International Institute for Software Testing
Promoting Disciplined Software Testing Practices

Effective Test Management Practices

Dr. Magdy Hanna

Chairman

International Institute for Software Testing

mhanna@testinginstitute.com

http://www.testinginstitute.com

What is most frustrating in your role as a test manager or test lead?

What are you doing about it?

What do you currently do to demonstrate the value of your test team?

Is testing a quality assurance activity? or a quality control activity?

Quality Assurance vs. Quality Control

- Quality control is any end-of-cycle activity that aims at evaluating the product to detect bad characteristics.

- Quality assurance involves all activities that aim at evaluating and improving processes to build good quality into the product and to prevent bad characteristics from getting into the product.

Disclaimer

The following set of practices reflect the speaker's experience and are not intended to provide a complete or comprehensive set of practices that test and QA managers must follow to manage their test projects.

Some obvious practices were intentionally left out of this presentation to keep a manageable scope.

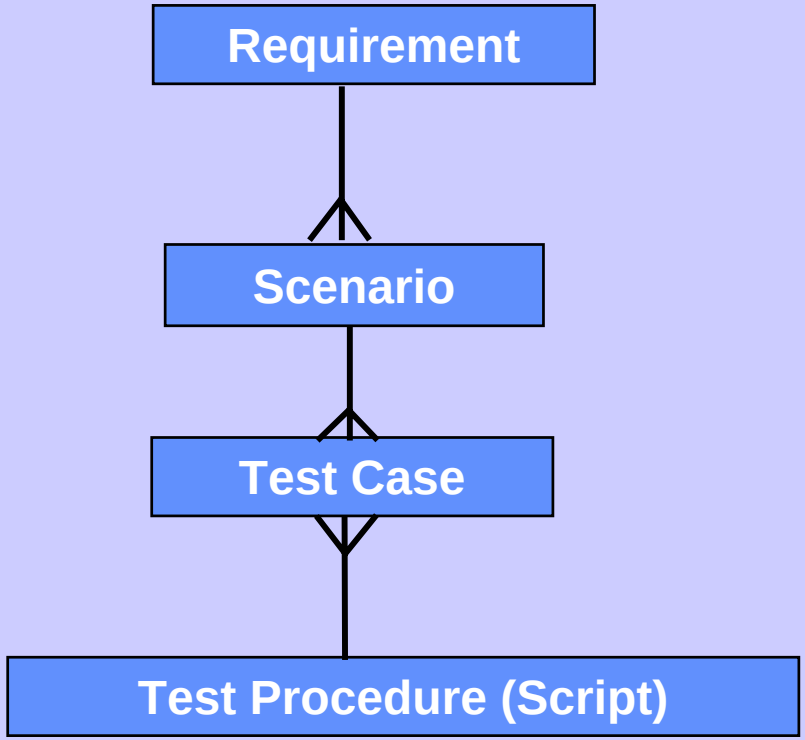
PRACTICE 1

DEFINE YOUR TEST TERMINOLOGY FOR YOUR TEST TEAM

Terminology is the Common Language Every Team Member Must Speak

- Whatever terminology you use, use it consistently and enforce using it
- Consult a number of references and choose the terminology that makes sense to you
  - IEEE
  - International Institute for Software Testing (www.iist.org)
- If you are using test tools, you may have to live with their terminology. It is not hard.
- Concepts is the key, not the terms.

