User Acceptance Testing

Concepts

Users/customers have a very strong need to be sure the systems they depend on actually meet business requirements, work properly, and truly help them do their jobs efficiently and effectively. However, users seldom are confident or comfortable testing system acceptability. This intensive interactive seminar shows users what they need to know to confidently make the best use of their time planning and conducting acceptance tests that catch more defects at the traditional tail-end of development, while also contributing in appropriate ways to reducing the number of errors that get through the development process for them to catch in UAT. Exercises give practice using practical methods and techniques.

Participants will learn:

- Appropriate testing roles for users, developers, and professional testers; and what each shouldn't test.
- How Proactive Testing throughout the life cycle reduces the number of errors left to find in UAT.
- Key testing concepts, techniques, and strategies that facilitate adaptation to your situation.
- Systematically expanding acceptance criteria to an acceptance test plan, test designs, and test cases
- Supplementing with requirements-based tests, use cases, and high-level structural white box tests.
- Techniques for obtaining/capturing test data and carrying out acceptance tests.

Who Should Attend:

This course has been designed for business managers and system users responsible for conducting user acceptance testing of systems they must depend on, as well as for system managers, project leaders, analysts, developers, quality/testing professionals, and auditors.

Outline

- Role of User Acceptance Testing
  - Why users may resist involvement
  - Making users confident about testing
  - Objectives, types, and scope of testing
  - Acceptance testing as user’s self-defense

- Why technical tests don't catch all the errors
- Essential elements of effective testing
- CAT-Scan Approach to find more errors
- Proactive Testing Life Cycle model
- Separate technical and acceptance test paths
- Place of UAT in overall test structure
- Making sure important tests are done first
- Developer/tester/user test responsibilities

- Defining Acceptance Criteria
- Defining acceptance test strategy up-front Source and role of acceptance criteria
- 5 elements criteria should address
- Functionality the user must demonstrate
- How much, how often user must test
- Determining system quality
- Who should carry out acceptance tests?
- How acceptance tests should be performed
- Added benefit, revealing requirements errors

- Designing Acceptance Test Plans
- Expanding the acceptance criteria
- Allocating criteria to system design
- Refining the design to catch oversights
- Checklist of common problems to test
- Equivalence classes and boundary values
- Making quality factors (attributes) testable
- Structural testing applicable to users
- GUI features that always need to be tested
- Defining requirements-based tests
- Constructing use cases
- Cautions about use case pitfalls
- One- and two-column use case formats
- Turning use cases into tests

- Consolidating tests into efficient test scripts
- Carrying Out Acceptance Tests
- Differentiating test cases and test data
- Traps that destroy value of acceptance tests
- Warning about conversions
- Documentation, training, Help tests
- Configuration, installation, localization
- Security, backup, recovery tests
- Suitability of automating acceptance testing
- Performance, stress, load testing
- Issues on creating test conditions, data
- Capturing results, determining correctness
- User's defect tracking and metrics