International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 2, Issue 12, December 2015

Optimized Component Development Life Cycle for Optimal Component-Based Software Development

Gaurav Kumar
Research Scholar, Punjab Technical University
Kapurthala, India
gauravk6in@yahoo.com

***Abstract*—**The traditional software development approaches are not optimal for developing the complex software system using the reusable software components as compared to the Component-based software development approaches, because traditional approach almost try to attempt to develop the system through scratch. That is why, component-based software development (CBSD) continues to be a key area of research in the systems development and software engineering fields as it focuses on the integration of pre-fabricated software components to build systems characterized by increased portability and flexibility. The main objective of CBSD is to build systems by assembling pre-existing software components. The existing CBSD life cycles have not discussed the optimal selection of components when there is unclear and ambiguous user requirement. Majority of the existing CBSD life cycles are based on two processes, one of which deals with the development life cycle of component for reuse and the other deals with the development life cycle of Component-Based system as whole. This paper discusses a single and an optimized component development life cycle which deals with the development of optimal Component-Based systems by selecting optimized components when there is unclear, ambiguous and more general user requirements and there is no specific user requirement and this proposed development life cycle performs testing both at component level and system level and that too by not having two separate life cycles.

Keywords —Component-Based software Development (CBSD), optimized components**.**

I. INTRODUCTION

The global market has become an intensely competitive environment moving at an accelerating rate of change. To gain the strategic advantages of speed and flexibility, corporations must remodel their business processes, then rapidly translate that model into software implementations. So, the need to reduce time to market for software products is extremely crucial in today‟s software industry[13]. Software organizations are facing simultaneous pressures to reduce time to market, reduce the cost of the product, improve the productivity of the organization, increase the reliability of the product and increase the quality of the product. In a recent study it was concluded that 80% of the studied software development projects did not result in a customer satisfying solution. The most common reasons for this was exceeded budget or time to market[14].

A software component is any element of software life cycle that can potentially be reused in several contexts. A software component is independent and reusable. It offers explicitly specified services through an explicitly specified interface. It can affect /be affected by other software components. It should have one documented specification. It can have several independent implementations, i.e. one component can be implemented in several different programming languages and can have several binary (executable) shapes, i.e. one component can be executed in different software environments. To be more specific, a component is a unit of independent deployment, a unit of third –party composition and it has no persistent state. An optimized software component is self- contained, clearly identifiable art efect that describe and/ or perform specific functions and have clear interfaces, appropriate documentation and a defined reuse status. To obtain optimized software components, and an optimal Component-Based software, a single and more mature development life cycle is required instead of having two separate life cycles one for component and other for Component-Based system. This development life cycle includes various sequential phases like domain engineering, software analysis and specification, component design for identification & qualification of the components for domain specific and other future but not established generic domains, creating component repository, search subset of components from repository, retrieving set of components according to specific & generic application domain by following optimal component selection by using any signature and specification matching approach when specific user requirements are given and when there is unclear and ambiguous requirements, the greedy optimal selection approach is used followed by component testing, component deployment, composition of components for component based development, system testing, deployment, maintenance.

In this paper Section I introduces software component and an optimized software component along with the activities performed during various phases of

proposed development life cycle of optimized components for developing optimal component-based system. Section II discusses the brief literature review conducted, Section III focuses on differences between CBSD and traditional software development. Section IV discusses various CBSD life cycles. Section V deals with the proposed optimized component development life cycle from optimal component-based systems. Section VI presents a comparative study of proposed development life cycle with existing life cycles. Section VII concludes the paper and Section VIII discusses future scope.

II. LITERATURE REVIEW

Sutherland, J. (1996) proposed remodeling of business processes by corporations to gain the strategic advantages of speed and flexibility[13].

Dellarocas, C. (1997) investigated that the traditional software development approaches are not optimal for developing the complex software system using the reusable software components as compared to the component-Based Software Development approaches, because traditional approach almost try to attempt to develop the system through scratch[4].

Veryard, R. (1998) studied various software development projects to understand their reasons of success or failures in customer satisfaction[14].

Crnkovic I.et.al. (2002) studied that CBSD technology develops the software system by using reusable components from component repository. This technology mainly discussed the architectural design and implementation. This technology used COTS components for developing the software system and reported reduced implementation time and increased reusability of COTS components[2].

Sommerville (2004) proposed a CBSD process[12].
Crnkovic, I. et. al. (2005) developed adapted V development process for CBSD[3].
Capretz, L.F. (2005) proposed a Y model for CBSD[1].
Slyngstad N.P.O. (2006)

Studied the attitudes towards software reusable component among developers allowing us to attain more detailed information on benefits of reuse as well as success evolution factors for software reuse[11].