Testware Relationships



Requirement

A Requirement (Functional): A statement of some desirable capability of the system

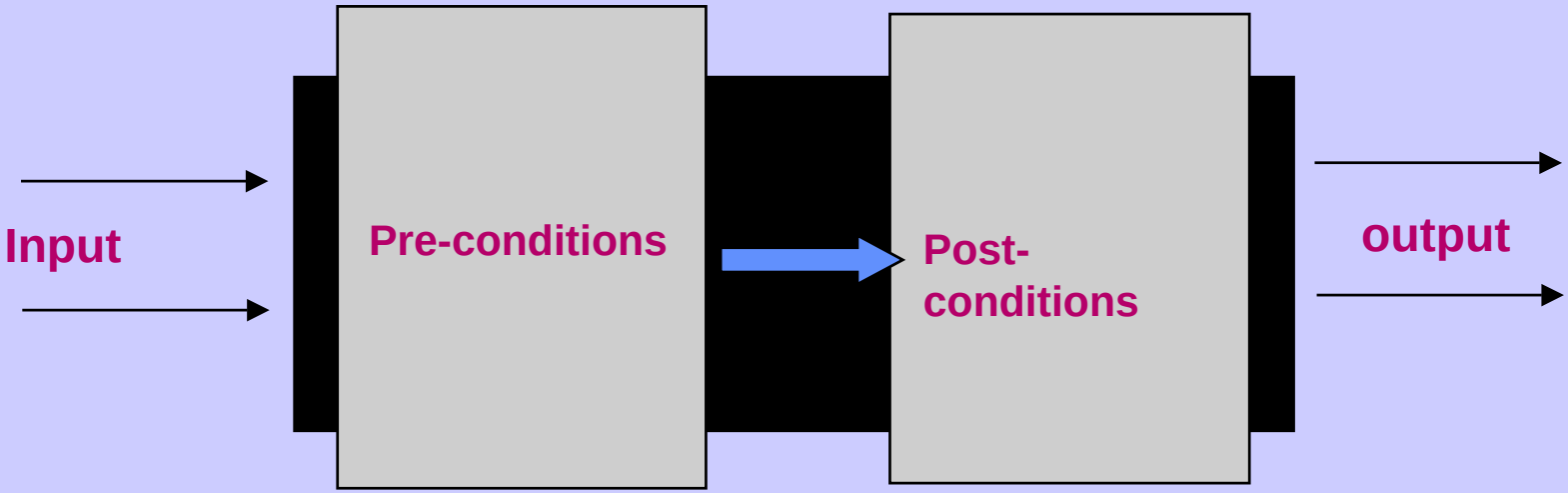
A Requirement (Quality): A statement of some desirable behavior or characteristic of the system

In order to be tested, a requirement must be stated as detailed, complete, precise, and as unambiguous as possible

Scenario

- A scenario is a situation that could possibly happen when the system is in production.
- Typically, many scenarios can be identified for each requirement.
- The goal of the test process must be to identify as many scenarios as possible.

Black Box Testing vs. Gray Box Testing



Input: Any data coming from outside the system

Output: Any data going from the system to outside the system

Pre-conditions: Current state of the system and/or environment before you run the test case

Post-conditions: Changes in the state of the system and/or environment as a result of running the test case.

A Test Case

A Test Case: Data sets that will cause the system to exercise a specific scenario or a condition. This includes:

- Input data
- Pre-conditions
- Post-conditions
- Output

Input + Pre-condition → Output + Post-condition

More than one test case may be used to cover a specific scenario or a condition.

