Basics of Testing – Theory and Practice

Nicholas Hjelmberg, Niconsult

Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Course Introduction

- The purpose of this 2-day course is to simulate a test project to help students understand the day-to-day challenges.
- The students are expected to have taken the ISTQB training and/or have testing experience
- This course will cover the following topics:
  - Test Project
  - Test Management
  - Test Strategy
  - Test Plan
  - Test Analysis
  - Test Design
  - Test Implementation
  - Test Execution
  - Test Closure
- In addition, you will put theory to practice in activities and a case work

Course Objectives

- Participants who have completed this training will be able to:
  - Plan a test project
  - Analyse requirements
  - Manage a test project
  - Follow up and report test progress

Are you missing any objectives?

Course Objectives cont.

- The objectives of this course is NOT about:
  - In-depth testing skills

    (This is covered by other courses)
  - General test management skills

    (This is covered by the ISTQB certifications, please visit

    http://istqb.org/display/ISTQB/Home)
  - Technology specific test management skills

    (This is covered by other courses)

Icebreaker Activity 1

- Test planning
  - Group in pairs
  - Get to know each other by asking questions of your own choice related to degree, project, testing experience, hobbies etc.
  - Do NOT document anything but the ”test subject” name

- Test preparation
  - Group in new pairs and exchange forms
  - Ask questions about your test subject
  - Document the information as ”test cases” (questions and expected answers)

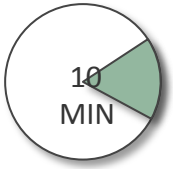
- Test execution
  - Group in new pairs and exchange forms again
  - ”Execute” your ”test cases” in front of class without naming the ”test subject”
  - After the ”test execution”, the ”test subject” should call out if he or she recognizes himself or herself
  - Was the test accurate?

Icebreaker Activity Form

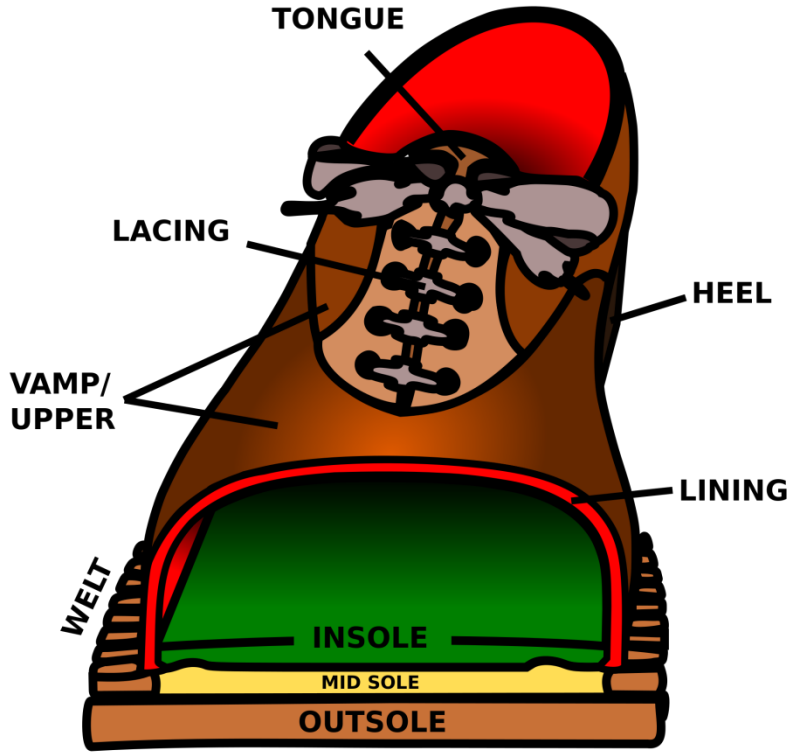
Test Subject (Name)		
Step	Question	Expected Answer
1		
2		
3		
4		
5		

Icebreaker Activity 2

- Test planning
  - Study the test object
  - Study the requirement
- Test preparation
  - Write a test case with step-by-step descriptions and expected results
- Test execution
  - Exchange forms with a colleague
  - ”Execute” your ”test cases” in front of class
  - Was the test accurate?



