International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622
NATIONAL CONFERENCE on Developments, Advances & Trends in Engineering Sciences
(NCDATES- 09th & 10th January 2015)

RESEARCH ARTICLE OPEN ACCESS

Component Based Software Engineering Systems

Prof.V.Anandam
HOD, Department of CSE & IT, Vardhaman College of Engineering, Hyderabad

Abstract
Component based software engineering is one of the major advancement in the field of software engineering. It is a process that emphasizes the design and construction of computer based systems using reusable software components. It provides the methodology of developing very large software systems. It supports both the Commercial-off-the-shelf and in-house components. This paper discusses the component based software engineering fundamentals. It also highlights the process involved. This paper surveys the various current metrics for component based software engineering systems namely for complexity, cost, etc. in detail. The metrics help in enhancing the quality and risk management in the component based system.
Keywords: Component based software engineering, Reusability, Complexity, Metrics.

INTRODUCTION

In early days, software engineering approach was ad hoc. Around 1970s, introduction of structured programming‖ gave a formal shift in software engineering from the ad hoc to a systematic approach. Then around 1980s, introduction of object oriented programming with some advancement explores new areas in software engineering. In recent dates, with the introduction of Component Based Software Development (CBSD), the industry is moving in a new direction. The basic insight is that most software systems are not new. Rather, they are variants of systems that have already been built. This insight can be leveraged to improve the quality and productivity of the software production process [1]. These day's software systems are more complex as compared to those of early. These complex, high quality software systems are built efficiently using component based approach in a shorter time. Component based systems are easier to assemble and therefore less costly to build than developing such systems from scratch. The importance of component based development lies in its efficiency. In addition, CBSE encourages the use of predictable architectural patterns and standard software infrastructure, thereby leading to a higher result. In the remaining part of this paper the term *"component"* and *"software components" will be used interchangeably.*

In 1968, Douglas McIlroy's [2] first share the idea of Component Based Software Development (CBSD) at the NATO conference on software engineering in Garmisch, Germany in his paper titled ―Mass Produced Software Components‖. He discussed the idea that, software may be componentized i.e. built from pre-developed software components. His subsequent inclusion of pipes and filters into the Unix operating system was the first implementation of an infrastructure for this idea. Later, Brad Cox set out to create an infrastructure and market for these components by inventing the Objective-C programming language. IBM led the path with their System Object Model (SOM) in the early 1990s. Some claim that Microsoft paved the way for actual deployment of component software with OLE and COM. As of 2010 many successful software component models do exist.

Component based software engineering is widely accepted as a new and latest approach to software development. Nowadays the software systems are highly complex, large and uncontrollable. These result in lesser productivity, higher risk management and greater software quality. So, there arises the need for component based systems. These systems follow the principle of component based software engineering. The Component based software engineering is a process that emphasizes the design and construction of computer based systems using reusable software "components". [1] It changes the methodology of developing very large software systems. It defines the components that are used in the software systems. It comprises of the system of components.

A component is a non-trivial, nearly independent and replaceable part of a system that fulfills a clear function in the context of a well-defined architecture. It offers the following features:

- Independent and replaceable part of the system
- Performs a particular function
- Works on a well-defined architecture
- Communicates with the help of interfaces.

A software component specifies a unit of composition with contractively specified and explicit contrast dependencies only. It embodies the principle

of "buy, don't build dependency". Here we follow the integration policy of the components. Thus, component based software engineering is a process of integrating the various software components to form an application to satisfy a functionality.

This approach is different and easy going from the other traditional approaches. The components can be developed in different languages and on different platforms which are further integrated to form a particular software system. The Component based approach can be elaborated using the following diagram:

