OBJECT - ORIENTED TESTING
Characteristics of object-oriented testing

• The meaning of unit testing changes dramatically because each class and each instance of a class (object) encapsulates attributes (data) and the operations (also known as methods or services) that manipulate these data
  • We can not test a single operation in isolation but rather as part of a class

• Classes have larger grain than individual functions so approaches to white-box testing have to be extended

• OO software does not have a hierarchical control structure; there is no obvious ‘top’ to the system
Testing levels

• Testing object classes
• Testing clusters of cooperating objects
• Testing the complete OO system
Object class testing

• Complete test coverage of a class involves
  • Testing all operations associated with an object
  • Setting and interrogating all object attributes
  • Exercising the object in all possible states

• Inheritance, polymorphism and dynamic binding can have
  • Positive effect on development
  • Negative effect on testing and maintenance
Problems with inheritance

• Inheritance makes it more difficult to design object class tests as the information to be tested is not localised

• When the base class is changed, all derived classes are changed in the same way
Problems with polymorphism and dynamic binding

- Suppose that **myFile** is declared to be an instance (an object) of **FileClass** and that the product fails on the invocation **myFile.open( )**
- Which version of open contains the fault?
Object integration

• Levels of integration are less distinct in object-oriented systems – conventional top-down and bottom-up integration have little meaning

• Cluster testing is concerned with integrating and testing clusters of cooperating objects

• Identify clusters using knowledge of the operation of objects and the system features that are implemented by these clusters
Approaches to cluster testing

- **Use-case or scenario testing**
  - Testing is based on a user interactions with the system
  - Has the advantage that it tests system features as experienced by users

- **Thread testing**
  - Tests the systems response to events as processing threads through the system

- **Object interaction testing**
  - Tests sequences of object interactions that stop when an object operation does not call on services from another object
Testing the complete OO system

- Once the OO system is integrated the classical strategies for verification and validation are applied.
- In addition, test cases may be derived from the object-behavior model and from the event flow diagram created as part of OOA.