Requirement: The user must be able to tie the shoelaces



Icebreaker Activity Form

Test Case (Name)		
Step	Description	Expected Results
1		
2		
3		
4		
5		

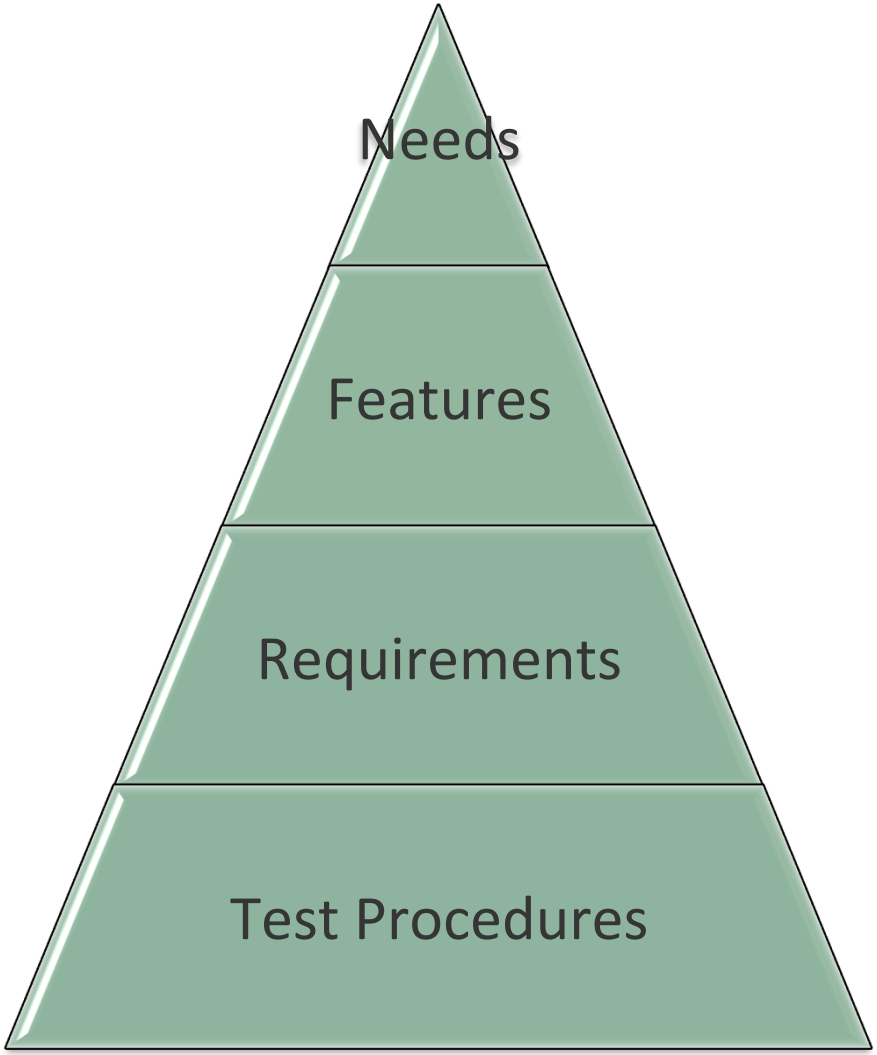
Questions/Comments

- What questions or comments do you have?

Course Agenda

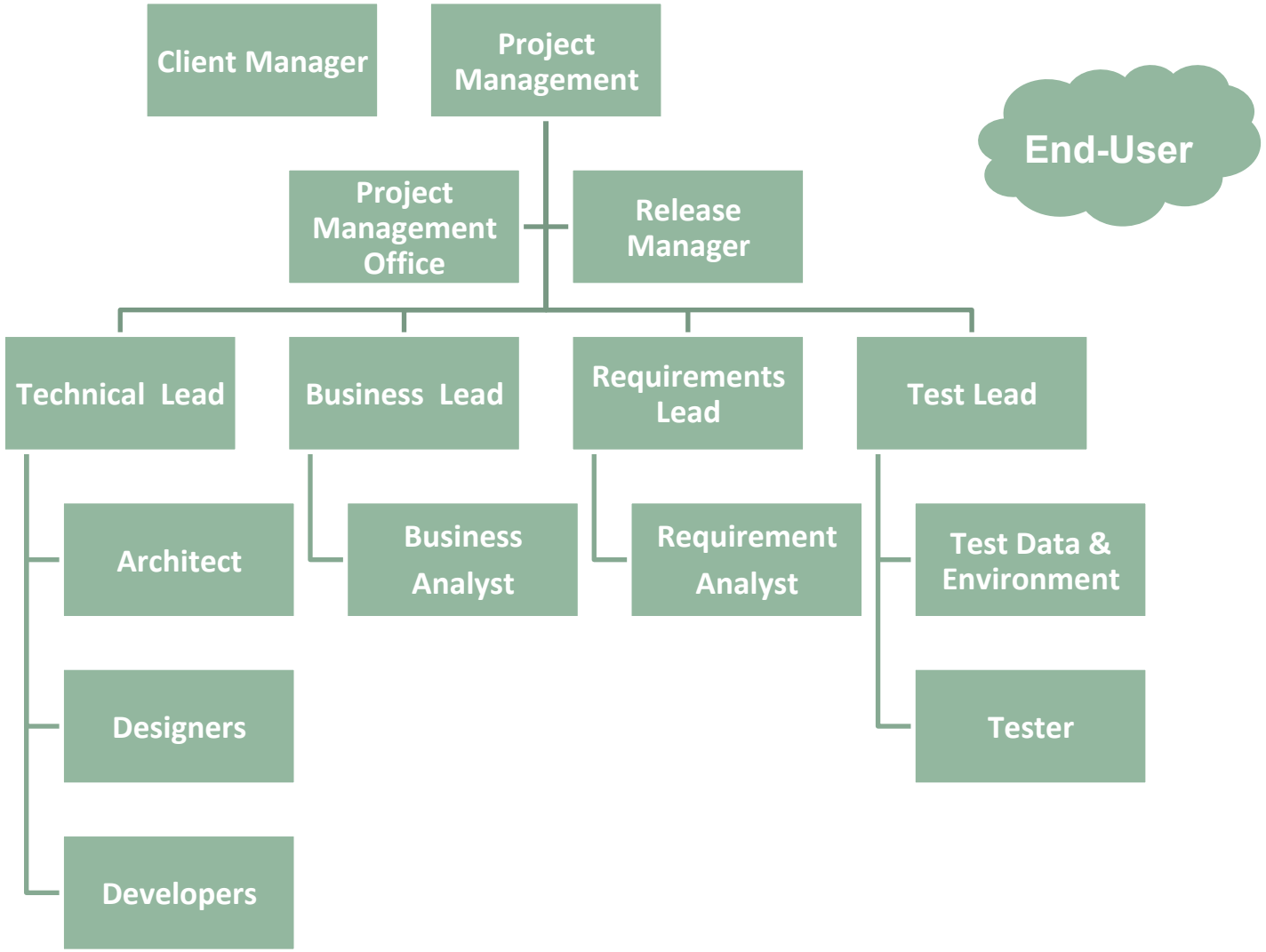
	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Why Projects?

