Object Oriented Testing methods

Testing is a continuous activity during software development. In object-oriented systems, testing encompasses three levels, namely, unit testing, subsystem testing, and system testing.

TESTING TECHNIQUES FOR OBJECT-ORIENTED SOFTWARE:

Certain subset of the testing techniques covered in the study can be favourably applied to object-oriented programs. At various levels of testing of object oriented software, techniques which can be applied are

1. Unit Testing
2. Method Testing
3. Class Testing
4. Integration Testing
5. System Testing

Testing Object-Oriented Systems

Testing is a continuous activity during software development. In object-oriented systems, testing encompasses three levels, namely, unit testing, subsystem testing, and system testing.

Unit Testing

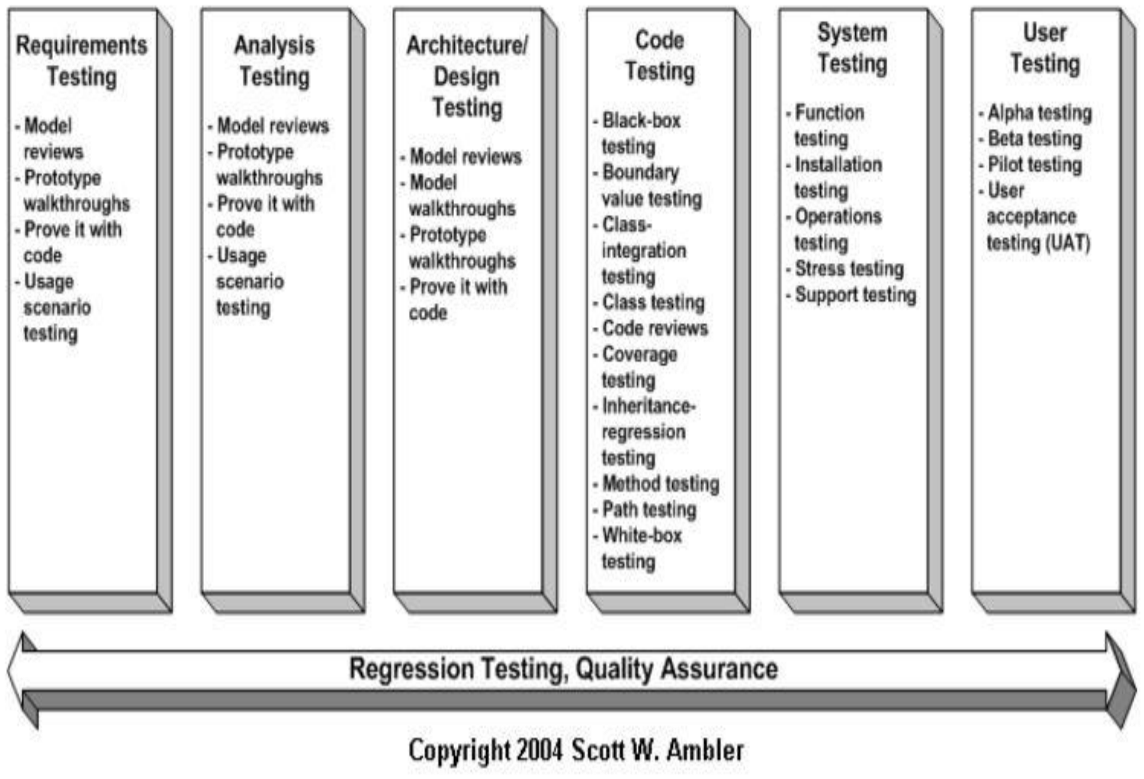
- In unit testing, the individual classes are tested. It is seen whether the class attributes are implemented as per design and whether the methods and the interfaces are error-free.
- Unit testing is the responsibility of the application engineer who implements the structure.

Subsystem Testing

- This involves testing a particular module or a subsystem and is the responsibility of the subsystem lead. It involves testing the associations within the subsystem as well as the interaction of the subsystem with the outside.
- Subsystem tests can be used as regression tests for each newly released version of the subsystem.

System Testing

- System testing involves testing the system as a whole and is the responsibility of the quality-assurance team. The team often uses system tests as regression tests when assembling new releases.



Object-Oriented Testing Techniques:

Grey Box Testing:

The different types of test cases that can be designed for testing object-oriented programs are called grey box test cases. Some of the important types of grey box testing are:

- **State model based testing:** This encompasses state coverage, state transition coverage, and state transition path coverage.
- **Use case based testing:** Each scenario in each use case is tested.
- **Class diagram based testing:** Each class, derived class, associations, and aggregations are tested.
- **Sequence diagram based testing:** The methods in the messages in the sequence diagrams are tested.

Techniques for Subsystem Testing:

The two main approaches of subsystem testing are:

- **Thread based testing:** All classes that are needed to realize a single use case in a subsystem are integrated and tested.
- **Use based testing:** The interfaces and services of the modules at each level of hierarchy are tested. Testing starts from the individual classes to the small modules comprising of classes, gradually to larger modules, and finally all the major subsystems.

Categories of System Testing:

- **Alpha testing:** This is carried out by the testing team within the organization that develops software.
- **Beta testing:** This is carried out by select group of co-operating customers.
- **Acceptance testing:** This is carried out by the customer before accepting the deliverables.

Black-box testing:
Testing that verifies the item being tested when given the appropriate input provides the expected results.

Boundary-value testing:

Testing of unusual or extreme situations that an item should be able to handle.

Class testing:
The act of ensuring that a class and its instances (objects) perform as defined.

Component testing:
The act of validating that a component works as defined.

Inheritance-regression testing:

The act of running the test cases of the super classes, both direct and indirect, on a given subclass.

Integration testing:
Testing to verify several portions of software work together.

Model review:
An inspection, ranging anywhere from a formal technical review to an informal walkthrough, by others who were not directly involved with the development of the model.

Path testing:
The act of ensuring that all logic paths within your code are exercised at least once.

Regression testing:

The acts of ensuring that previously tested behaviours still work as expected after changes have been made to an application.

Stress testing:

The act of ensuring that the system performs as expected under high volumes of transactions, users, load, and so on.

Technical review:

A quality assurance technique in which the design of your application is examined critically by a group of your peers. A review typically focuses on accuracy, quality, usability, and completeness. This process is often referred to as a walkthrough, an inspection, or a peer review.

User interface testing:

The testing of the user interface (UI) to ensure that it follows accepted UI standards and meets the requirements defined for it. Often referred to as graphical user interface (GUI) testing.

White-box testing:
Testing to verify that specific lines of code work as defined. Also referred to as clear-box testing.