Manual vs. Automated Software Testing

- BSc. (IT) – Curtin University, 2009
- MBA – PIM (University of Sri Jayewardenepura), 2014
- ISTQB Certified Advanced Test Manager
- 10+ Years Industry experience as a QA Engineer and Team Lead
- Working as Technical Lead at WSO2 Inc.
- Visiting Lecturer at SLIIT

Deshal Weerasinghe

Why do we ‘Test’ Things?

Why Test Anything?

- To know whether it is ‘working’
- To know whether it ‘fits the purpose’
- To know whether ‘something is wrong’ with it
- To know ‘how it fails’, when it fails

Why Software Should be Tested?

- Humans develop software
- They may make errors/ mistakes
- These errors built in to software are ‘defects’
- The defects will cause the software to ‘fail’
- The failures cause damages – financial and otherwise
- Testing helps to identify these defects/ failures before the software is used by customers
- One reason to test software is to reduce (not eliminate) defects
- Additionally, testing is done in order to ensure the right thing is build

We Test Software to Gather Information

- “I am only a good tester when I don't care if it works or not, only that I provide the information needed to confirm or deny that it works.”

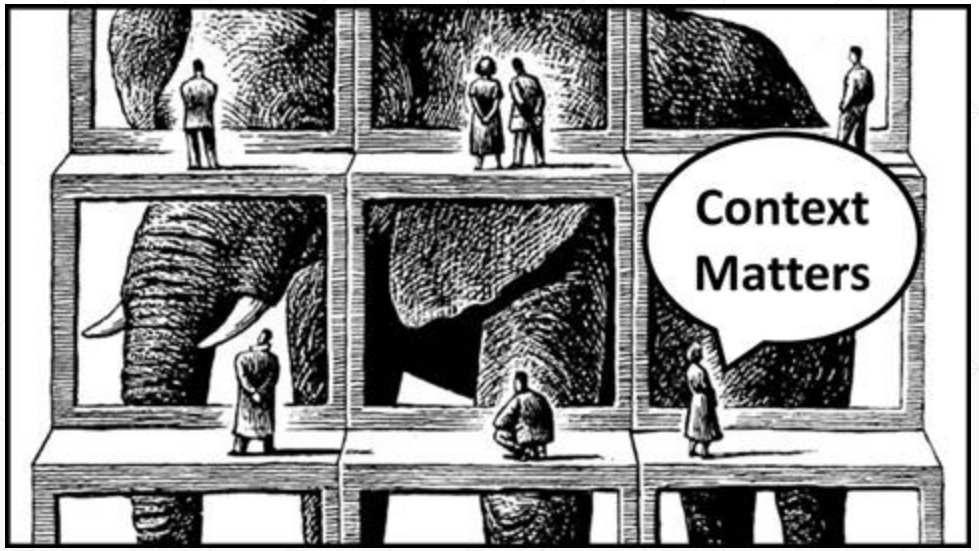
- B. Hal Metz (Test Lead Consultant, Fusion Alliance)

How Software is Tested

- Identify test context/ conditions
- Prepare test scenarios/ test cases
  - Use test design techniques
- Test in different levels of software development
- Use multiple test types
- And different test execution types to suit the task at hand
- Report test outcomes

Test Context

- Identify the type of application under test – the context
  - Eg. Bank ATM application vs. mobile email client
- Identify what features of the application to test
  - Eg. Web front end vs. API