Vescan A. and Pop H.(2008) reported that CBSD is a latest technology for the development of the complex software components with the help of using the COTS software components. The main objective of this technology is to satisfy the client-specific requirements. Software has so many benefits like.

a) To increase the productivity of software.
b) To reduce the development time.
c) To reduce the cost.
d) To increase the quality of software.
e) To improve the reliability.
f) To reduce the interdependency or coupling.
g) To increase the cohesion.
h) To increase the performance of software system.
i) To increase the productivity of system.
j) To ease the maintenance of software system.

Because Researchers use the reusable software components which are already tested, so software component selection technique is a difficult task in CBSD[15].

Fahmi S.A. and Choi H. J. (2008) conducted a study of various life cycles proposed for CBSD[5].
Khan Ali M. and Mahmood S. (2010) stated that with the help of polynomial time, user can solve the problem of software component selection. This article also discussed the problem of selecting software components to optimize one or more non-functional attributes[6].

Kwong et.al. (2010) has described an optimization model for selection of optimal component[7].
Lau K.K. et. al. (2011) proposed and implemented the W-Model for CBSD by using model driven engineering approach[8]. -

Pande J.(2012) reported that most software companies want to develop the software using the CBSD. This technology has some benefits like reduced development cost, increased reliability and less time to develop[9].

Pande J. (2013) reported that CBSD is a latest development technology for developing the optimal software components with reduction of development cost, development complexity, development time and increased software reliability, interoperability. Software reuse increases productivity of software, reduces the cost and improves the quality of software. Main benefits of reusable software components are ease of maintenance, reduced development time and improved reliability[10] Vinay et. al. (2014) proposed W-shaped model for component selection and discussed various steps required for selection of components and product development process[16]

III. DIFFERENCE BETWEEN CBSD AND TRADITIONAL SOFTWARE DEVELOPMENT

TABLE 1. Differences between CBSD and traditional software development[5].

Component-Based Software Development	Traditional Software Development
Building system from pre-existing components.	Building system from scratch.
Components and systems integrated from those components are developed.	Software system is developed.
Component selection and evaluation are special lifecycle phases.	In the lifecycle, there is no special phase like that.
Much effort is required in the selection of components, testing & verification phase.	Much effort is required for system development
Reusability is the main theme.	Reusability usually not considered.

IV. VARIOUS EXISTING CBSD LIFE CYCLES

Figure 1. The CBSD Process [12].

A. *The CBSD Process by sommerville Figure*

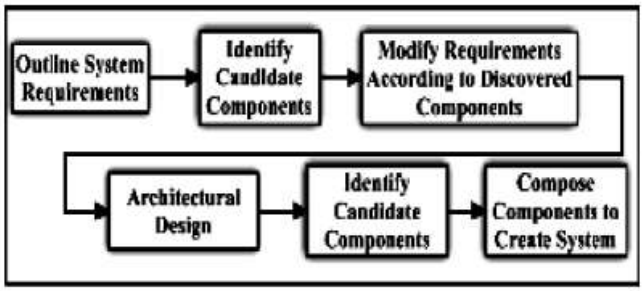

Figure 1. The CBSD Process [12].

In CBSD process by Sommerville shown in Figure 1, the user requirements are developed in outline rather than in detail as specific requests limit the number of components that might be used. A complete outlined set of requests are used to identify as many components as possible for reuse. Requirements are refined and modified so that they can comply with components. Architectural design is developed. After system architecture is designed, identify candidate components & modify requirements according to discovered components steps may be repeated. Finally the components are integrated which turns into a complete system.

B. *V- Development Process for CBD*

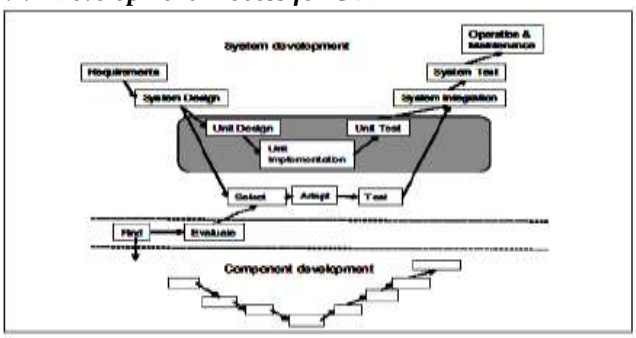

Figure 2. V- Development Process for CBD[3]

Ivica Crnkovic described a "V" development process for component-based development shown in Figure 2. For components that can possibly be used, the requirements are analyzed and specified in requirements analysis & specification phase. Potential components compliant with a particular component model are designed in system and software design. Design units are usually implemented as assemblies of several components and a glue code. The system integration process includes integration of standard infrastructure components that build a component framework and the application components. The integration of a particular component into a system is called a component deployment which is basically the mechanism of integration. The standard tests and verification techniques are used in system verification and validation.