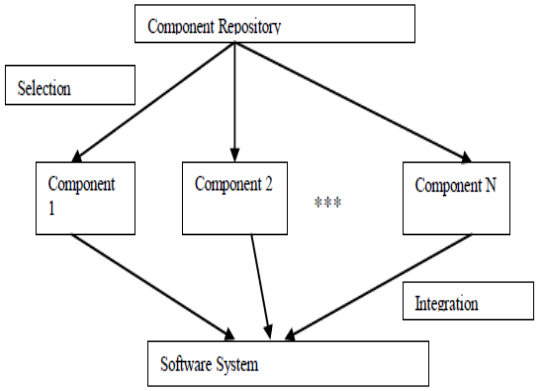

Figure1. Component based software development

A software component provides a package of the modules of the software that are independently developed and delivered integrating to form an application on the system. [2] It provides us with selection of the off shelf components and then assembling them into a useful architecture model. The components play a trivial role in the formation of a software system. Thus, [3] CBSE offers a set of prebuilt, standardized software components making them available to fit a specific architectural style for an application domain. Some of the advantages of component based software systems over the traditional systems are:

- Less time consuming
- Components can be integrated with ease
- Easy to assemble
- Lesser cost incurred in building
- Lesser time-to-market
- Follows predictable architectural patterns
- Provides higher quality results
- Increases reusability
- Increases maintainability

Thus, component based software engineering offers huge advantages or benefits and one of the promising systems in the today's world. The CBSE process is elaborated in the second section. And the metrics involved are discussed in section third in detail.

I COMPONENT BASED SOFTWARE ENGINEEING PROCESS

The component based software engineering follows the two parallel engineering activities. These activities perform the main task of evaluating the system.

A. Domain Engineering

Domain Engineering explores the application domain with the specific intent of finding behavioral, functional and data components that are the candidates for reuse [4]. Its main task is to identify, construct, store and distribute a set of components to provide meaningful applications. Its goal is to share the components making them reusable in the application domain. It provides three major activities:

i. Domain Analysis Phase
ii. Domain Construction Phase
iii. Domain Dissemination Phase

These phases play an import part in the finding of a particular domain. [5] In the analysis phase, we identify the domain and the items in the domain. Then these are analyzed by the system creating an analysis model. The parameters to be considered are functions or objects, taxonomy, features, domain languages, relationships and functional model. In the construction phase, it checks whether some generic attributes of the system exist in the domain. The model is constructed on the basis of the values specified in the domain of the system.

Here the characterization of the components is performed in the system that is., whether they are reusable or not in the system or whether they are relevant and what basis. In the last phase, the dissemination of data is performed on the basis of the structural modeling and structural points. The structural modeling is a pattern based approach that is used for modeling the data in the system as a whole. It shows that the repeating patterns have of data or function had a large amount of reuse potential in them. These highlight the correct amount of interaction among the systems as a whole.

The structure point is an abstraction that has a limited number of instances in the system. These are governed by rules of the interaction process and implement the principle of information hiding by isolating the complexity of the systems. They act as an alternative to the lines of code and the function points in the metrics used. They help in the generation of the domain model and the structural model using the data stored in the repository. Thus, domain engineering identifies the data, functional and behavioral components of the system and in turn

providing reusability to the system.

B. Component Based Development

It obtains requirement from the customers and selects an appropriate architectural style to be built. It helps in the selection of particular potential components for reuse. It also adapts the components so that they can properly integrate to form a proper application. It integrates the components to form a particular subsystem. This activity occurs parallel to the domain engineering phases. The data is transferred to and obtained from the domain engineering phase also. The component based development involves two approaches. First, if there are already existing components they need to be integrated in the system. Secondly, if the components do not existing then they need to be re-engineered in the system. The components based approach has 4 main activities:

i. Component Qualification
ii. Component Adaptation
iii. Component Composition
iv. Component Update

Reusable components namely commercial-of-the-shelf or in-house components are used which are derived from domain engineering. In the Component Qualification, the module of the component will perform the required function and properly in the correct architectural style sowing the characteristics of reliability, reusability, quality and performance of the application system. The interface provides useful information about the working and usage of the system. The composition qualification depends on the on application programming interface (API), development and integration tools, exception handling, requirements, security features, embedded designs etc. In Component adaptation, the system specifies particular task. Easy and proper integration can be achieved by consistent methods of resource and data management and integration of interfaces are used. In case or any incoordination we follow the principle of component wrapping in the system. It can be of two types namely white-box wrapping and gray-box wrapping.