Test Scenarios & Test Cases

- A test scenario in general is an end-to-end flow of inter-related events

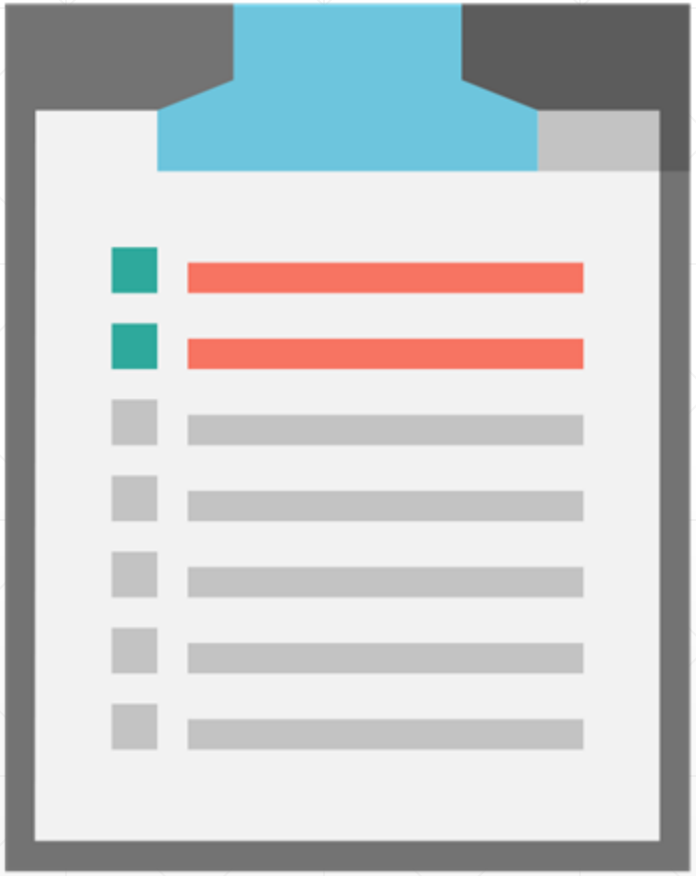
  Eg. Log in > edit profile > change profile picture > save

- A test case defines specific inputs, operations and outputs of a selected test scenario or a part of it

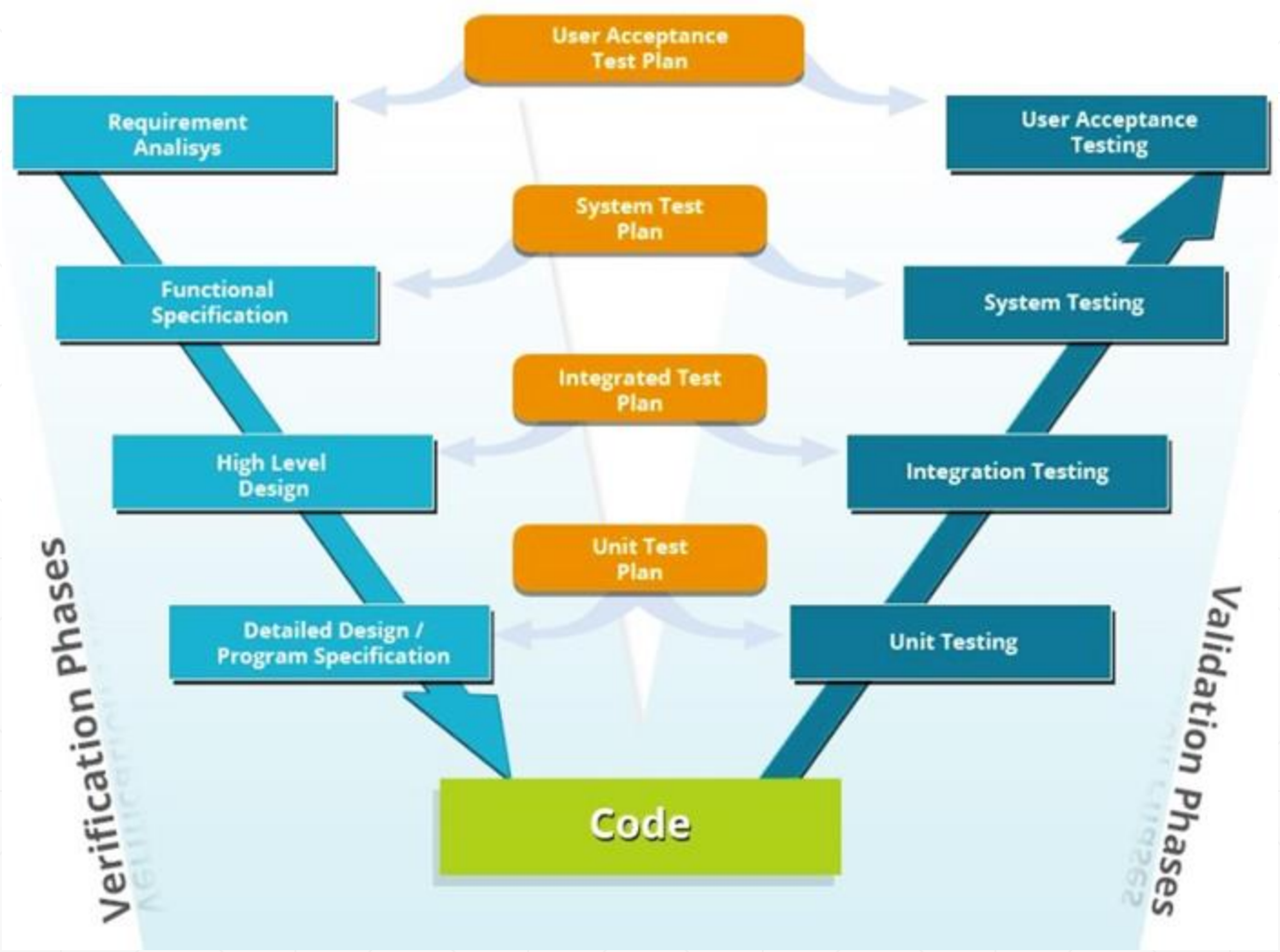
- Use test design techniques to make the test scenarios efficient and effective