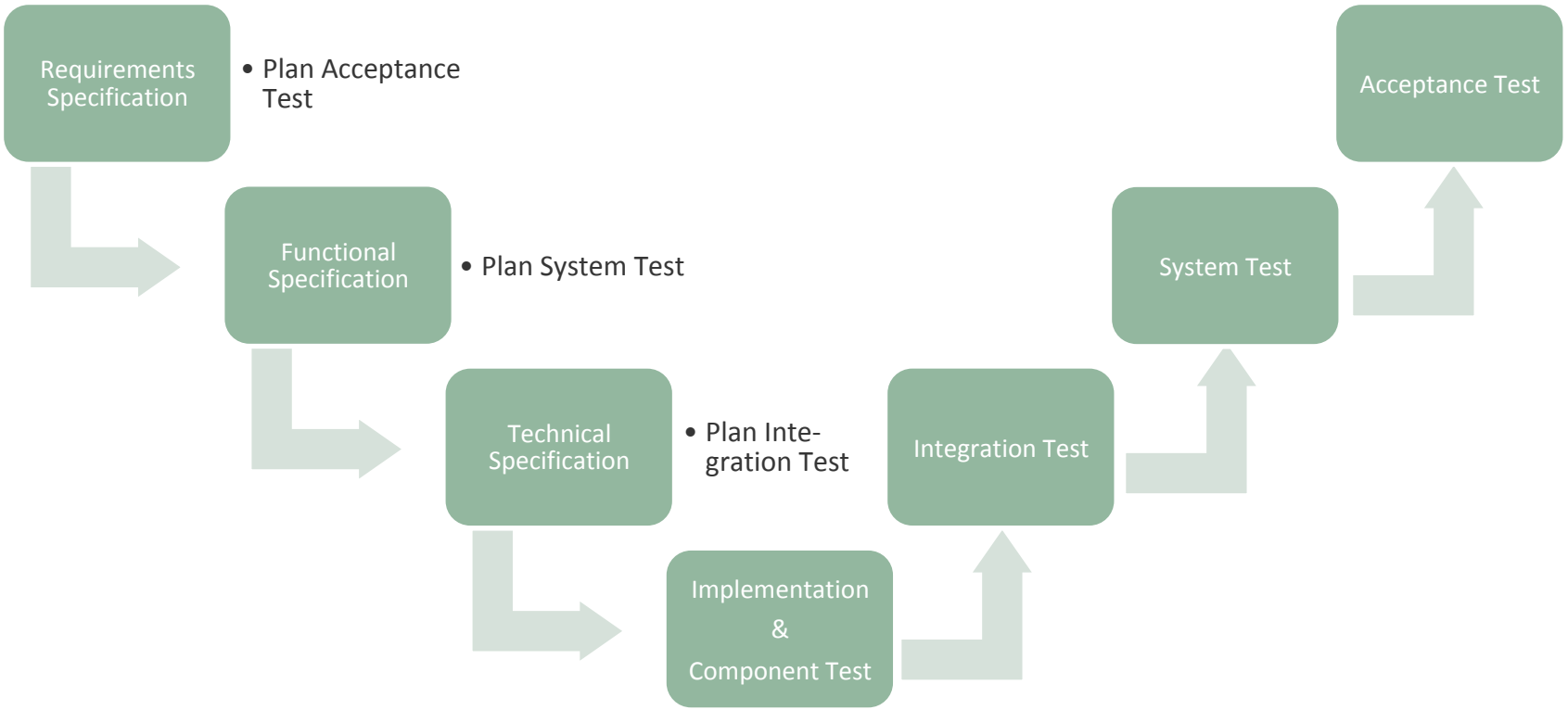


- Projects are temporary organizations for unique deliverables.
  - Problem Domain: The business need
  - Solution Domain: The features and the requirements
- Test Procedures must cover both domains

