ISSN No. 0976-5697

Volume 7, No. 6(Special Issue), November 2016

International Journal of Advanced Research in Computer Science

REVIEW ARTICLE

Available Online at www.ijarcs.info

A Review on Major Issues in Software Component Selection

Nitish Madaan
Student
ITM University, Gurgaon
nitishmadaan@gmail.com

Rekha Kalra
Assistant Professor
Guru Nanak College, Budhlada
rekha_nskalra@yahoo.co.in

Nikita Madaan
Student
Thapar University, Patiala
nikitamadaan1@gmail.com

Abstract- Component based software development(CBD) , as defined by name to develop a software by selecting appropriate components and then assemble all the components in a well-defined manner of software architecture. The selection of components becomes more complex day by day to meet the expected requirements. In this paper a model is represented to define the current needs of an application. This model confirms expected requirements for the component selection we also did survey upon the current CBD technologies, their advantages and disadvantages and also about the characteristics they inherit. Research and challenges of Component based software systems are also discussed.

***Keywords** : CBD,CBSE, CBSD,OS,PC.*

1. INTRODUCTION

Software systems day by day providing more functionality and due to that system are becoming more complex. Component Based Software development provides ease to the software developer to develop the software by selection pre build software components instead from scratch. It also helps to reduce the cost of the software development and also the cost[1]. Due to the saving of time and cost it is possible for the developer to produce more functionality with the saved time and cost. CBD is still associated with so many problems which remain unsolved. Likewise configuration Management was introduced as a solution with the help of values and methods from former engineering disciplines.

In modern software systems the problem is faced as it becomes more large, uneasily controlled and complex. So, as a result cost of development is high, productivity is low, quality of software is unmanageable and risk is very high to move to the new technology. All these are solved today with the help of component based software development (CBD)[3] . According to the experiences, due to different requirements and constraints of system, so many difficulties occurred during the use of same component technology in different domains. Components are available as commercial off-the shelf components (COTS) and it is developed using different languages and platforms by different developers. In the given figure 1, the commercial off-the shelf components can be checked out from a repository of components and then combined into a target software system.

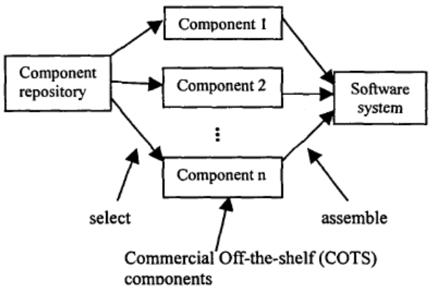

Figure 1. Component-based software development

The software development model in CBSD is much different from the traditional ones. In this new area of development there are no existing standards or guidelines, and we are unable to define a key item "Component". [2]In selection of software component has mainly these three features:

1. Working of a component is under a well defined architecture
2. Communication between components is done by its interfaces;
3. A component is having a clear functionality, so that it must be independent and replaceable part of a system.

2. A HISTORY OF COMPONENT-BASED DEVELOPMENT:

[8]The component based software development (CBD) is close connected to reuse. In early sixties, when software crisis was mentioned first time , at that time the idea of reuse of software pieces originates. The basic simple idea behind this is that a

978-93-85670-72-5 © 2016 (RTCSIT)

developer develops software in a way that other can use this. In the software system it looks like it uses the simple principle, but during the time of implementation it is relatively hard. The method of getting better the reuse has been long and laborious. [4] We have an example of mathematical libraries, in which it was the successful reuse in the development of different libraries. so many functions are included in these libraries (e.g. functions of mathematics like sine , cosine, matrix operations ,etc.), all these are referred to in the source code and then connect together with the proprietary code. By using this type of reusable entities gives success in several facts:

1. These types of functions accomplished with well defined theory.

2. The communication between these functions and the applications is simple. 3 the inputs and outputs for these functions are accurately defined.

Let us take the example of Operating System (OS), Operating Systems are typical reusable "components". [5]There was a problem occurred in the past with OS that OSs were costlier and big , So at that time system had to include so many parts that are required but never have been used which may lead to increased the costs. for these reasons companies decided to develop their own OSs which were adjusted to needs of the system. Figure 2 shows the example of system evolution:

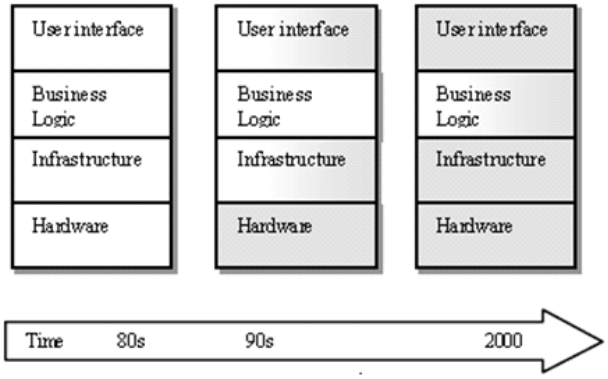

Fig. 2 . System reusability evolution

In 1980s the systems were developed from the scratch , which includes the development of the hardware , the basic development of the software, such as OSs and even development environment is also need to be developed like debuggers, compilers etc.