In Component composition, the components are qualified, adapted, integrated, used and engineered to form the architecture for the system. It provides the coordination among the components of the system. Coordination acts as an important basis for the proper working of an application. We follow a set of architectural constraints in the system. Data exchange model is used to provide user and application to interact and exchange the data and system resources among the various reusable components. Automation provides tools and macros for the proper interaction of the components. Structured storage provides us the facility of storing, accessing and organizing heterogeneous data in a structured form. It maintains the structured index of the data. Underlying object model is used to specify that the components developed in different programming languages on different platforms support interoperability. It plays a major role in enhancing the quality of the system. [6] OMG/CORBA, Microsoft COM, Sun JavaBean component supports interoperability. In Component Engineering, the software components are specified in the system and used for resource allocation in the system. The analysis and design of components help in the proper reuse of the system. Automated tools help in the selection of the repository tools of the system. The CBSE supports a large number of automated tools. Thus, this is the process of component based software engineering.

METRICS FOR COMPONENT BASED SOFTWARE ENGINEERING

It helps in providing the data to the system and increasing the quality of the system. It is largely helpful in the case of the business systems for retrieving huge data. System requires higher quality and do not afford any risks in the system. This could be achieved with the help of the software metrics in the system. These metrics are helpful in guiding the quality and risk management in the component based system by checking the factors that influence risk and quality. [7] The metrics play an important role in guiding the software development and deployment models. The metrics play an important role in highlighting the system. Metrics help developer identify the probable risks and take the proper corrective action. The metrics are surveyed in detail below:

A. Metrics for quality, productivity and cost

We verify that the system components used are reusable and correct having no defects. As the systems are analyzed and tested the defects are detected and corrected. We analyze the quality using KLOC (Kilo lines of code) and LOC (lines of code). These evaluate the instructions and generate the system. These identify the percentage of the errors or defects of the system. For the cost, the net savings (C_d) are estimated from the scratch of the components (C_s) in the system and the components of the reuse (C_r).

$$C_d = C_s - C_r$$

Thus, the cost incurred can be calculated in the system. This plays an important role in the system. The cost on the analysis model can be easily

analyzed in the system. We can also perform by using the effort on the system. To estimate the effort we use

Overall effort=$E_{new} + E_{qual} + E_{adapt} + E_{int}$

Where,
E new = effort needed to engineer new components
E qual = efforts needed to qualify structure efforts
E adapt = efforts needed to qualify adapt structure points
E int = efforts needed to integrate structure points

The effort is estimated using various structure points. Structure point plays an important role in structural modeling of the system. It is highly helpful in creation of the domain of the system. Thus, effort also amounts to the load.

B. Reuse Metrics

The metrics in the system are used for finding reusability of the system. It can be explained in the form of ratio:

$$R\ b(S) = [C\ noreuse - C\ reuse] / C\ noreuse$$

Where,
$C_{noreuse}$: is the cost of developing S with no reuse
C_{reuse}: is the cost of developing S with reuse

The value of $R_b(S)$ is a multidimensional value where its value lies between 0 and 1. Another metrics defined is the reuse leverage metrics used in the object oriented systems.

C. Quality Metrics

The metrics which take into consideration the quality as its prima facie are quality metrics. They are of following types:

- Adaptability: it provides the system with the ability to adapt to the changes in the network.
- Integration Test Coverage: It evaluates the fraction of the system which undergoes integration testing.
- Complexity of interface and integration: here the complexity of the interface and integration code is determined.
- End-to-end test coverage: The system undergoes satisfactory E2E testing.

Fault Profiles: It detects the number of cumulative faults

- Reliability: calculates the probability of the fault free system.
- Customer satisatisfaction: It calculates the degree of customer satisfaction towards software.

These metrics help in enhancing the quality of the particular system.

D. Management Metrics

These metrics define the amount the amount of management on the particular software components. These metrics pertains to the cost of the particular system [9]. An influence diagram shows the metrics in the particular order in the system. The metrics used for it are:

- Cost: calculates the expenditure incurred.
- Time-to-market: Time spent between the start of the system development till it is deployed to the users or customers as an application.
- Software Engineering Environment: It measures the capabilities of the software.
- Software requirement utilization: It measures the reusability of the software components.

