CHAPTER 5:- QUALITY SOFTWARE
CBSE is a latest technology which is used to increase the reusability because reusability is a way to improve productivity and efficiency of software systems. Researcher can say it is used to develop a mechanism which helps in identification of software components and to reuse them for CBSD. CBSE is based on three pillars.

1) Component Selection

2) Cost and Time

3) Component-Based software quality.

It is a big challenge to make CBSE three pillars strong, but if we apply some algorithm’s to select an optimal component according to customer requirement from the repository then the first pillar would be strong, and we talk about the second pillar then there should be a condition that – as we know that Reusability means the reuse of code with modification or without modification as per according to the need of customer, so when the code will modified then we will see the customer requirement s again then the design would be prepare again and as like it the code will also written again so if after coding phase we apply the a method which can find the fault in program before testing then there will be a positive effect on cost and time and know when we talk about the last pillar means to ensuring our product on the bases of quality with the help of software quality then we have to follow a software quality model and if we are using ISO-9126 software quality model, then firstly we have to add an attribute with the six main attribute of ISO-9126 because without this seventh attribute it is not sufficient to justify or proof the quality of Component-Based Software, because it is totally reusability of code.

5.1 Three Pillars-

5.1.1 Algorithm for Optimal Selection of Component

CBSE is a mechanism in which the integration of Domain Engineering and CBSD, both run in a parallel manner, so if we want to get a quality product with CBSE then we have to maintain our process from initial step and for that we have to apply an algorithm to select optimal component from the repository which would be comes after the domain engineering and after getting a optimal software components set
from the repository according to the customer need, we will compose them to make a product.
CBSE process has eight basic steps.

1) Domain Engineering
2) Software Analysis and Specification
3) Making Component repository
4) Optimal Component selection Algorithm
5) Composition of Component
6) System Testing
7) Deployment
8) Maintenance

Researcher can make our product according to the customer requirement and the quality of product should be good.

5.1.2 Method to predict the fault before the testing phase
Halstead’s software can identify fault with reusability for CBS before the testing phase. We are using Halsted Software Science as a base for quality of component based software. According to Halstead model defined the following equation 1 and 2.

\[
\text{Volume (V)} = N \log_2 (n_1 + n_2)...................... 1
\]

\[
\text{Fault B} = \frac{V}{S_0}................................. 2
\]

Now to find the faults and the factors which affect the reusability we will use Halsted Software Science equations about fault and volume with addition of a variable for the reusability affecting factors.

\[
S_{\text{REU}} = S_{\text{RC}} \cdot S_{\text{DC}} \cdot C_{\text{CC}} / C_{\text{C}} \cdot S_{\text{C-REU}}............. 3
\]
$S_{C\text{-REU}}$ is software complexity for analysis of reusability factor at different levels $C_{CC}$ is \textit{Code Change} in Component

$S_{DC}$ is \textit{Design Change} in software,

$S_{RC}$ is \textit{Requirement Change} in software,

To calculate the above variables for $C_C$ is \textit{Component Complexity} the factors which have positive effect taken as numerator and the factors which has negative effect taken as denominator to make a variable. And then put this variable in equation 2, through which we can get the maximum number of faults occurred before the testing and also get the value of factors which effect the reusability.

<table>
<thead>
<tr>
<th>S.NO</th>
<th>General Factors for analyzing Reusability of CBS</th>
<th>Impacts on Reusability</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Requirements</td>
<td>Change</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No Change</td>
</tr>
<tr>
<td>2.</td>
<td>Design</td>
<td>Change</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No Change</td>
</tr>
<tr>
<td>3.</td>
<td>Code</td>
<td>Change</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No Change</td>
</tr>
<tr>
<td>4.</td>
<td>Component Complexity</td>
<td>Increase</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Decrease</td>
</tr>
<tr>
<td>5.</td>
<td>Software Complexity</td>
<td>Increase</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Decrease</td>
</tr>
</tbody>
</table>

\textit{Figure 5.1 Impacts on Reusability}
After putting the value of equation 3 in 2 we get-

\[ B = S_{REU} \times \frac{V}{S_0} \]

This method is very helpful in reducing the risk and cost in our component based development process to remove the obstacle before software testing phase according to the Halsted software science.

### 5.1.3 Quality Component–Based Software

When we talk about the quality of component-based software then we will use some quality models to satisfy it by satisfied the different conditions of different attributes of that software quality model. Like when we talk about the software quality model ISO-9126 then we will find that it not has an important attribute according to Quality Component-Based Software, and that attribute is Reusability because CBSE is based on the concept of reusability. We are using this model because ISO 9126-1 is an extension of previous work done by McCall (1977), Boehm (1978), FURPS and others in defining a set of software quality characteristics. The ISO 9126-1 software quality model identifies **6 main quality characteristics**:

- Efficiency
- Portability
- Functionality
- Maintainability
- Reliability
- Usability
- Reusability
But when we are talking about the CBSE then we have to use one more attribute to use the ISO - 9126 as a quality model for the Component–Based Software Engineering or for CBSE process to produce a quality product. Because the concept of CBSE is totally depend upon the reuse of code.

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>Factors for Analyzing Reusability of Software</th>
<th>Impacts on Reusability</th>
<th>Factors for Analyzing Reliability of Software</th>
<th>Impacts on Reliability</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Requirements</td>
<td>Change</td>
<td>Requirements</td>
<td>Change</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No Change</td>
<td></td>
<td>No Change</td>
</tr>
<tr>
<td>2</td>
<td>Design</td>
<td>Change</td>
<td>Design</td>
<td>Change</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No Change</td>
<td></td>
<td>No Change</td>
</tr>
<tr>
<td>3</td>
<td>Code</td>
<td>Change</td>
<td>Code</td>
<td>Change</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No Change</td>
<td></td>
<td>No Change</td>
</tr>
<tr>
<td>4</td>
<td>Component Complexity</td>
<td>Increase</td>
<td>Component Complexity</td>
<td>Increase</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Decrease</td>
<td></td>
<td>Decrease</td>
</tr>
<tr>
<td>5</td>
<td>Software Complexity</td>
<td>Increase</td>
<td>Software Complexity</td>
<td>Increase</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Decrease</td>
<td></td>
<td>Decrease</td>
</tr>
</tbody>
</table>

*Table 5.1: Different Factors to Measure Reliability and Reusability*
So we can say that without the reusability attribute the ISO 9126 is not able applicable for Component-Based Software Engineering. Now the ISO-9126 model would be a contribution of seven main attributes and there sub attributes like as shown in figure 5.2.

The different factors have different sub factors with their different functionality parameters. We are introducing reusability as a seventh factor with its sub factors and the sub factors are-

1) Coupling
2) Cohesion
3) Flexibility
4) Compliance