Component Based Software Development: A new Paradigm

Authors

Nikita Soni, S. K. Jha
Amity Institute of Information and Technology

Email id: 1soninikita9@gmail.com, 2skjha2@amity.edu

Abstract -

Component Based Software Development is the new trend in software engineering that lay down the emphasis on reuse of existing components, thereby, ensuring lower cost and also leads to a flexible development approach by separating the wide ranging functionality in terms of independent components, which can be described as set of related functions or data.

This research paper gives a short introduction of Component Based Software Development methodology and the issues, advantages and limitations that this branch of software engineering possesses. It provides a brief overview of components and component models and highlights some key points about the performance and quality factors of Component Based Software Development.

Keywords - CBSD, components of CBSD, issues, performance, quality assurance and component models.

I. INTRODUCTION

Component Based Software Development is a methodology of software engineering that is based on the concept of reusability by constructing a proposed system with the help of several independent functional sub-systems, or components, so as to lower down the development cost and time and also providing ease in any kind of changes to be done in near future.

Component-based software development is an advancement of object oriented software development (OOD). While both methodologies share the concept of software reusability, OOD is an implementation methodology, whereas Component Based Development is an interface methodology.
Software Engineering is being complex by the intricate requirements that leads to incorporating more functionality in the proposed system. Software Engineering Industry is still in the struggle phase due to the unmanageable, large-scale and complex systems which if not managed properly, can result into an inappropriate and inadequate target system. In order to make the system robust, efficient, cost-effective, and to reduce the market time, Component Based Software Development (CBSD) turns out to be a promising solution. It focuses on improving the maintainability, reliability and overall quality of the system.

While the traditional software development methodologies use the development approach in which a system is brought in existence from scratch, CBSD approach implements the system by integrating the loosely coupled, independent functional units known as Components which can be thought of as set of related functions (or data), thereby leading to shorter development as well as market time.

II. COMPONENTS

A component does not possess any specific definition with respect to this new software development paradigm, but it does possess the following attributes:

- Independent part of system
- Intended for a specific course of action
- Has a well-defined architecture
- Communicates via interfaces, so it acts as an interface methodology, and that’s where it distinguishes itself from the Object Oriented Development which is an implementation methodology

In order to construct a system using components, we need to select the appropriate components from the component repository, and compile them into the target software system.

![Diagram of Component-based software development](image_url)
III. ISSUES OF COMPONENT BASED SOFTWARE DEVELOPMENT

Despite of all the benefits of component approach of software development, there exists several issues like dynamic configuration, scalability etc., which need to be resolved in order to frame the proposed system as per the requirements provided. CBSD is intended to address these issues by providing the appropriate methods, guidelines, and models for the developers of Component Based systems. These methods and guidelines are intended to assist the software engineers in the development of reusable software products and to use an existing system’s reusable components.

The component based software development raises several technical as well as management issues that the software developers must consider while using this approach. Technical issues pertaining to the CBSD approach include:

- Setting up guidelines and methods for reuse concept
- Identifying and defining reusable entities

Management issues for CBSD approach include:

- Setting up training programs for reusable environment.
- Defining the policies, metrics and measurements for reusable framework.

IV. BENEFITS AND LIMITATIONS

The cause of popularity of the CBSD approach lies in the following factors:

- Increase in productivity
- Quality improvement
- Enhanced consistency
- Wider range of usability
- Reduced delivery time and cost
- Reduced market time
- Effective management of complexity

The software development methodology does possess some limitations like:

- Less control over system’s evolution
- Compromise over requirements
- Tradeoffs between usable and reusable components.
- Increased Maintenance cost for components

V. EXAMPLES OF COMPONENT BASED SOFTWARE DEVELOPMENT

Component-based approach is frequently used to develop the domains such as distributed systems, web-based systems, desktop and graphical applications.

A typical example of a system using component based software development methodology is of a reservation system having several distinct components like billing, hotel reservation, air reservation, car reservation, loyalty programs, session management etc. Every component plays a distinct role as per its functionality, and all components integrate to form a system as a whole. If any change is needed to be performed in the system, modification is made only in the desired components.

Figure: Reservation system developed using CBSD approach

VI. COMPONENT PERFORMANCE CRITERIA

The performance of components depends on implementation as well as on the context of deployment. The factors that influence component performance are:

Figure 1: Factors influencing Component Performance
• **Component Implementation:** Functionality defined by an interface can be implemented in different ways.

• **Required Services:** The execution time of a component service always depends on the execution time of component services that it requires while its execution.

• **Deployment Platform:** A deployment platform consists of several software layers. Performance also depends on type of deployment platform provided by the software architect.

• **Usage Profile:** Execution time depends on the value of input parameters with which the component services are invoked.

• **Resource Contention:** A component service does not execute in isolation on a platform, but the waiting time while accessing limited resources adds to the execution time.

### VII. QUALITY ASSURANCE

The components and overall system following the component based software development approach needs a model for quality assurance. Many guidelines and standards exist to control the quality activities of the systems developed via CBSD approach. The *ISO9001* and *CMM* models are one of the quality assurance schemes.

The component and system undergo various quality checks while performing the quality assurance testing. The phases in quality assurance models are:

- Component requirement analysis
- Component development
- Component certification
- Component customization
- System architecture design
- System integration
- System testing
- System maintenance

### VIII. COMPONENT MODELS

Component models define the standard forms and standard interfaces between the components. It maintains the quality and integrity of the components by forcing them to fall under certain standards and allow the
components to effectively communicate with each other. Three majorly used component models are COM, JAVA (EJB) and CORBA.

COM is an open standard, although is Microsoft Windows platform oriented, but can be implemented on other different platforms too. Enterprise JavaBeans or EJB is a distributed system oriented architecture intended for the sever-side components and thereby serving multiple clients and servers. Common Object Request Broker Architecture or CORBA is a part of OMA (object Management Architecture) which bridges the common facilities, application objects and object services via object request broker layer. It was brought into existence by OMG (Object Management Group).

**IX. CONCLUSION**

This article introduced briefly the research results carried out by various researchers regarding the basics of component based software development approach. It has surveyed about the components, issues, examples and benefits and limitations of component based software development. It has also highlighted the component models, performance measure of components and quality assurance models to be followed while developing a system using CBSD approach.

**REFERENCES**

[1] Component-Based Software Engineering – New Paradigm of Software Development Ivica Crnkovic, Magnus Larsson