Software Testing: System & Acceptance Testing

Outline

- System testing
  - Function testing
  - Performance testing
  - Acceptance testing

System Testing

- Functional Testing
  - Validates functional requirements
- Performance Testing
  - Validates non-functional requirements
- Acceptance Testing
  - Validates clients expectations

Functional Testing

Goal: Test functionality of system

- Test cases are designed from the requirements analysis document (better: user manual) and centered around requirements and key functions (use cases)
- The system is treated as black box
- Unit test cases can be reused, but new test cases have to be developed as well.

Performance Testing

Goal: Try to violate non-functional requirements

- Test how the system behaves when overloaded.
  - Can bottlenecks be identified? (First candidates for redesign in the next iteration)
- Try unusual orders of execution
  - Call a receive() before send()
- Check the system's response to large volumes of data
  - If the system is supposed to handle 1000 items, try it with 1001 items.
- What is the amount of time spent in different use cases?
  - Are typical cases executed in a timely fashion?

Types of Performance Testing

- Stress Testing
  - Stress limits of system
- Volume testing
  - Test what happens if large amounts of data are handled
- Configuration testing
  - Test the various software and hardware configurations
- Compatibility test
  - Test backward compatibility with existing systems
- Timing testing
  - Evaluate response times and time to perform a function
- Security testing
  - Try to violate security requirements
- Environmental test
  - Test tolerances for heat, humidity, motion
- Quality testing
  - Test reliability, maintainability & availability
- Recovery testing
  - Test system's response to presence of errors or loss of data
- Human factors testing
  - Test with end users

Acceptance Testing

- Goal: Demonstrate system is ready for operational use
  - Choice of tests is made by client
  - Many tests can be taken from integration testing
  - Acceptance test is performed by the client, not by the developer.
- Alpha test:
  - Client uses the software at the developer's environment.
  - Software used in a controlled setting, with the developer always ready to fix bugs.
- Beta test:
  - Conducted at client's environment (developer is not present)
  - Software gets a realistic workout in target environment

Testing has many activities

Establish the test objectives

Design the test cases

Write the test cases

Test the test cases

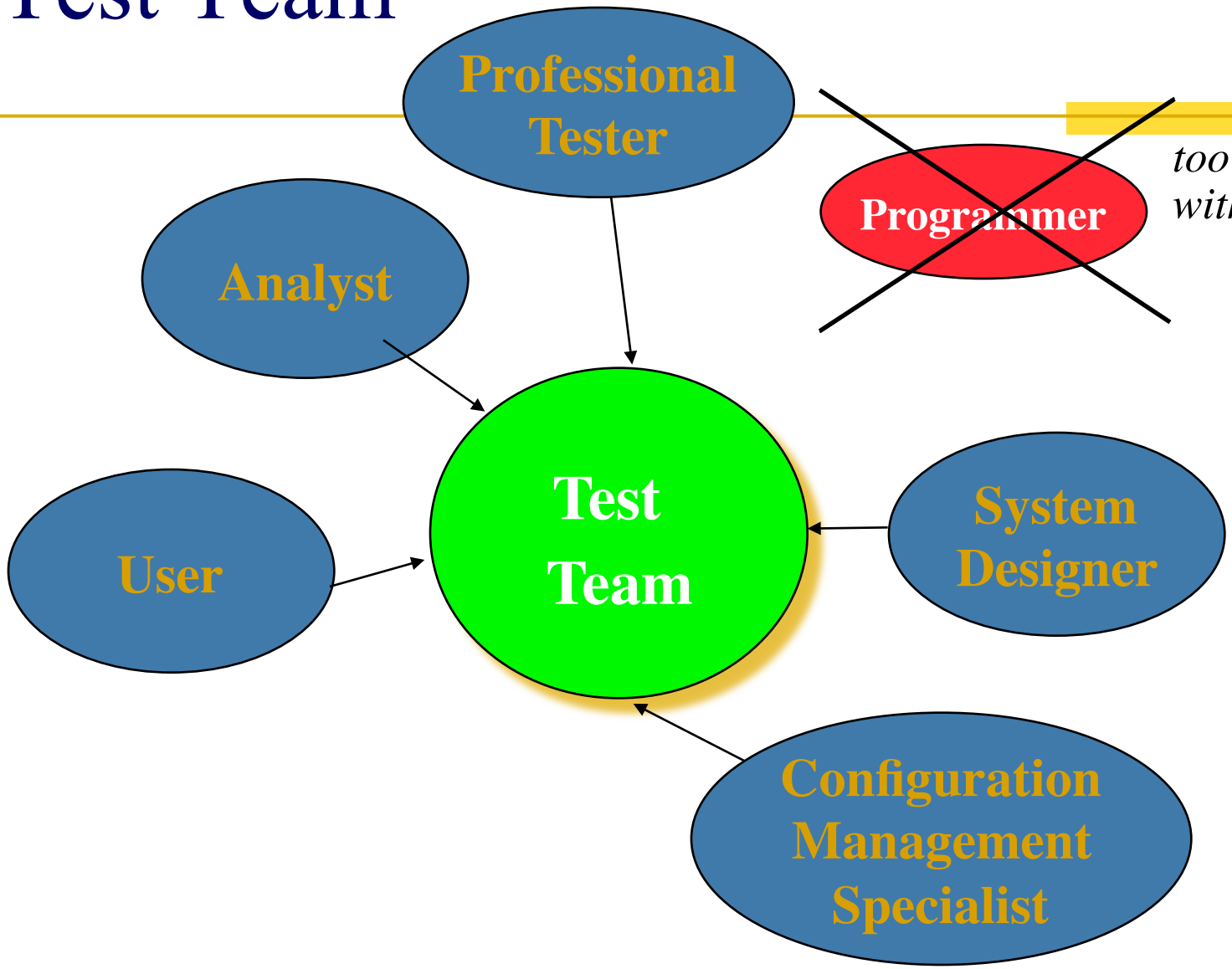
Execute the tests

Evaluate the test results

Change the system

Do regression testing

Test Team


Professional Tester
Programmer
too familiar with code
Analyst
User
Test Team
System Designer
Configuration Management Specialist

The 4 Testing Steps

1. Select what has to be tested
   - Analysis: Completeness of requirements
   - Design: Cohesion
   - Implementation: Source code
2. Decide how the testing is done
   - Review or code inspection
   - Proofs (Design by Contract)
   - Black-box, white box,
   - Select integration testing strategy (big bang, bottom up, top down, sandwich)
3. Develop test cases
   - A test case is a set of test data or situations that will be used to exercise the unit (class, subsystem, system) being tested or about the attribute being measured
4. Create the test oracle

- An oracle contains the predicted results for a set of test cases
  - The test oracle has to be written down before the actual testing takes place

Guidance for Test Case Selection

- Use analysis knowledge about functional requirements (black-box testing):
  - Use cases
  - Expected input data
  - Invalid input data
- Use design knowledge about system structure, algorithms, data structures (white-box testing):
  - Control structures
  - Test branches, loops, ...
  - Data structures
  - Test records fields, arrays, …
- Use *implementation knowledge* about algorithms and data structures:
  - Force a division by zero
  - If the upper bound of an array is 10, then use 11 as index.