Topic: High level requirements for an automating tool to test the reusability of a software code

Let’s discuss what we mean by the reusability of the software code. It refers to the process of using set of code from a software or an application in other application or software development process to reduce time consumption and also to reduce usage of resources. Though it sounds easy but there are lot of constraints present in reusing the code. Let us see what are they?

1. First of all, the functional requirements between these two systems where the reusing is about to be established should be the same.
2. Though we reuse the code, this has to be tested under various conditions to ensure that the quality and the correctness remains unchanged and it satisfies the need of the customer.
3. If the code is reused and if there are any changes in the future for that software, like adding new feature or expanding the domain of the software it shouldn’t cause any conflict.
4. If we use more code for reuse then it’s basically a copy of another application. Hence the amount of code to be reused must be constrained.
5. We should also consider all the cost that it requires to reuse the code and it shouldn’t exceed the value of the time we save. And also consider the amount of error which we would be facing when we implement old reusable code into a new version

The steps that a reusable code should follow are below:

1. Consistency: The application or the software should be consistent when we try to implement a used code and it shouldn’t be erratic when performing operations.
2. Extensible: The code should support additional features which might be added in the future and it shouldn’t create any conflict. If this requirement is not satisfied then the whole software might get stuck at one point of time and remain stagnant. It will cost a fortune to revert everything to initial position and start all over again.
3. Generic: The main point of reusing the code is that it is generic enough to be implemented. The more common way the code works then more use it has for other software.
4. Flexible: The code should be flexible enough to handle some changes. Generally the reusability of the code doesn’t meant they would be using the code as it is without any changes. Sometimes a small tweak or repair might be needed so it can fit inside the software.
5. Complexity: The complexity of the code should be reduced as much as possible. Complex code can be difficult to make changes to and also it is more prone to give error when there is some critical piece is changed.
6. Adaptability: The code should be adaptable to different platforms and devices. More or less if the code is platform independent then it would be even better. This is where sometimes the test case fails. When the code is tested in an organization usually they all might have the same kind of device or platform but the software should be developed in the client or the customer perspective.