Components of a Typical Test Case

- Summary
- Description
- Pre-conditions
- Post-conditions (optional)
- Test data (optional)
- Test Steps
- Expected Results/ Outputs



Test Levels


User Acceptance Test Plan
Requirement Analisys
User Acceptance Testing
System Test Plan
Functional Specification
System Testing
Integrated Test Plan
High Level Design
Integration Testing
Unit Test Plan
Detailed Design / Program Specification
Unit Testing
Code
Verification Phases
Validation Phases

Test Types

- Functional Tests
- Non-functional Tests
  - Performance, load, stress, usability, accessibility
- Structural
  - 'White-box' Testing
- Change-related Tests
  - Regression/ retesting

Test Execution Types

- Build Verification Tests (BVT)
  - Verify whether all basic functionalities of a selected application works after creating a new build with changes
- Smoke Tests
  - Verify whether a new build is stable enough to continue the rest of the testing
- Functionality Verification ('Progression')
  - Verify one or more selected functionality after initial implementation or making changes
- Regression Test
  - Verify a functionality or related functionality to assess the impact of changes made
- Re-testing
  - Test everything once again

Test Outcomes Reporting

- Bugs/ defects
- Change requests
- Status reports
- Periodic health-checks
- Test coverage reports

How Software is Tested

- Identify test context/ conditions
- Prepare test scenarios/ test cases
  - Use test design techniques
- Test in different levels of software development
- Use multiple test types
- And different test execution types to suit the task at hand
- Report test outcomes