Test in a Project Organization



Test in a Project Lifecycle



Test Process

Planning
- Define objectives of testing
- Specify test activities

Analysis and Design
- Review and evaluate test basis
- Design and prioritze test cases

Implemen-tation and Execution
- Schedule test cases

Evaluation and Reporting
- Assess test against objectives
- Report test results

Closure Activities
- Consolidate test deliverables

Test Levels

- Component Test (Unit Test): Tests the individual units of an entire solution. The test ensures that the component (or module) is built and behaves as detailed in the specifications.
- Component integration test (String test): Tests the assembly and combined operation of related components. It ensures that the interactions between the components function correctly.
- Integration Test: An end-to-end test of the business requirements across all applications and platforms.
- System Integration Test: Tests the integrations with existing or 3rd party applications and platforms not in the scope of the project.
- System test (Product test): Tests that all functional and business requirements have been met by the system.
- Acceptance Test (UAT): Ensures that the users and stakeholders are satisfied with the solution.

Non-functional Tests

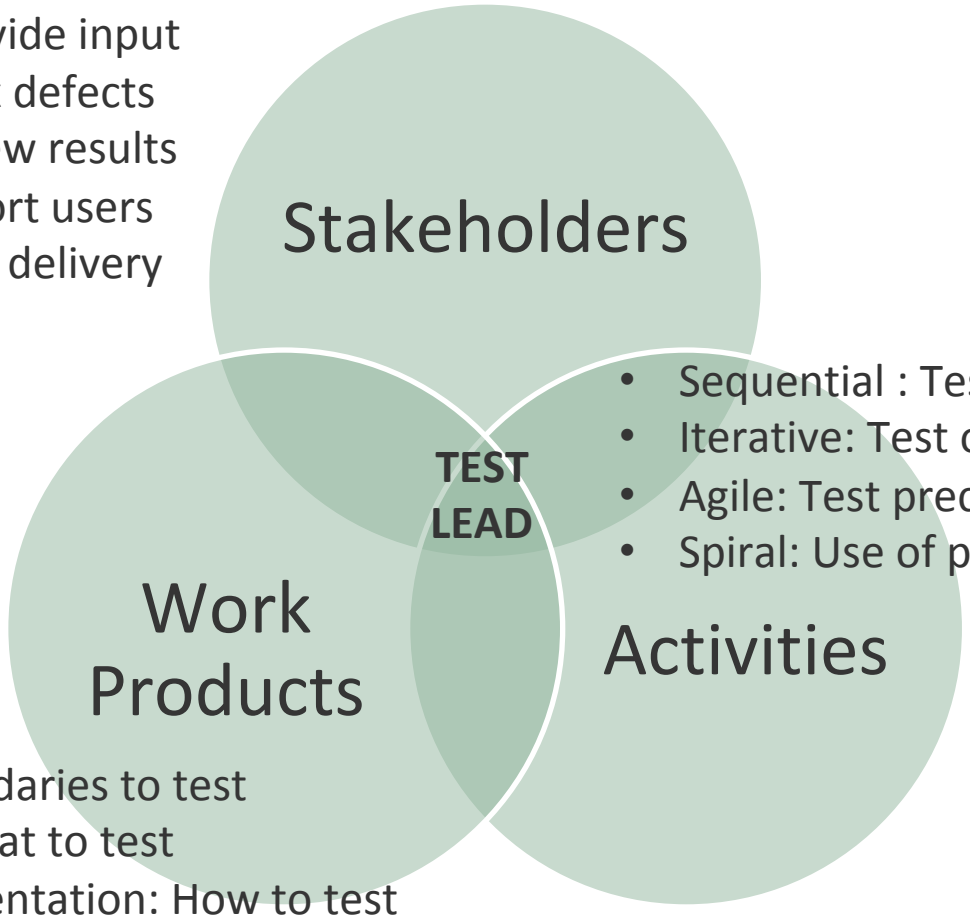
- Accuracy test: Tests application's adherence to specified or implied requirements.
- Suitability test: Evalueates and validates appropriateness of functions for specified tasks.
- Interoperability test: Tests whether an application functions in all targeted environments.
- Functional security test: Tests ability to prevent unauthorized access.
- Usability test: Measures suitability of application for its users.
- Accessibility test: Accessibility of software to those with particular requirements or restrictions.
- Technical security test: Prevent unintended use.
- Reliability test: Tests robustness and recoverability.
- Efficiency test: Tests performance, load, stress and scalability.
- Maintainability test: Tests the ease with which software can be maintained.
- Portability test: Tests ease with which software can be transferred.

Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Test Management Context

- Management: Provide resources
- Designers: Provide input
- Developers: Fix defects
- Business: Review results
- Support: Support users
- Users: Users of delivery



- Sequential : Test follows development
- Iterative: Test overlaps development
- Agile: Test preceeds development
- Spiral: Use of prototypes

- Project Plan: Boundaries to test
- Requirements: What to test
- Functional documentation: How to test
- Technical documentation: How to test
- Release Plan: When to test
- Defect Reports: Result of test

Test must be coordinated with the rest of the project

Test Management Context

- Test Objective: What is success?
- Test Content: What to test?
- Test Basis: What to test against?
- Entry and Exit Critera: When to start and end?
- Test Deliverables: What to deliver?
- Test Techniques: How to test?
- Test Metrics: How to measure?
- Test Tool: How to work with test?
- Resources: How to test and where to test?
- Stakeholders: Internal and External
- Compliance: Internal and External

Risk Management

Risk Identification

- Expert interviews
- Independent assessments
- Use of risk templates
- Lessons learned
- Risk workshops
- Brainstorming
- Checklists
- Past experience

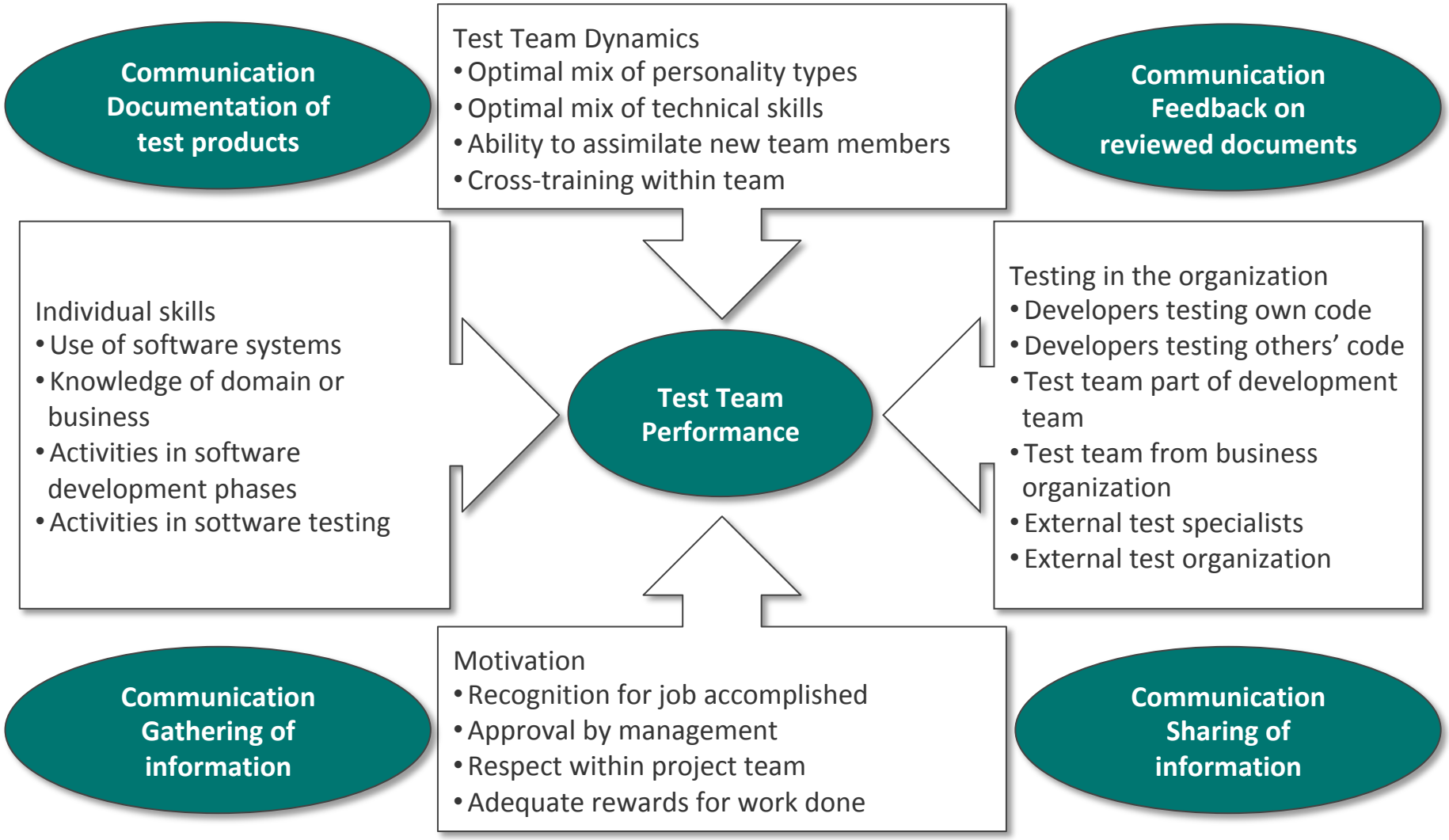
Risk Analysis

- Technical risks
- Business risks

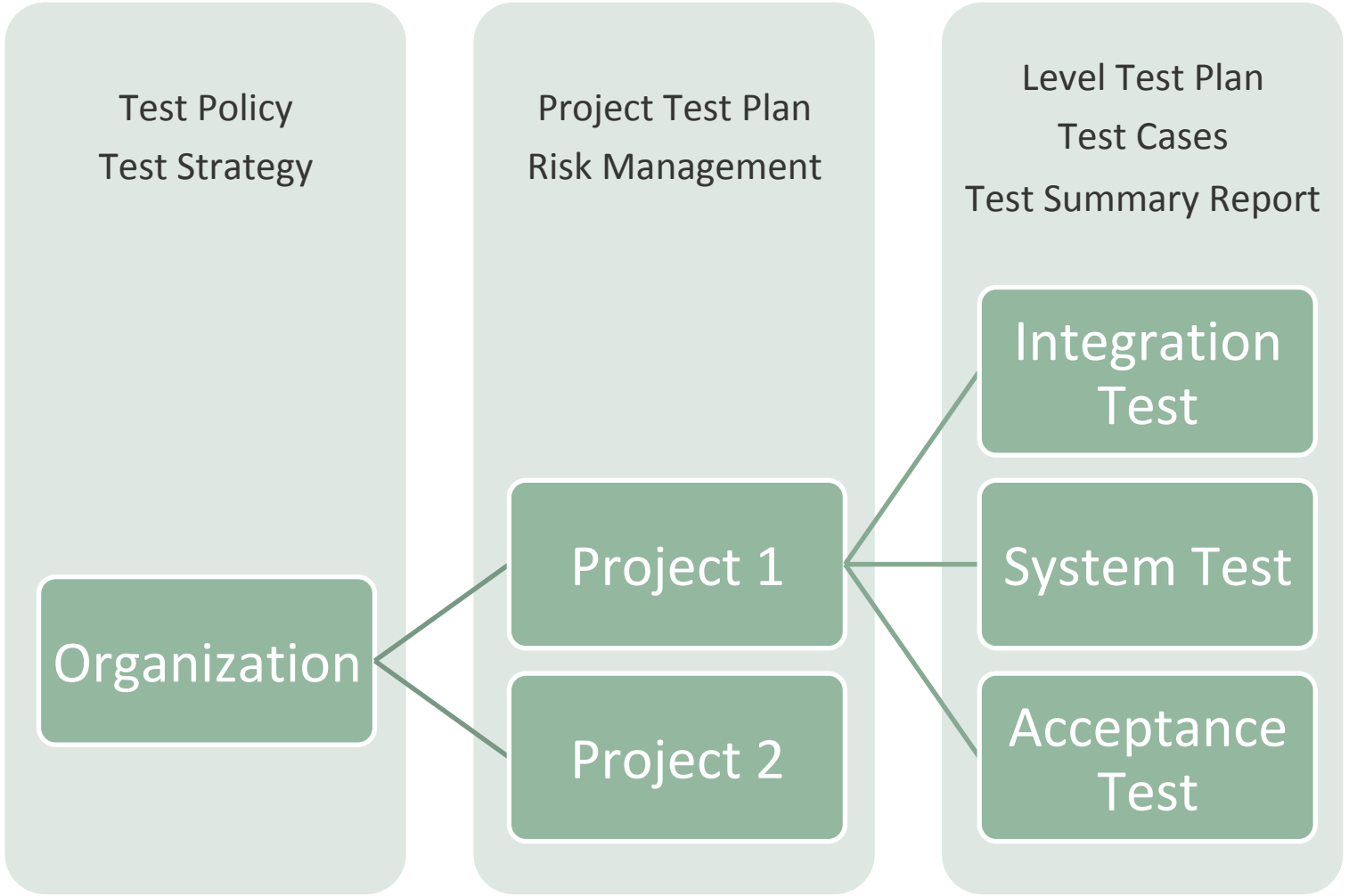
Risk Mitigation

- Handle the risk
  - Prevent
  - Reduce
  - Transfer
  - Ignore
- Project risk mitigation
  - Tools
  - Staff
  - Test Basis
  - Standards
- Product risk mitigation
  - Testing
  - Reviewing
  - Prioritizing

People Management



Test Deliverables



Test Strategy

- Test Methodology
- Integration Procedures
- Test Specification Techniques
- Independence of Testing
- Standards
- Test Environments
- Test Automation
- Test Tools
- Reusability of work products
- Retesting and Regression Testing
- Test Control and Reporting
- Test Measurements and Metrics
- Defect Management
- Configuration Management
- Roles and Responsibilities