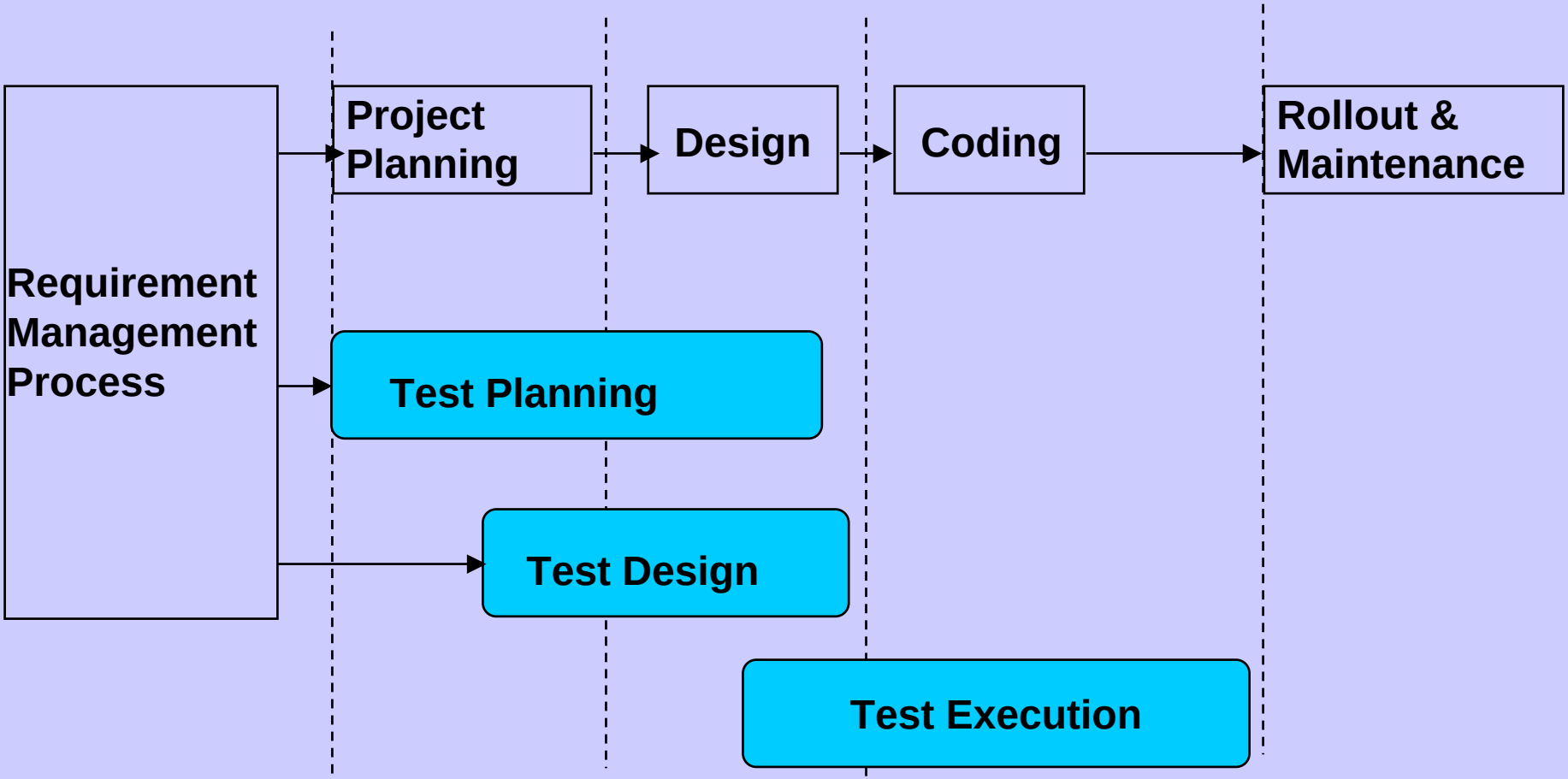
A Test Script or A Test Procedure

A set of instructions to be followed to execute a number of related test cases. A test script may test one or more scenarios for a given requirement or for different requirements.

PRACTICE 2

DEFINE YOUR TEST PROCESS

Testing is Not a Phase



Define The Process

- Define tasks within the process (Planning, Design, Execution, etc).
- Define activities within each task and responsibilities
- Define contents and format of all deliverables from each task
- Define entry and exit criteria for each task
- Define all interactions between each task and other “external” processes
- Be aware of any side communications between developers, customers and analysts

PRACTICE 3

GET THE TEAM INVOLVED WITH THE REQUIREMENT PROCESS AS EARLY AS POSSIBLE

It Is Not Easy!