C. *Y- Model for CBSD*

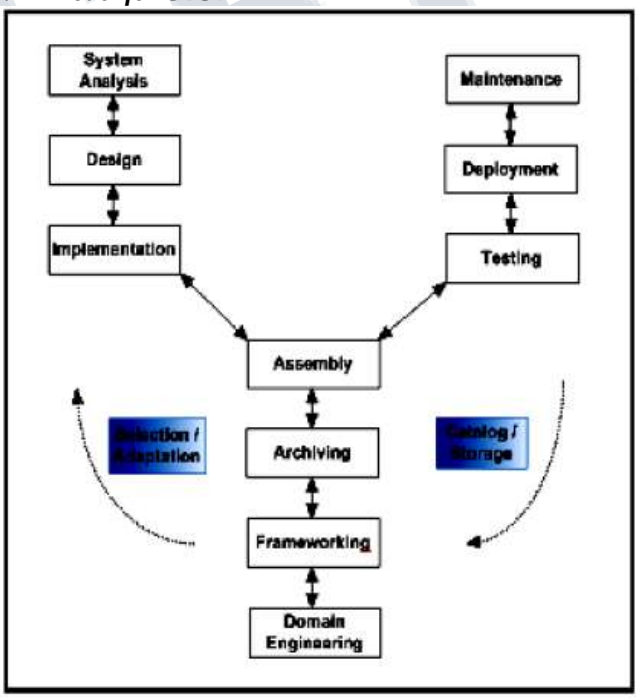

Figure 3. Y- Model for CBSD [1].

In Y model shown in Figure 3, domain engineering phase discusses areas of commonality and ways to describe it using a uniform vocabulary. The essential properties of a particular domain are captured and initial candidate for re-usable components come out. The frameworking phase acts as the skeleton for developing software in a certain application domain and at the same time recognizes components and identify inter-relationships within the

application domain. In assembly phase, the developers basically select reusable components or frameworks from the specific application domain. In archiving phase, cataloging and storing of newly developed components is performed.

D. *The W- Model for Component-based Software Development*

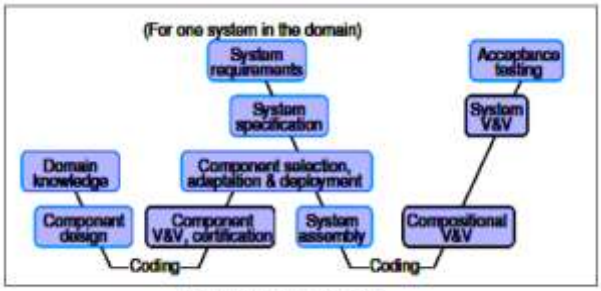

Figure 4. The W- Model[8].

The W-Model given in Figure 4, consists of two conjoined V models, one for the component life cycle and one for the system life cycle. These two V models are conjoined via the step of component selection, adaptation, and deployment. This „double V" process is called W-Model.

V. PROPOSED DEVELOPMENT LIFE CYCLE OF OPTIMIZED COMPONENTS

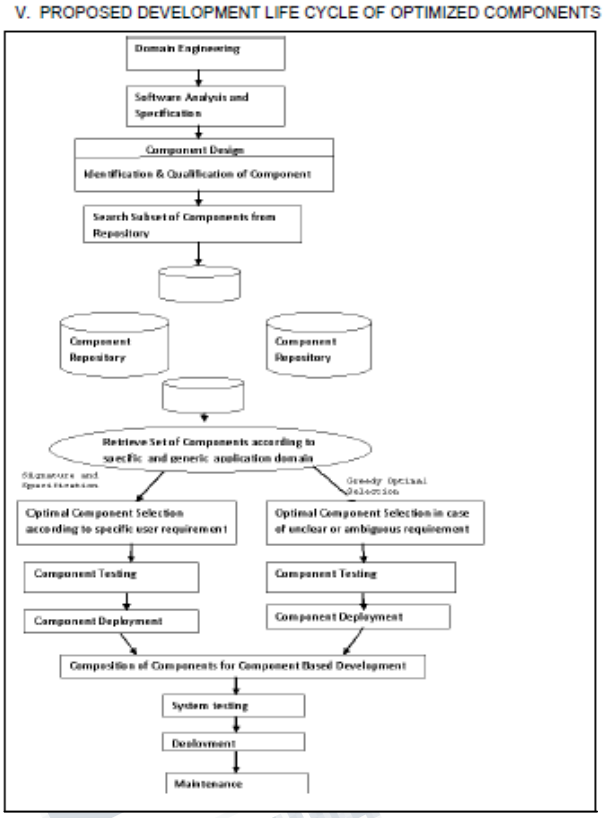

Figure 5. Proposed Optimized Component Development Life Cycle.