CNN

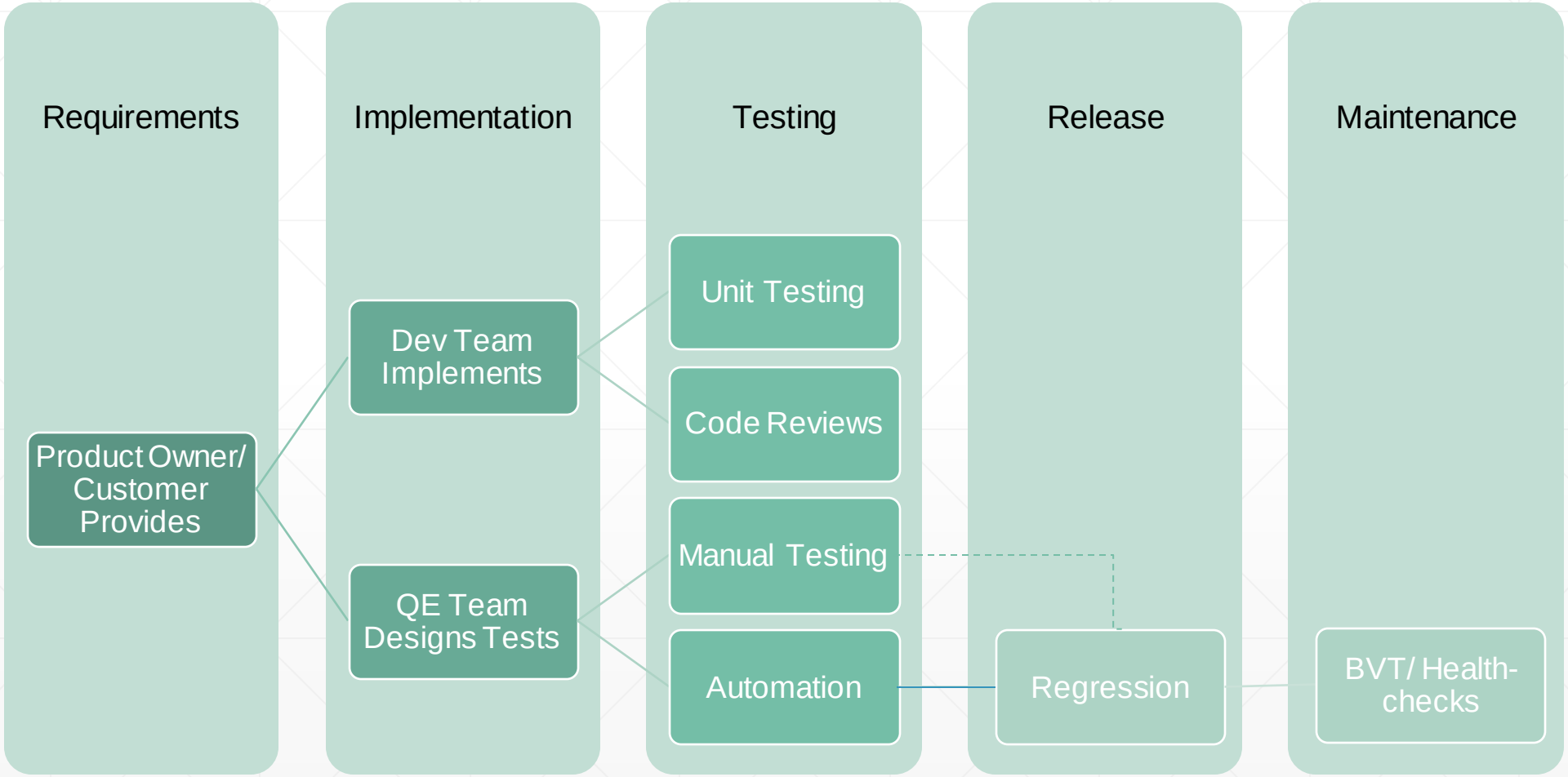
The Seven Principles of Testing from ISTQB

- Testing shows presence of defects
- Exhaustive testing is impossible
- Early testing
- Defect clustering
- Pesticide paradox
- Testing is context-dependent
- Absence of errors fallacy

Manual Testing

- A *person* executes test cases *manually*
  - Perform input tasks
  - Observe outputs/ outcomes
  - Report observations
- Requires the tester to be knowledgeable about the software and often the domain
- At least the first round of testing of a new application should be done manually

How Testing is done in a Typical SW Project



Automated Testing

- *A tool* executes test cases *automatically*
  - Perform inputs
  - Verify outputs/ outcomes (compare with expected outcomes – *'assert'*)
  - Report (generate matrices, share with stakeholders)
- More suitable for a matured application, which delivers frequent updates/ changes