- Speak with management
- Speak with developers and analysts
- Learn their language
- Learn requirement techniques
  - Use cases
  - Data models
  - Process models
  - Object models
  - Decision tables
  - Equivalence classes

- Show contribution

Requirement Testability

- Requirements must be reviewed for:
  - Correctness
  - Consistency
  - Completeness
  - Precision and unambiguousness

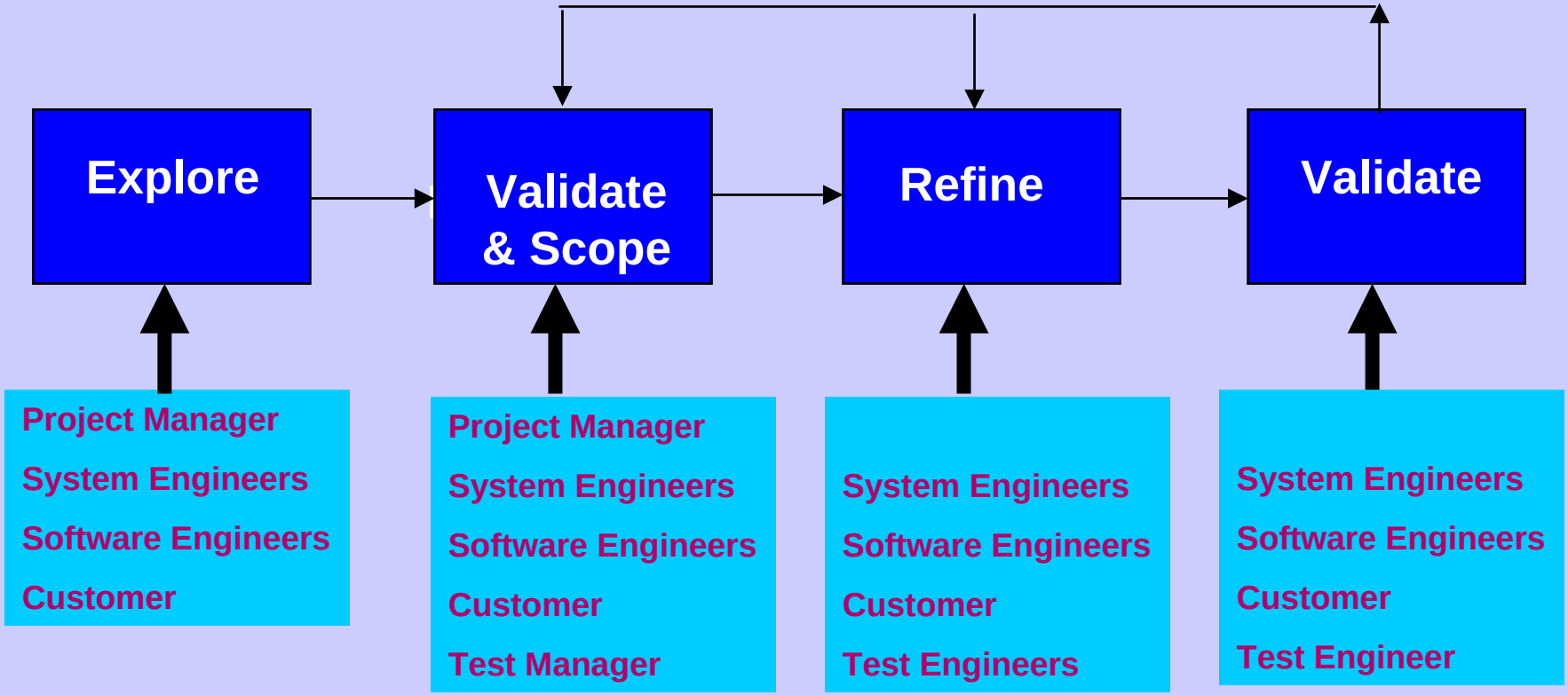
No Requirements?

