testing reduces a significant amount of testing efforts and time duration required for testing software so that the overall development time of software is reduced.

7. **Conduct formal technical reviews to evaluate the nature, quality or ability of the test strategy and test cases.**
   The formal technical review helps in detecting any unfilled gap in the testing approach. Hence, it is necessary to evaluate the ability and quality of the test strategy and test cases by technical reviewers to improve the quality of software.

8. **For the testing process, developing a approach for the continuous development.**
   As a part of a statistical process control approach, a test strategy that is already measured should be used for software testing to measure and control the quality during the development of software.

Unit: IV
Lecture: 6
Software Testing Strategies
(Part-II)

Unit Testing

Software testing is one element of a broader topic that is often referred to as verification and validation (V&V).

- **Verification refers to the set of activities that ensure that software correctly implements a specific function.**
- **Validation refers to a different set of activities that ensure that the software that has been built is traceable to customer requirements.**

Verification: "Are we building the product right?"
Validation: "Are we building the right product?"

Difference between Verification & Validation:

Verification	Validation
Are you building it right?	Are you building the right thing?
Ensure that the software system meets all the functionality.	Ensure that functionalities meet the intended behavior.
Verification takes place first and includes the checking for documentation, code etc.	Validation occurs after verification and mainly involves the checking of the overall product.
Done by developers.	Done by Testers.
Have static activities as it includes the reviews, walkthroughs, and inspections to verify that software is correct or not.	Have dynamic activities as it includes executing the software against the requirements.
It is an objective process and no subjective decision should be needed to verify the Software.	It is a subjective process and involves subjective decisions on how well the Software works.

The software engineering process may be viewed as the spiral illustrated in below figure:

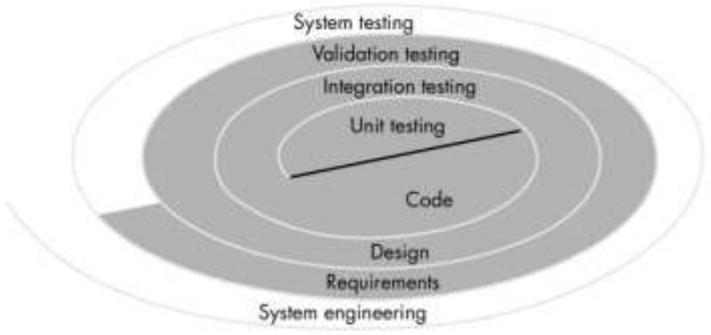


Fig: Testing Strategy

A strategy for software testing may also be viewed in the context of the spiral:

- **Unit testing begins** at the vortex of the spiral and concentrates on each unit (i.e., component) of the software as implemented in source code.
- Testing progresses by moving outward along the spiral to **integration testing**, where the focus is on design and the construction of the software architecture.
- Taking another turn outward on the spiral, we encounter **validation testing,** where requirements established as part of software requirements analysis are validated against the software that has been constructed.
- Finally, we arrive at **system testing**, where the software and other system elements are tested as a whole. To test computer software, we spiral out along stream□lines that broaden the scope of testing with each turn.

Unit Testing:

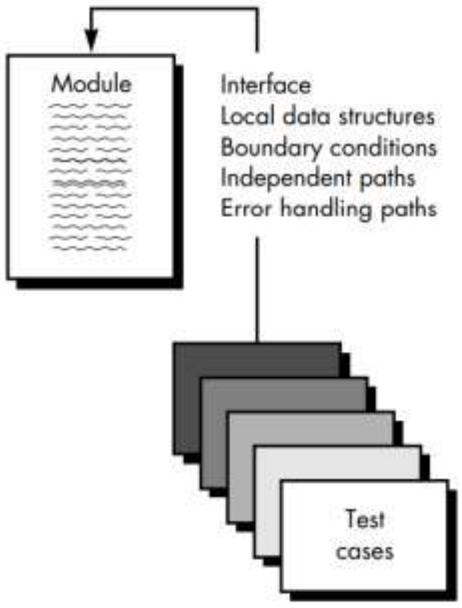
Unit testing is defined as a type of software testing where individual components of a software are tested.

Unit testing of software product is carried out during the development of an application. An individual component may be either an individual function or a procedure. Unit testing is typically performed by the developer.

The objective of unit testing is:

- To isolate a section of code.
- To verify the correctness of code
- To test every function and procedure.
- To fix bug early in development cycle and to save costs.
- To help the developers to understand the code base and enables them to make changes quickly.
- To help for code reuse.

The tests that occur as part of unit tests are illustrated schematically in below figure:



The module interface is tested to ensure that information properly flows into and out of the program unit under test.

The local data structure is examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithm's execution.

Boundary conditions are tested to ensure that the module operates properly at boundaries established to limit or restrict processing.

All independent paths (basis paths) through the control structure are exercised to ensure that all statements in a module have been executed at least once. And finally, all error handling paths are tested.

Unit testing is simplified when a component with high cohesion is designed. When only one function is addressed by a component, the number of test cases is reduced and errors can be more easily predicted and uncovered.