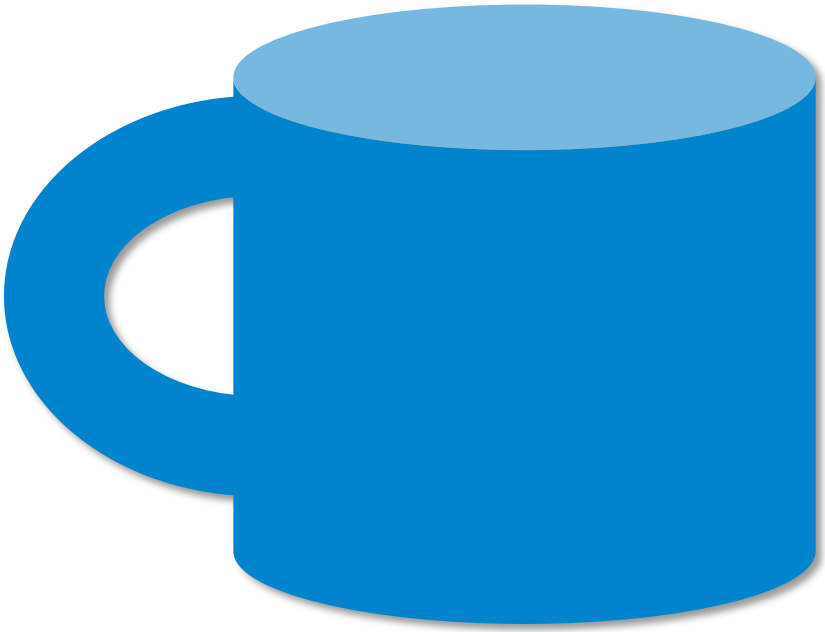
Level Test Plan

- Items to be tested and not to be tested
- Quality characteristics to be tested and not to be tested
- Testing Schedule and Budget
- Testing Execution Cycles
- Relationships and Deliverables
- Test Items in Scope and Out of Scope
- Entry and Exit Criteria
- Test Project Risks
- Overall test governance
- Responsibilities
- Outputs from previous test level and input from next test level

Test Plan (IEEE 829)

- Test plan identifier
- Introduction
- Test items
- Features to be tested
- Features not to be tested
- Approach
- Item pass/fail criteria
- Suspension criteria and resumption requirements
- Test deliverables
- Testing tasks
- Environmental needs
- Responsibilities
- Staffing and training needs
- Schedule
- Risks and contingencies
- Approvals

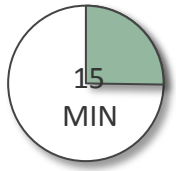
Coffee Break!



Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Case Work Introduction



- You will be working in a group, using any tools you want
- You have been assigned as test leads for a project
- You are expected to perform the following tasks:

  Gather information: 30 min

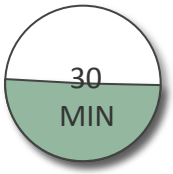
  1. Create Test Strategy: 1 hour
  2. Create Test Plan: 1 hour
  3. Requirements and Use Cases: 1 hour
  4. Design Test (Test Cases): 45 min
  5. Implement Test (Test Schedule): 45 min
  6. Execute 1st Delivery: 1 hour
  7. Report Status: 45 min
  8. Execute 2nd Delivery: 1 hour
  9. Execute 3rd Delivery: 45 min
  10. Close Test (Test Report): 60 min

- Each task will be structured as follows:
  - Short introduction
  - Preparing the task
  - Presenting the task (if applicable)

Case Work Introduction cont.

- The client is Tset Books, an established chain with 100 stores all over the Nordics
- Tset Books is starting an on-line book store and needs help with the following:
  - Business process development: Performed by client team
  - Development of E-trading site and Warehouse solution: Performed by 3rd party vendor
  - Integration test: Performed by 3rd party vendor
  - **System test of Warehouse solution: Performed by project team**
  - Acceptance test: Performed by client team
- The project team consists of one project manager (part time), and one test team (5 testers)
- You have no previous history with the client, nor with the 3rd party vendor
- You join the project in the Requirement Stage with no other information than the above mentioned

Case Work Introduction cont.



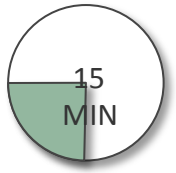
- In this task you will practice gathering information
- Inform yourself of the project
  - List the sources that you would like to obtain information from (project or client – use ISTQB for hints!)
  - Schedule 5 min interviews with the sources
  - Faculty will roleplay the interviews and hand out appropriate information sheets afterwards
  - Only 5 information sheets will be handed out and only one every 5 minutes so select and prioritize carefully!
- Use the information to prepare a test strategy in the next task

