Journal of Basic and Applied Engineering Research
p-ISSN: 2350-0077; e-ISSN: 2350-0255; Volume 4, Issue 1; January-March, 2017, pp. 20-23

http://www.krishisanskriti.org/Publication.html

Component based Software Development Lifecycle

Goutam Bhatta

G.L Choudhury College, Barpeta Road, Assam, India
E-mail: bhatta_goutam@yahoo.com

Abstract—*Component based software development has become a major approach in recent years. Component-based Software Engineering (CBSE) deals with the entire life cycle of component based software products. It has been focusing on the technologies which are related to the implementation and design of the software components. Component based Software Engineering approach requires certain changes in life cycle of the development processes. Therefore, a few CBSE works either research field or practical field they are also deals with the Component based software development. This paper describes the differences between the component based and non-component based processes. Therefore some extents summarize the knowledge of areas to be pointed in this report.*

Keywords*: Component based software, Component based Life cycle, Component Development process.*

1. INTRODUCTION

The Component based software development approach has shown considerable successes in many application domains in last few years. Distributed system and web-based systems like desktop and graphical applications are typical examples of domains in which component based software development approach has been very successful implemented. In these domains the general purpose component technologies, like EJB, J2EE, COM, .NET etc. are used.

However a little knowledge about the development processes which is specific for the component based software development. In this paper shows the characteristics of component based life cycle, the reasons for this component based development and the differences between the component based and non-component based development process.

2. COMPONENT-BASED LIFE CYCLE PROCESS MODELS

Component based software engineering (CBSE) shows the challenges similar to those encountered elsewhere in software engineering. Many of the tools and principles of software engineering used in some other system, which will be used the similar way in CBSE. However one main difference of CBSE which specifically focuses the questions related to components. In that sense it distinguishes the component based development from that system development with component.

3. BUILDING SYSTEMS FROM THE COMPONENTS

The main idea of the component based approach is taken from building systems from pre-existing components. It has several consequences for the system lifecycle. At First step of the development processes, the component based systems are separated from the development processes in which the components should already been developed and possibly used in the other products when the system development process begins. Secondly a new separate process will appear that is "Finding and evaluating the components". Thirdly the activities in the processes will be different from the activities in non-component-based approach for the system development in which the emphasis will be finding the proper components and verifying them for the component development and design for reuse will be the main concern.

There is a difference in between the requirement and business ideas in these three cases and different approaches are necessary. The components are built to be used and reused in many applications, some possibly not yet existing, in some possibly unforeseen way. In system development with components is focused on the identification of reusable entities and relationship between them. Beginning from the system requirements and the availability of the components already exist in the system. Implementation effort in the system development will no longer be necessary but the effort required in dealing with components, locating them, selecting those most appropriate, testing them, etc. will increase the component life cycle.

In reality the processes are already separate as many components, which are developed by third parties independently in the system development. Even components are being developed internally in an organization, which uses same components often treated as separate entities developed separately.

I have shown the differences in more detail as follows. Fig. 1 show a V development model adapted to component-based approach.

V model is widely used in many organizations typically large organization building complex for long life of the products such as cars or robots. In this V model the process starts in a usual way by requirements specification followed by system specification. In the non-component based software development approach the process would continue with the unit design, implementation and testing. Instead of performing these activities that are efforts and time consuming, developers simply select appropriate components and integrate them in the system. Two problems appear in V development model which break the simplicity as follows: (i) It is not necessary to select any component and (ii) The selected component only partially fits to the overall design of the product. The first fact shows that we must have a process for finding components. This process includes the activities for finding the components and then the component evaluation. The second fact indicates the need of component adoption and testing before release. Fig. 1 describes the V development process as shown below:

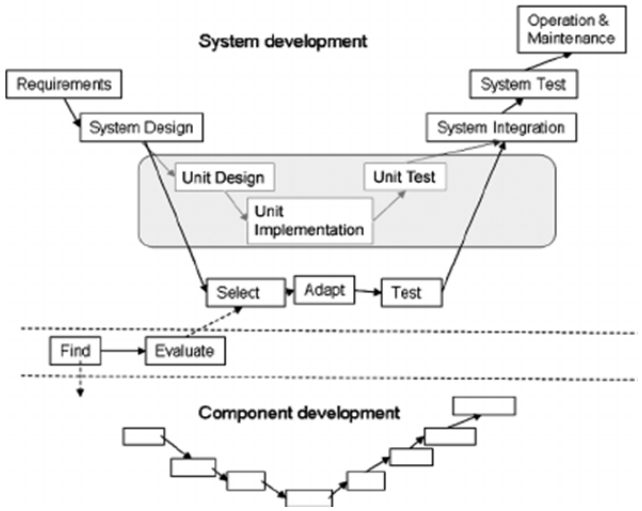

Fig. 1: V development process for CBD

In Fig. 1 shows a simplified and an idealized V development process. It is a supposition that the component selected and used sufficiently close to the units identified in the design process, so that the adaptation process significantly less efforts then the units' implementation. Further it does not deliberate what happens in the maintenance process; if a system malfunctions due to a problem occurred in a component or incompatibilities of the components. It indicates that the component based approach is not only limited to the development process but also the entire life cycle of the development. Let us take a look at Fig. 2 in which the activities at different phases of the development process shown in detail.

4. REQUIREMENT ANALYSIS AND SPECIFICATION

In this phase one of the most important activities is to analyse the possibility of realizing the solutions that will be meet these requirements. In a component based approach, It is necessary to analyse whether the requirements can be fulfilled or not by available components. This means that the requirements engineers must be aware of components which can be used. Some appropriate components can always be found but there is a risk that the new components have to be implemented. To keep component based software development approach one possibility is to negotiate the requirements and modify existing components for reuse.

5. SYSTEM SPECIFICATION AND SOFTWARE DESIGN

System specification and design is strongly related to the availability of the components. The potential components are complying with a particular component based model. Component based model requires a particular architectural framework and the supported applications are use in this framework impact on architectural decisions. For example if the component based model requires client server architecture, it is obvious that the application will use the same. This will put the limitations on the system design. Therefore other properties of components can have a direct influence on the design. So the design process is tightly connected to the availability of the components.

6. IMPLEMENTATION AND UNIT TESTING

When building component based system using glue code an ideal case is to build an application by direct integration of components that is directly connected with components. In programming the "glue code" is a secure code that specifies the connection. In practice the role of the glue code will also include adaptation of the components and even implementation of new functions. In an ideal case the components themselves built and tested. However testing of the components in isolation is not sufficient. The design units will be implemented as assemblies of several components with a glue code. These assemblies must be tested separately and incorrect components themselves are correct.

7. SYSTEM INTEGRATION

The system integration process includes integration of standard infrastructure components that build a component framework and a application components. The integration of a particular component into a system is called a component deployment. In difference to the entire system integration, component deployment is a mechanism for integration of particular components which includes download and registering of the component.

8. SYSTEM VERIFICATION AND VALIDATION

In system verification and validation, standard test and verification techniques are used. The specific problem for component based approach is location error especially when

the components are of black box type and delivered from different vendors. Typically a component can exhibit an error but the cause of the malfunction lies in another component. Some interfaces play an important role in checking the proper input and output from components. These interfaces enable a specification of input and output and checking the correctness of data.

9. OPERATION SUPPORT AND MAINTENANCE

In operation support and maintenance process a new or modified component is deployed into the system and it may be necessary to change the glue code. In some cases an existing component will be modified or a new version of the same component will be integrated into the system. But new problems may be created by incompatibility between components or by broken dependencies. For this reason the system must be verified either formally or by simulation or by testing. In Fig. 2 describe details in V development process for CBD as shown below:

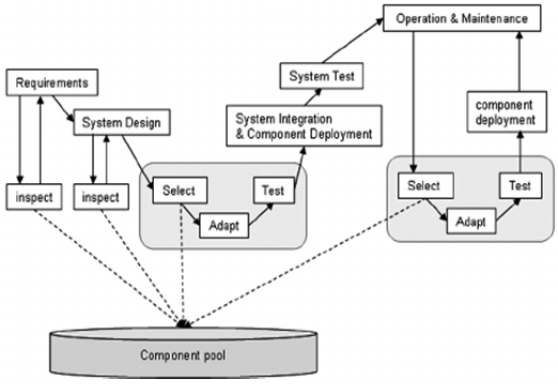

Fig. 2: A Detailed V development process for CBD

In comparison with a non-component based approach, a component based development process there are significantly less efforts in programming but the verification and testing require considerably more efforts.

The verification activity repeats in several phases with slightly different goals which are mention bellow:

- Verifying the component in an isolation,
- Verifying the systems when the component has been deployed into the system,
- Verifying the components in an assembly.

10. BUILDING REUSABLE COMPONENTS

Building components can follow an arbitrary development process model. However, any model will require certain modification to achieve the goals in addition to the demands on the component functionality a component is built to be reused. Reusability includes generality and flexibility. In which requirements may significantly change the component characteristics. For example there might be a requirement for the portability. This requirement could imply a specific implementation solution like choice of programming language, implementation of an intermediate level of services and programming style etc. The generality requirements include more functionality, design, development efforts with the more qualified developers. The component based development will require more efforts for testing and specification of the components. The components should be tested not only in isolation but also in different configurations. Finally the documentation with product delivery will require more efforts, since the extended documentation is very important for increasing the understanding of the component. An example of extended component specification can be found in the "Robocop" component model. Therefore component is specified by a row of modules: functional model, executable model, simulation model and resource model etc. Each model includes its corresponding documentation.

11. CONCLUSION

A component based software development approach cannot be fully utilized if the development processes are not adopted according to basic principles of CBSE. This approach aims for increase the reusability of existing components, decrease the implementations effort and increase the system verification effort. This may requires adjustments of the development processes.

This report pointed out the difficulties to achieve a complete separation of the development processes of systems from the components as well as the need for a project which puts a more important role on the architectural issues and components verification.

REFERENCES

[1] Technical Concepts of Component-Based Software Engineering, Vol -II,CMU/SEI-2000-TR-008.

[2] Szyperski C, Gruntz D, and Murer S, ― Component Software: Beyond Object-Oriented Programming, Addison Wesley, second edition, 2002.

[3] Bass L., Clements P., and Kazman R., Software Architecture in Practice, Addison-Wesley, 1998.

[4] Garlan D., Allen R., and Ockerbloom J., Architectural Mismatch: Why Reuse is so hard, IEEE Software, Vo.12, issue 6, 1995.

[5] Morisio M., Seaman C. B., Parra A. T., Basil V. R., Kraft S. E., and Condon S. E., "Investigating and Improving a COTS-Based Software Development Process", In Proceedings , 22nd ICSE, ACM Press, 2000.

[6] Broy M, Deimel A, Henn J, Koskimies K, Plasil F, Pomberger G, Pree W, Stal M, and Szyperski C, What Characterizes a Software Component?‖ Software—Concepts and Tools, vol. 19, no. 1, pp. 49-56, 1998.

[7] Sommerville I, Software Engineering‖, Chapter 19, 7th edition, 2004.

[8] Lau K K and Wang Z, A Survey of Software Component Models, second ed., School of Computer Science, Univ. of Manchester.

[9] Ivica Crnkovic and Magnus Larsson (editors), Building Reliable Component-Based Software Systems, Artech House Publishers, ISBN 1-58053-327-2, 2003.

[10] ITEA project, ROBOCOP- Robust Open Component Based Software Architecture for Configurable Devices Project http://www.hitech-projects.com/euprojects/robocop.

[11] http://www.cs.man.ac.uk/cspreprints/PrePrints/cspp38.pdf, May 2006.

[12] Allan R , Garlen D, A formal basis of architectural connection, ACM transactions on Software Engineering and Methodology, 6(3) 213-249, 1997.