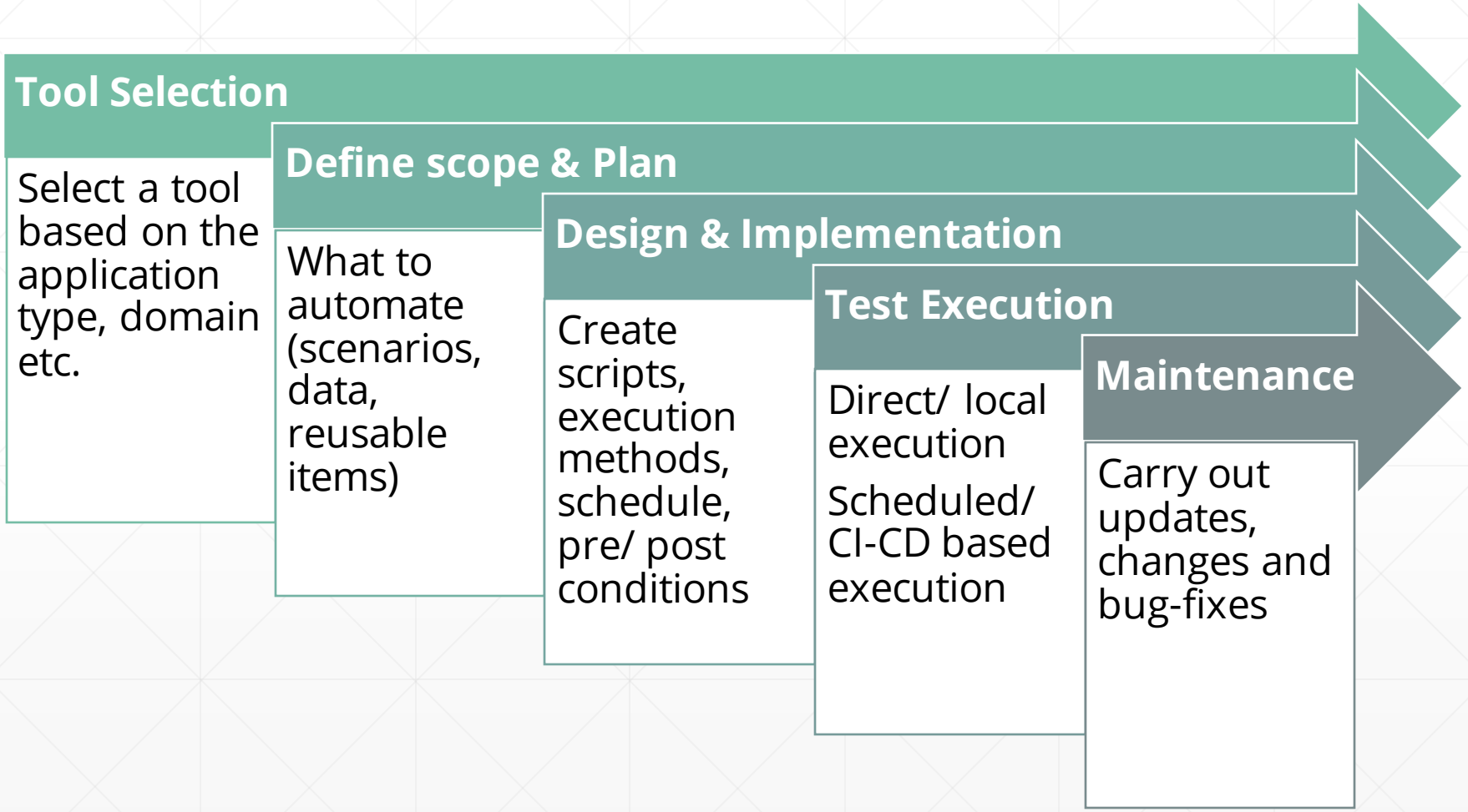
Why Testing should be Automated?

- Manual testing is time-consuming (time = money!)
- Higher error margin associated with repetitive manual test execution
- Manual testers can focus more on exploratory tests rather than repetitively executing same set of tests

What should be Automated?

- Tests that require frequent execution
  - Eg. Daily executions of BVTs/ Health-checks
- Test cases that require higher precision in execution
  - High-frequency concurrent user actions
- Tests that are difficult to perform manually
  - High number of multi-user tests (performance)

How Testing is Automated?



Advantages of Test Automation

- **Reduced execution time**
- Convenient way of executing repetitive tests frequently
- Minimal human intervention required for execution
- Easy to maintain consistency of tracking and reporting
- Helps to improve test coverage
- Less error-prone than manual testing

Common Test Automation Tools and Frameworks

- Selenium
- Junit
- TestCafe
- Gauge
- SoapUI
- Jmeter
- Postman
- TestComplete
- Appium
- Cypress

Custom Test Automation Frameworks

Overheads of Test Automation

- Requires test designers to be more tech-savvy
- May require higher setup and maintenance costs
- No one-tool-fits-all type of solutions
- Maintenance of custom test automation frameworks

That’s all!

Thank You!

Want to raise a question later? Send to *deshal.lec@gmail.com*