The Figure 5. shows the the proposed optimized component development life cycle for component based system.

A. Domain Engineering

The main aim of this phase is to develop a mechanism which helps in identification of software components and to reuse them for CBSD according to the problem domain. Domain engineering includes the domain analysis, design and implementation process which help in identification and selection of specific application domain of component-based software.

B. Software Analysis and Specification

The software analysis phase emphasizes discovery of high-level components in a real-world application and decomposition of the software system. It requires thorough information about the application domain where the components are to be deployed.

C. Component Design for Identification and Qualification of Components

In this phase the process of identifying components is performed according to the application domain and after identification the formal specification or documentation is performed which is known as qualification process which helps the new developers to understand the functionality of the components with ease.

D. Creating Component Repository

Component repository is a collection (store) of software components and contains the related information about components. These information are name, hardware platform, and required operating system and programming language(high level language, low level language) etc. The main aim of this phase is to create a repository of identified and qualified components for reuse which helps to reduce the development time, cost, risk and improves the quality by storing the reusable software component with good quality. The subsets of components are searched which help in retrieving optimal components. After this phase the components are available for reuse.

E. Retrieve Set of Components according to Specific and Generic Application Domain

This phase is used for optimal selection of components when the specific user requirements are known in advance by using some already established signature and specification matching approaches. This phase also helps in finding optimal components in case where the user requirements are more general, unclear and ambiguous and not known in advance in specific form,

F. Component Testing

In this phase the verification and validation of individual components selected in previous phase is performed.

G. Component Deployment

In this phase the the components are finally deployed in the application software after which the integration of the components is performed during composition phase.

H. Composition of Components for Component- Based Development

Components are integrated with each other with or without the help of a glue code. This phase requires much effort to automated means to verify that it satisfies specified requirements or to identify differences between expected and actual results.

I. System Testing

After composition phase, System testing is performed which is the process of exercising or evaluating a system by manual or automated means to verify that it satisfies specified requirements or to identify differences between expected and actual results.

J. System Deployment

The system deployment phase is characterized by implanting the software at user site.

K. Maintenance

This is a process of modifying a software system after delivery to correct faults, and to improve performance or other attributes, or adapt to a changed environment. When the system is implemented and deployed at user site then the maintenance phase start. assemble the different kinds of components written in different programming languages. Similar programming language components can be easily integrated and mismatched programming language components can be assembled by using glue code

VI. COMPARATIVE STUDY OF VARIOUS EXISTING LIFE CYCLES WITH THE PROPOSED LIFE CYCLE

TABLE 2. Comparative Study Table

Life Cycle Phases	CBSD Process by Somoerville	V Development Process for CBSD	V Model for CBSD	W Model for CBSD	Proposed Optimized component Development Life cycle
Requirement Analysis and Specification	Outline system requirements	Requirements	System Analysis (Partial) Domain engineering(Partial)	System requirements Domain Knowledge	Domain Engineering Software Analysis and Specification
System & Software Design	Architectural Design	System Design	Design	Component Design	Component Design, Identification and Qualification of components
Implementation And Unit Testing	Identify Candidate components	Select, Adapt and Test	Implementation Testing	Coding Select, Adapt and Deploy	Creating Component repository Optimal Component selection
System Integration	Compose Components to create system	System Integration	Deployment	Deployment Assembly	Component Deployment Composition of components
System Verification and Validation	System Validation	System Test	Testing	Component V & V, composition V & V, system V & V, Acceptance Testing	Component Testing System testing
Operational Support and Maintenance	n/a	Operation and Maintenance	Maintenance	Operation and Maintenance	System Maintenance

VII. CONCLUSION

As described in TABLE 2 the selection of component is supposed to be optimal as proposed. The proposed Optimized Component Development life cycle discusses a solution by catering to an important and common issue for most of the component-based systems which are developed in majority with an objective to be reused in any generic application which is not known to the component developers when they are developing the components. It means when it is not known for which application a particular component is being developed, a greedy optimal selection approach may be used and in case of known user requirements and specific user requirements, the various established signature and specification matching approaches may be used for optimal selection of components. This proposed life cycle follows verification

and validation at both component and system level without using separate life cycles for both.