These metrics play an important role in determining the cost effective component based systems.

E. Requirement metrics

The requirement metrics depend on the process to product system. It derives out the metrics components needed depending on the requirements supplied. These are:

- Requirement Stability: It amounts to the level of changes in the software systems checking whether the system ate stable in provided conditions.
- Requirement Conformance: It portrays the compliance of the integrated component forming a particular system.

F. Complexity metrics

Complexity defines the inter relationships in the system between the components. These metrics measure the complexity of a component. A high value for the metrics RFC, CBO, DIT, CPD, CIID, COID, and CPC, imply that the corresponding component.[8] These depend on the size of the system and the lines of code of the system. These depend on the number of functions or components in the system. Some of the components of the complexity metrics are:

i. Cyclomatic Complexity
ii. Number of Logical Operators
iii. Essential Cyclomatic Complexity
iv. Myer.s Interval
v. Maximum Nesting of Control Structures
vi. Estimated Static Path Count

Thus, the metrics play an important role in determining the benefits and aspects of the component based software engineering systems

CONCLUSION

The Component based software engineering is one of the promising fields of software engineering. The components are specified in the system. A component is a trivial component is a non-trivial, nearly independent and replaceable part of a system that fulfills a clear function in the context of a well-defined architecture. It is communicated using the interfaces in the system. The components are used to increase the reusability in the system as a whole. Thus, components located on different platforms in different languages are integrated in a fruitful application of the system. The CBSE process is a parallel engineering process involving domain engineering and component based systems. These generate the domain and structural environment with the help of structural points. The Component based system involves the construction, coordination, adaptation and reliability of the system. The metrics provide us a measure regarding the various parameters pertaining to a successful product in the market. This approach can be implemented using java net beans environment or Microsoft COM. The metrics deal with the following factors like complexity, size, reliability, usability, maintainability and understandability. The metrics also focuses on the production, cost and reuse of the system finally disclosing the management and the requirement metrics. This approach of the metrics can lead to development of more metrics including a large number of components in the system. In future more work could be done towards the reusability metrics in the components.

REFERENCES

[1] V. Lakshmi Narasimhan, P. T. Parthasarathy, and M. Das, "Evaluation of a Suite of Metrics for Component Based Software Engineering (CBSE)", Issues in Informing Science and Information Technology Volume 6, 2009

[2] Xia Cai, Michael R. Lyu, Kam-Fai Wong Roy Ko, "Component-Based Software Engineering: Technologies, Development Frameworks, and Quality Assurance Schemes"

[3] Sahra Sedigh-Ali, Arif Ghafoor and raymond A. Paul, "Metrics-Guided Quality Management for Coomponent-Based software Systems", 0-7695-1372-7/01 $10.00 @ 2001 IEEE

[4] Abhikriti Narwal, "EMPIRICAL EVALUATION OF METRICS FOR COMPONENT BASED SOFTWARE SYSTEMS", International Journal of Latest Research in Science and Technology ISSN (Online):2278-5299, Vol.1, Issue 4: Page No.373-378, November-December (2012)

[5] Roger S. Pressman, "Software Engineering- A Practitioners approach".

[6] M. Nadeem, M. R. Asim and M. R. J. Qureshi, "A STEP FORWARD TO COMPONENT-BASED SOFTWARE COST ESTIMATION IN OBJECT-ORIENTED ENVIRONMENT" , Pakistan Journal of Science (Vol. 62 No. 4 December, 2010)

[7] VINIT KUMAR, "Component Based Effort Estimation During Software Development: Problematic View", IJCSMS International Journal of Computer Science and Management Studies, Vol. 11, Issue 03, Oct 2011

[8] Antonia Bertolino and Raffaela Mirandola, "Towards Component-Based Software Performance Engineering".

[9] P. Edith Linda, V. Manju Bashini, S. Gomathi, "Metrics for Component Based Measurement Tools", International Journal of Scientific & Engineering Research Volume 2, Issue 5, May-2011.