

This is a 3 day, practical hands-on course that covers the critical path of testing and numerous techniques to implement quality into the process. Students will learn the terminology, process, and challenges of testing in the real world. You will also learn how to find software defects early in the development lifecycle before they become more costly and risky to fix. You will gain a good working knowledge of testing and what it takes to design and conduct an effective test of software, regardless of the technology.

Course Objectives:

- Learn how to develop test cases and test plans.
- Identify the appropriate metrics to measure progress, performance & quality.
- Learn techniques to ensure that an information system protects data and maintains functionality.

Audience: Quality assurance specialists, quality control analysts, system testers, programmers, business analysts, systems analysts, project managers, support analysts, engineers, and acceptance testers.

Prerequisites: None.

Number of Days: 3 days

1	<p>Introduction to Testing & QA Objectives / observations Impediments, opportunities, and managing Responsibilities during testing Testing definitions Starting testing early vs. late start testing Quality assurance Quality tools / steps / suggestions Opportunities to improve the testing process Defining the development life cycles Measuring performance / reliability metrics Product development and testing phases and objectives</p>		<p>Specification problems / defect classification Detailing the scripts and cases Unit vs. System or acceptance testing Positive and negative testing Blind testing Use case analysis Regulating the change control process</p>
2	<p>Major Software Development & Testing Issues Functional specifications and design documents Preparing and validating the specifications Quality assurance</p>	3	<p>Test Methodologies & Checklists Setting test objectives and identifying tests Test planning Using test methodologies Computing the test coverage Black box vs white box testing Boundary value testing Path analysis or Cyclomatic complexity Decision tables State machines State transition Factor analysis OATS– orthogonal array testing strategy Pairs and magic squares</p>

	Using checklists to improve testing quality		Abstraction
4	Risk Analysis		Encapsulation
	Ascertaining the Value of a Test		Inheritance
	Assessing the Level of Risk		Object-Oriented systems testability issues
	Assigning a Relative Cost to Testing		Object-Oriented testing approach
5	Test Planning		Using test clients
	Unit testing (early testing)		Other testing issues
	Creating and auditing the unit test plan	10	Software Tools for Testing
	Integration testing and system testing		Automated Testing Considerations
	System / acceptance testing		Test Tools
	Creating and auditing the system test plan	11	Web-Based Testing
	Regression testing		Web-based testing: where to begin
	Defining the traceability matrix		Determining what to test
	Operability/Usability testing		Where to test: client side or server side
	Determining when testing is complete		Web testing responsibilities/ checklists
	Estimating the testing effort		Agile methodology and testing
	Estimating techniques		What changes with agile?
	Data dictionaries		Agile principles
	Approaches to testing		Extreme programming
	Using the test notebook		Productivity measure: velocity
6	Defect Prevention		XP basic rules and definitions
	Identifying functional specifications defects	12	Testing / practices in XP shops
	Identifying design defects		Capability Maturity Model Integration (CMMI)
	Identifying coding defects		Defining the CMMI
	Identifying testing defects		Capability level 0: incomplete
	Defining the coding/testing standards		Capability level 1: performed
7	Test Management		Capability level 2: managed
	Exploring the test logs		Capability level 3: defined
	Test logging scenarios		Capability level 4: quantitatively managed
	Exploring the defect tracking report		Capability level 5: optimizing
	Retesting and follow - up procedures	13	Security Testing
	Understanding root cause analysis		Security guidelines, rules, and regulations
8	Problem Solving Techniques		Requirements
	Error isolation		Development
	Variable tracers		Installation
	Flowcharts		Security services
	Deductive questioning		Cryptographic security mechanisms
	Structured walkthroughs		Security infrastructures
	Joint Application Design (JAD)		
9	Object-Oriented Testing		
	Overview		
	Object-Oriented vs. Traditional testing		
	Managing Complexity		