In 1990s, that it was possible to buy hardware due to its standardization (e.g. PCs). The general purpose OSs has been used more as they become cheaper.

Nowadays, typically, standard user interface is used as well as standard infrastructure and standard hardware. Only those parts of the system are developed internally which are directly related to end user. Some of the important consequences in this evolution the development time and development cost is decreased very extensively.

3. SELECTION ISSUES IN SOFTWARE COMPONENTS

Software components selection is major task from the subset of components. In CBD very important role is played by component selection. End user may get quality software if developer and researcher keep all the requirement and challenges in mind at the time of the component selection.

3.1 Components selection factors

[6]There are several component selection factors like fault tolerance, components compatibility, component size, Reusability of Component, performance, time etc.

a) Fault Tolerance

Increasing of fault tolerance is basically due to Increase in Mean Time to Failure (MTTF). Enabling a system to operate properly in the mean time of failure of some of its components is to the property of fault tolerance.

b) Components Compatibility

The main challenge in the reusability of component is to check the compatibility between the several versions of that component. If a new component is compatible with its previous version than it is easy to replace with the previous one. It is mandatory to do the test plans like regression test, functional compatibility etc.

c) Size of Component

User of the software always wants that the software should be of less in size. Component size completely depends on the languages programming languages that can be used for the development. SO to achieve this it may be notice that only high level programming languages are used because it needs lesser size.

d) Reusability of Component

The main challenge to select a component of good quality, because degree of a component must be checked for its reuse.

CBD helps to enhance the reusability from the pre built software components and it also saves the time and cost of development.

e) Performance

Performance is the main challenge at the time of selection of component. Performance is not calculated for a single component; rather it is calculated for the whole system after integration. To increase the performance of the system select only those components of software which contains high cohesion modules and less coupling of module, and also less number of interfaces of components.

f) Time

Development time and testing time saves when we use COTS components and it also improves the quality of our software.

3.2 Common Steps of Component Selection Methods

[7]Although there is no commonly accepted method for component selection, all methods share some key steps that can be iterative and overlapping. These steps are described as follows:

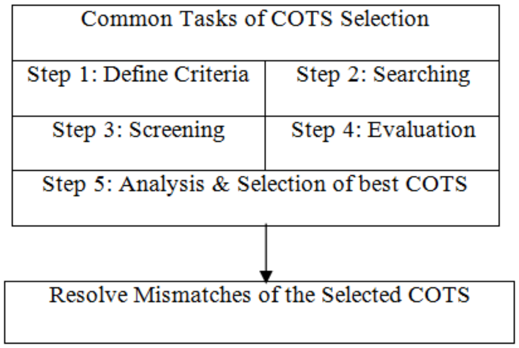
Common Tasks of COTS Selection
Step 1: Define Criteria
Step 2: Searching
Step 3: Screening
Step 4: Evaluation
Step 5: Analysis & Selection of best COTS
Resolve Mismatches of the Selected COTS

Figure 3 The general COTS selection process

Step1: Define the evaluation criteria
Step2: Search for components.
Step3: Filter the search results based on a set of requirements. This result defines a short list of most promising component.
Step4: Evaluate components
Step5: Analyze and select the best component

4. CONCLUSION AND FUTURE SCOPE

In this paper, we review about the issues of software component selection based on several dimensions like cost, quality, performance etc. During software development the developer analyze all challenges to select the best optimal components from the reusable component repository. In future work we analyze that development of component based software engineering needs more quality metrics for the software components that are easy to compute and more practical to use. Thus, it would be suggested to apply trapezoidal, bell-shaped, triangular or other possible distributed patterns for representing imprecise numbers in solving COTS selection model using the fuzzy logic approach.

REFERENCES

[1] Gill, N. S. and Tomar, P. (2007). “Software Component Technology: An Easy Way to Enhance Software Reusability”, proceedings of 94th Indian Science Congress Conference, Annamalai University, Tamilnadu, INDIA, pp. 18-19.

[2] G. Ruhe, "Intelligent Support for Selection of COTS Products,", LNCS, Springer, vol. 2593 2003. pp. 34-45

[3] Mohamed, Abdallah Sami Abbas Shehata. Decision support for selecting COTS software products based on comprehensive mismatch handling. Diss. University of Calgary, 2007.

[4] Pande, Jeetendra, Christopher J. Garcia, and Durgesh Pant. "Optimal component selection for component based software development using pliability metric." ACM SIGSOFT Software Engineering Notes 38.1 (2013): 1-6.

[5] Kaur, A. and Mann, K.S. 2010. Component selection for component based software engineering. International Journal of Computer Applications, 2, 1(2010),109–114.

[6] Wes J. Lloyd “*A Common Criteria Based Approach for COTS Component Selection”*, in *Journal of Object Technology*, April 2005.

[7] Syed Ahsan Fahmi, Ho-Jin Choi *“A Study on Software Component Selection Methods”* Feb. 15-18, 2009 ICACT 2009.

[8] Arvinder Kaur, Kulvinder Singh Mann, “*Component* Selection for Component based Software Engineering, International Journal of Computer Appl ications”(0975 – 8887) Volume 2 – No.1, May 2010.