- Look for existing documents
  - Manuals
  - Help files
  - User interface
- The ‘reasonable’ user
  - What would you expect it to do?
- Document everything you
  - Discover
  - Theorize
  - Prove

Effective Requirement Management Practices

- Requirement Management process in place
- Reviews and validation by testers, developers, analysts, and customers
- Use of models or artifacts other than natural language text
- Change Control Process in place
- Control side interactions between developers, analysts and customers
- Release management process in place to control scope creep
- Use of requirement management tools will help, but they are expensive

Where Do Test Professionals Fit in the Requirement Process



PRACTICE 4

START TEST DESIGN DURING THE REQUIREMENT PROCESS

It Is NOT That Difficult

- Mainly, identify test scenarios
- Best Practices:
  - Utilize Use Cases
  - Utilize Decision Tables and Decision Trees
  - Utilize Equivalence Classes
  - Utilize other Requirement-Based Test Design Techniques
- Acceptable Practices:
  - Examine user interface if available
  - Examine vague and incomplete requirement statements and ask questions, clarify, and document.
- Worst Case Practices
  - No requirements and no cooperation from developers and analysts?
  - Make your case to management
  - Try to help; take responsibility and start doing something
  - Requirement Management Process is a must. Do not go for too long without it

PRACTICE 5

CONSISTENTLY CONDUCT TEST DESIGN REVIEWS

What to Review?

- Test Scenarios:
  - Mandatory
- Test Cases:
  - Pre-conditions and post conditions
- Test Scripts ???

Use a flexible review process.

PRACTICE 6

PREPARE YOUR TEAM TO ATTEND CODE AND DESIGN REVIEW MEETINGS

What do they do there?

- Listen carefully and ask intelligently
- Look for mismatch between what you know and what is in the code or design
- Document what you learn
- Share with other team members
- Enhance your test designs

Testers need to market themselves to the rest of the project team!

PRACTICE 7

DEFINE YOUR TEST STRATEGY

First: Define The Goal of the Test Effort

- Must be stated precisely
- Only possible with good understanding of:
  - The size of the project
  - Deadlines and budget
  - Customer expectations
  - Potential risks
- A well defined goal is essential for effective test planning

Some Examples of Test Effort Goals

- All Functional requirements and quality requirements described in the requirement documents will be tested
- All tests must pass
- 100% Regression test will be performed
- Regression test will include only those tests that are directly related to a changed function
- Performance testing will be performed on an identical copy of production data
- Testing will cover only critical and high priority functions as defined by the user – medium and low priority functions are assumed to have been tested by the development team
-
-

Examples of Issues to Consider When Developing A Test Strategy

- Selecting test design techniques
- Use of test automation tools
- Developing in-house test harnesses
- Determining staffing and skills
- Determining test coverage (code, requirements, etc.)
- Establishing the order of tests
- Establishing release criteria
- Setting the schedule

PRACTICE 8

PERFORM REGRESSION TESTING THAT IS BASED ON IMPACT ANALYSIS

What is Impact Analysis?

- A study of the impact of changes made by developers on other functional and QUALITY requirements

Performing Impact Analysis

- When a requirement is changed, what components are likely to be affected?
- When a component is changed, what requirements need to be retested?
- Impact analysis starts in development by programmers and continues during system testing.
- Test manager must request impact analysis reports from the development team.
- *An impact analysis report* is simply a list of functional and quality requirements that the development team "suspects" have been impacted by the changes they made.

Enabling Impact Analysis

	Req 1	Req 2	Req 3	Req 4	Req n
Component 1	X		X		
Component 2			X	X	
Component 3	X	X	X		
Component 4					
Component n					

PRACTICE 9

GET READY FOR TEST EXECUTION BEFORE IT STARTS

Test Execution Time is Too Short

- Don’t count on using it for anything other than test execution activities.
- Plan your pre-execution activities very carefully.
- Assign responsibilities, stay focused, get status.
- Plan your test execution and build your timeline.
- Share plan and timeline with project management, development management and test team as soon as they are ready.

PRACTICE 10

COLLECT SIMPLE MEAUREMENTS FOR PROCESS IMPROVEMENT AND REPORTING

What to Measure?

