. Importance of SoftwareComponent Characterization For Better Software Reusability, ACM SIGSOFT Software Engineering Notes, Issue 1, Vol. 1, 2006.

[5]. Ed Mancebo and Anneliese Andrews. A Strategy for Selecting Multiple Components. SAC'05 March 1317, 2005, Santa Fe, New Mexico, USA, 2005 ACM Symposium

[6] Jon Hopkins, Component Primer. Communication of ACM: October 2000/Vol. 43, No.10.

[7] Don Batory and Sean O'Malley. The Design and Implementation of Hierarchical Software Systems with Reusable Components, ACM TransactIons on Software Engmeer]ng and Methodology, Vol 1, No 4, October 1992, Pages 355-39

[8] M. A. Jan, P. Nanda and X. He, "Energy Evaluation Model for an Improved Centralized Clustering Hierarchical Algorithm in WSN," in Wired/Wireless Internet Communication, Lecture Notes in Computer Science, pp. 154–167, Springer, Berlin, Germany, 2013.

[9] M. A. Jan, P. Nanda, X. He and R. P. Liu, "Enhancing lifetime and quality of data in cluster-based hierarchical routing protocol for wireless sensor network", 2013 IEEE International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing (HPCC & EUC), pp. 1400-1407, 2013.

[10] F. Khan. "Secure Communication and Routing Architecture in Wireless Sensor Networks" in *the 3rd Global Conference on Consumer Electronics (GCCE)* (p. 4). Tokyo, Japan: IEEE Tokyo, 2014.

[11] F. Khan, K. Nakagawa, "Comparative Study of Spectrum Sensing Techniques in Cognitive Radio Networks" in World Congress on Computer and Information Technology, 2013, pp.1-8

[12] M. A. Jan, P. Nanda, X. He and R. P. Liu, "PASCCC: Priority-based application-specific congestion control clustering protocol" Computer Networks, Vol. 74, PP-92-102, 2014.

[13] Mian Ahmad Jan and Muhammad Khan, "A Survey of Cluster-based Hierarchical Routing Protocols", in IRACST–International Journal of Computer Networks and Wireless Communications (IJCNWC), Vol.3, April. 2013, pp.138-143.

[14] Mian Ahmad Jan and Muhammad Khan, "Denial of Service Attacks and Their Countermeasures in WSN", in IRACST–International Journal of Computer Networks and Wireless Communications (IJCNWC), Vol.3, April. 2013.

[15] M. A. Jan, P. Nanda, X. He and R. P. Liu, "A Sybil Attack Detection Scheme for a Centralized Clustering-based Hierarchical Network" in Trustcom/BigDataSE/ISPA, Vol.1, PP-318-325, 2015, IEEE.

[16] M. A. Jan, P. Nanda, X. He, Z. Tan and R. P. Liu, "A robust authentication scheme for observing resources in the internet of things environment" in 13th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 205-211, 2014, IEEE.

[17] F. Khan, F. Bashir, K. Nakagawa "Dual Head Clustering Scheme in Wireless Sensor Networks". *in the IEEE International Conference on Emerging Technologies* Islamabad: IEEE Islamabad. 2012

[18] F. Khan, S. A. Kamal, F. Arif "Fairness Improvement in long-chain Multi-hop Wireless Ad hoc Networks". *International Conference on Connected Vehicles & Expo* (pp. 1-8). Las Vegus: IEEE Las Vegus, USA, 2013.

[19] F. Khan "Throughput & Fairness Improvement in Mobile Adhoc Networks". In *the 27th Annual Canadian Conference on Electrical and Computer Engineering*. Toronto, Canada: IEEE Toronto, 2014.

[20] S. Khan, F. Khan. S.A. Khan, "Delay and Throughput Improvemenet in Wireless Sensor and Actor Networks" *in 5th National Symposium on Information Technology: Towards New Smart World (NSITNSW)* Riyadh: IEEE Riyad Chapter, 2015.