Model-based testing (MBT) means using models for describing test environments and test strategies, generating test cases, test execution and test design quality. With MBT there is the possibility to trace correspondence between requirements, models, code, and test cases used for testing the system.

In model-based testing the input data or test cases can be generated from a domain model, environment model, oracles or from abstract test. Manual testing is one of the testing processes used by various industries. Test designs are done manually and are based on requirement documents. The document is writing in a way that they are readable. With model-based testing we can with the help of application models generate test cases automatically which complements the work of manual testing in test case designs. This in the long run reduce maintenance costs of the test suite.

*Can we achieve full automated testing using model-based testing approach?*

There are a lot of benefit to using model-based testing. The MBT is good for Fault detection, reducing testing cost, improving test quality, and automatic generation of traceability matrix. Just as there are advantages, there are also limitations to MBT. Some limitations include, inadequate skilled model designers, MBT is mostly limited to functional testing, difficulty in modelling some parts of the real system among others. In view of this limitations, we can make the conclusion that we cannot achieve full automated testing with MBT.

*How model-based testing reduces the maintenance costs of the test suite?*

Test suites helps identify the gap in testing efforts and find where tests have not successfully completed. This helps us know where the next test case should begin. When test case is generated in MBT, they can be stored and be referred later with the help of traceability matrix this help observe any mishap, without necessarily generating new test cases. This in essence reduce time and maintenance cost.

*Software testing and quality assurance can be used interchangeably sometimes for the same meaning. However, there is a clear difference between quality assurance and testing. With the help of practical examples, clarify how you differentiate between them?*

Quality assurance ensures that projects are completed according to the approved standards, specifications, and functionality. Quality assurance is the bigger concept that contains quality control and testing. Software testing on the other hand focuses on detecting and solving software flaws and error.

With a practical example, when you buy 2 kilos of green apples from the market, the quality assurance bit ensures that the apples you get are green and red or mangoes. They also make sure that you get 2 kilos and not more or less. But how do one determine that the apples are ripped or have no worms. The only way to determine this is to take a bite, this is where software testing comes in. You carry out testing on the apples to ensure that indeed, they are apples, and they taste as such. Just like software testing one can does not see flaws in software unless it is tested.

Another example will be having a web application that can fit all screen sizes and appear to look very good but will crush or fail to respond when numbers are used as input instead of letters. This is a clear example of quality assurance and testing.

*What is the relationship between the two practices?*

The relationship of the two practices go hand in hand, whilst Quality control ensure general conformance to standards, functionalities and requirements, Software testing ensures the source code and design are also tested.

*How testing the functional and non-functional attributes (characteristics) can contribute to software quality assurance? Give examples for both functional and non-functional attributes.*

Functional attributes ensure that software products meet the design and requirement specification. Some examples of functional attributes in a software product from the perspective of a customer include ease of use, performance, absence of defects among others.

Non-Functional ensures the core characteristics of the systems are realised and requirements are tailored to fit a specific business goal. This includes security, efficiency, maintainability, legal or regulatory requirements and so on.