Keep it simple, set goals for improvement, monitor.

- *Defect Fix Time:* Time from opened to fixed
- *Defect Test Time:* Time from fixed to tested
- *Defect Fix Failure Rate:* % of fixes that fail on retest (quality of fixes)
- *Defect Age:* Time from opened to closed (including re-testing)
- Test Execution Efficiency: % of time spent on test execution activities as oppose to other “Catch up” activities

PRACTICE 11

TRACK TEST EXECUTION PROGRESS FOR EFFECTIVE STATUS REPORTING

	R1	R2	R3	R4
Test Case 1	P		P	P
Test Case 2	P	F		
Test Case 3	F		P	
Test Case 4	P	P		
Test Case 5		P		
Test Case 6			P	
Test Case 7		P		
Test Case 8				X
Test Case 9				X
Test Case 10		F		
Test Case 11			P	
Test Case 12				P
Test Case 13				P
Test Case 14				P
Test Case 15			P	
Test Case 16		P		
Total	4	6	5	6
Total Pass	3	4	5	4
Total Fail	1	2	0	0
% of Completion	75%	66%	100%	66%

PRACTICE 12

BUILD A TEAM OF TEST PROFESSIONALS NOT JUST A TEAM OF TESTERS

The Test Team

- Two bad practices that I have seen:
  - Test team members are development wannabees (they must serve their time in test before “graduating” to developer)
  - Test team members are programmers who couldn’t cut the mustard in development
- The most important trait for testers:

They Want To Be Testers!

Keep in mind:

Testing is a profession and we must be proud to belong to it.

Skills Needed for The Test Professional

- Domain experts (business analysts)
- Technology experts (tools, techniques, etc.)
- Automation? (developer skills)
- People skills (the art of diplomacy)
- Persistent (must not give up)
- Analytical (What if I do this?)
- Test design techniques
- Requirement techniques
- Technical writing
- Good listener
- Good communicator

Developing Skills of The Test Team Through Continuing Formal Education

- Testing software is an engineering discipline
- On-the-job training can only go so far
- Tester Certification is one way
- The bottom line is:

EDUCATION
EDUCATION
EDUCATION

The need for education…

Q: "How many weeks of training on software testing have you completed in your professional life?"

None	43%
Less than one week	19%
One week (5 days)	7%
One to two weeks	7%
Two weeks to one month	6%
More than one month	18%

Source: Martinig & Associates Poll of 240 Software Testing professionals

The Results…

• According to the 2003 edition of The Standish Group's annual CHAOS Research Study, software project success rates are still only just over a third or 34% of all projects. 15% of all projects fail completely, while 51% of software projects remain "challenged."

• Additionally, the 2003 research shows only 52% of required features and functions make it to the released product.

•Lastly in the U.S., software bugs cost companies nearly $60 billion per year.
(Sustainable Computing Consortium)

IIST Education-Based Certifications

Certified Software Test Professional (CSTP)

Certified Test Manager (CTM)

The Value of IIST Certifications

- 80 Hrs. of education (compared to 0 for other certifications)
- Students achieve competence in areas described in the Body of Knowledge for each certification.
  - CSTPBOK
  - TMBOK
- Organization will see real money ROI as a result of individuals being certified
- Prepares individuals for a wider range of testing assignments
- Increase professional recognition and career enhancement due to real skills and knowledge gained from education
- Establishes disciplined, repeatable processes and emphasizes disciplined adherence to these processes in order to achieve consistent results

CSTP Body of Knowledge

1. **Principles of Software Testing**
   - Levels of Testing
   - Testing client/server applications
   - Testing Internet and web applications
   - Testing object-oriented applications
   - Testing embedded systems
   - The testing life cycle

2. **Test Design**
   - Code-based test case design techniques
   - Requirement-based test case design techniques
   - Test design specification