VIII. FUTURE SCOPE

The components obtained by implementing proposed life cycle can be analyzed against the similar components obtained by implementing other life cycle models to check the worth of the proposed life cycle model. As the proposed life cycle uses greedy approach so it can never guarantee that it is the only method to optimize the development of components.

REFERENCE

[1] Capretz, L. F.,(2005), "Y: A New Component-Based Software Life Cycle Model", Journal of Computer Science, 1(1): 76-82.

[2] Crnkovic, I. , Larsson, S. and Stafford, J., (2002), "Component-Based Software Engineering :Building Systems from Components" , 9th IEEE Conference and Workshops on Engineering of Computer-Based Systems, Lund, Sweden, (April 2002), ACM SIGSOFT Software Engineering Notes, 27(3):47-50.

[3] Crnkovic I., Larsson, S. and Chaudron, M., (2005), "Component-based Development Process and Component Lifecycle", Journal of Computing and Information Technology - CIT 13, 4: 321-327.

[4] Dellarocas, C., (1997), "The SYNTHESIS Environment for Component-Based Software Development", Proc. 8th Int. Workshop on Software Technology and Engineering Practice(STEP‟97), London, UK, (July 14-18, 1997), IEEE Computer Society, ISBN 0-8186-7840-2,Washington, DC, USA, Page 434.

[5] Fahmi, S. A. and Choi, H.J., (2008), "Life Cycles for Component-Based Software Development", Proc. IEEE 8th International Conference on Computer and Information Technology Workshops, IEEE Explore DOI 10.1109/CIT.2008.Workshops.82.

[6] Khan, Ali M. and Mahmood, S., (2010), "Optimal Component Selection for Component-Based System Innovation in Computer Science and Software Engineering", Innovations in Computing Sciences and Software Engineering, DOI 10.1007/978-90-481-9112-3_79, Sobh, Tarek, Elleithy, Khaled(eds.), Springer Science + Business Media , 2010, 19:467-472.

[7] Kwong, C., Tang, J. and Chan, J., (2010), "Optimization of Software Components selection for Component-Based Software System Development", 58(4): 618-624.

(8) Lau, K.K., Taweel, F. M. and Tran, C.M.,(2011), "The W Model for Component-based Software Development", Proc. 37th EUROMICRO Conference on Software Engineering and Advanced Applications, IEEE Explore DOI 10.1109/SEAA.2011.17.

[9] Pande, J., (2012), "On Some Critical Issues in Component Selection in Component-Based Software Development", International Journal of Computer Applications publication, 46(4): 44-50.

[10] Pande, J., (2013), "Optimal Component Selection for Component-Based Software Development using Pliability Metric", ACM SIGSOFT Software Engineering Notes, New York, USA, 38(1): 1-6.

[11] Slyngstad, N.P.O. , Gupta, A., Conradi, R., Mohagheghi, P., Ronneberg, H. , and Landre, F., (2006), "An Empirical Study of Developers Views on Software Reuse in Statoil ASA", Jose Carlos Maldonado and Claes Wohlin (Eds.), Proc. 5th ACM-IEEE Int‟l Symposium on Empirical Software Engineering(ISESE‟06), Rio de Janeiro, IEEE CS Press, ISBN 1-59593-218-6: 242-251.

[12] Sommerville L., "Software Engineering", 7th Ed., Addison Wesley, 2004.

[13] Sutherland J., (1996), "The Object Technology architecture: Business Objects for Corporate Information Systems", OOPSLA'95 Workshop on Business Object Design and Implementation, Springer-Verlag, Berlin. http://link.springer.com/chapter/10.1007%2F978-1-4471-0947-1_3#page-1 (accessed on March 6, 2015).

[14] Veryard, R., (1998), "Making CBD effective in your organization",http://www. users.globalnet.co.uk/~rxv/scipio/SCIPIOrm.PDF (accessed on March 6, 2015).

[15] Vescan, A. and Pop, H., (2008), "Constraint Optimization-Based Component Selection Problem", Studia Universitatis Babes-Bolyai, INFORMATICA, 53(2) : 3-14.

[16] Vinay, Kumar, M., Johri, P., (2014), "W-Shaped Framework for Component Selection and Product Development Process", World Applied Sciences Journal, ISSN 1818-4952, 31(4): 606-614.