3. **Managing the Testing Process**
   - Planning
   - Scheduling
   - Reporting
   - Resources
   - Risk Management
   - Measuring and improving the test process

CSTP Body of Knowledge

4. Test Execution and Defect Tracking

- Test scripting
- Reporting
- Defect tracking

5. Requirement Definitions, Refinement and Verification

- Writing testable requirements
- Exploring requirements
- Refining requirements
- Defining requirements
- Requirement verification
- Requirement traceability

CSTP Body of Knowledge

6. Test Automation

- Tool evaluation and selection
- Architectures
- Automation standards and guidelines
- Planning the test automation process
- Automation team roles

7. Static Testing (Inspections, Reviews, and Walkthroughs)

- Types of static testing
- The process of static testing
- Defect data analysis
- Improving the process

Test Management Body of Knowledge

- **Test Process Management**
  - Quality policies, processes, and standards
  - Defining quality goals
  - Process definition
  - Process control
  - Process documentation
  - Relationship with service management infrastructure processes (incident management, problem management, configuration management, change management, release management, etc.)
  - Best practices, including use of both static and dynamic testing
  - Test processes for different development models (XP, RAD, JAD, waterfall, etc.)
- **Test Project Management**
  - Test planning
  - Effort estimation
  - Task identification
  - Scheduling
  - Tracking
  - Reporting
  - Resource allocation (people, hardware, software, and facilities)
  - Identification of roles and responsibilities
  - Project controls
  - Financial analysis and ROI
  - Metric tracking and presentation
  - Presentation skills
  - Using GANTT/PERT charts and other project management techniques
  - Using automated project management tools
  - Directing, supervising, and assessing individuals' performance
  - Leadership

Test Management Body of Knowledge

- **Test Process Measurement and Improvement**
  - Test coverage analysis
  - Incident tracking and management
  - Defining and capturing test measurements
  - Basic “best practices” development metrics
  - Alternative measurement goal-setting with the Basili goal/question/metric paradigm
  - Test maturity models
  - Establishing process goals
  - Performing assessments and using surveys
  - Benchmarking
  - Overview of process improvement models such as CMM/CMMI, Six Sigma, TQM, ISO, etc.
  - Overview of the applicable IEEE documentation standard
- **Test Organization Management**
  - Resource management
  - Staffing, hiring, contracting, and reviewing performance
  - Equipment, facilities, hardware, and software resource management
  - Politics
  - Training and career development
  - Ethics
  - Team building and retention
  - Compensation
  - Budgeting
  - Presentation skills including data preparation
- **Risk Management**
  - Risk analysis methodologies
  - Risk identification, classification, and prioritization
  - Calculating costs and probability
  - Risk reporting
  - Monitoring and controlling risks
  - Contingency planning and mitigation
  - Risk-based test planning and management: sizing and resource planning

Test Management Body of Knowledge

- **Test Automation Strategies and Architectures**
  - Defining a test automation strategy and plan
  - Test tool evaluation and selection
  - Build it vs. Buy it: automation strategies/ approaches
  - Developing skills and relevant test automation roles
  - Long term maintenance considerations
  - Selecting which tests to automate and converting from manual to automated
  - Calculating ROI of automation
  - Building a performance test team
  - Process automation and metric analysis
  - Test environments - test data, architecture, security, networks, etc.
  - Categories of automated tools that can be used to aid testing
- **Software Quality Assurance**
  - Quality Assurance Vs, Quality Control
  - Defining processes
  - Implementing Quality Assurance
  - Developing effective standards
  - Inspections and review of artifacts other than code
  - Development and Test lifecycles and methodologies
  - QA concepts, methods and approaches
  - Prominent quality assurance model such as CMM/CMMI, Six Sigma, IEEE standards, TQM, and ISO
  - ROI justification for Quality Assurance
  - Quality Assurance according to W. Edwards Deming
  - Quality Assurance for modern development methodologies: RAD, Agile and eXtreme
  - Independent Verification and Validation