Course Agenda

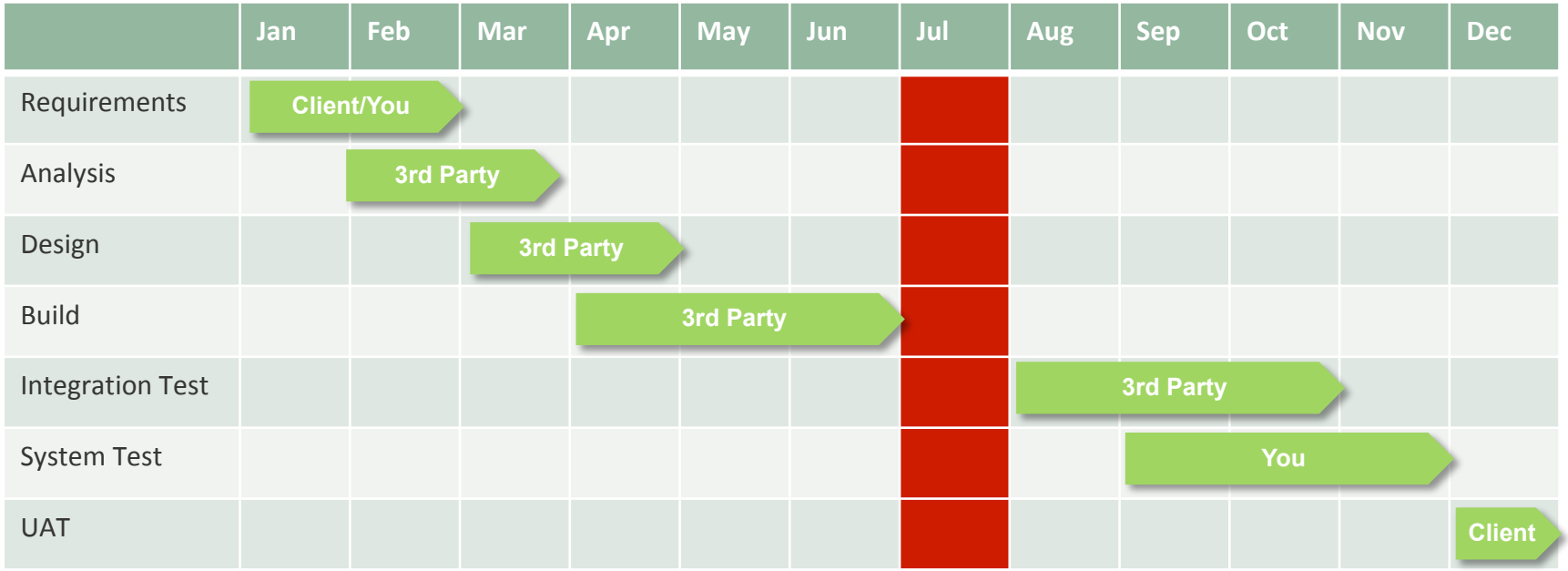
	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Case Work Task 1 – Create Test Strategy

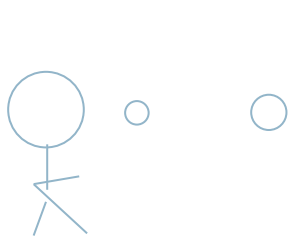
- Use the information from the previous task to prepare a test strategy
  - Focus on headlines and key bullets rather than full text
  - Focus on discussing and identifying key issues and risks
- Share your conclusions with the class
  - Which information did you obtain?
  - What is your overall test strategy?
  - Which additional information do you need?

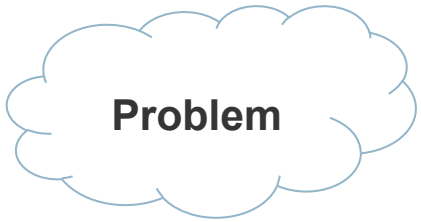


Case Work Task 1 – Project Plan (Project Manager etc.)



Case Work Task 1 – Problem Statement (Client Manager)

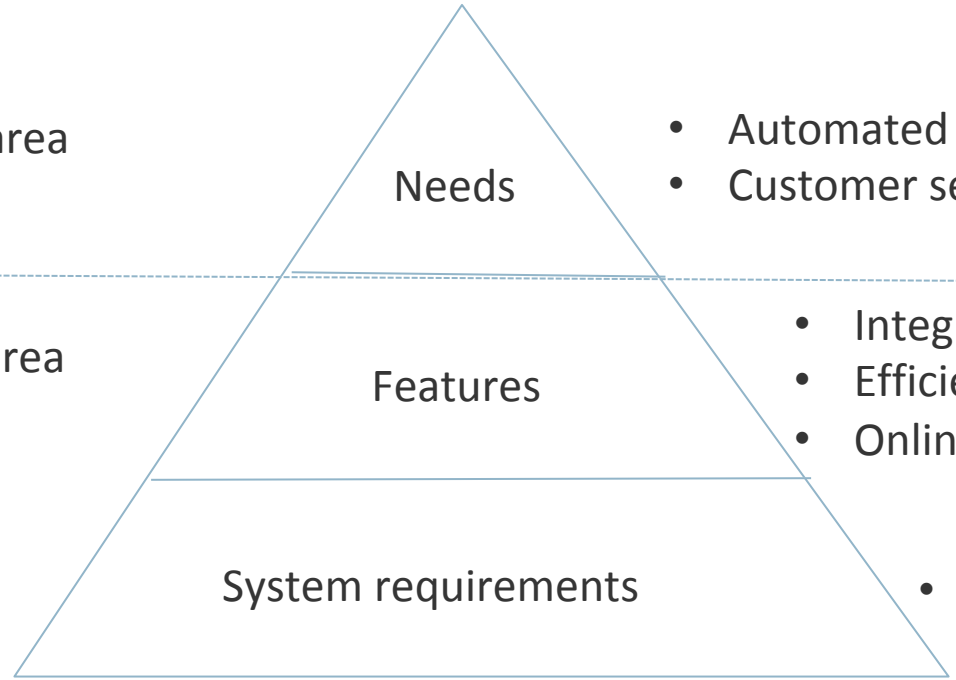




- Competition
- Manual workload

Problem area

Solution area

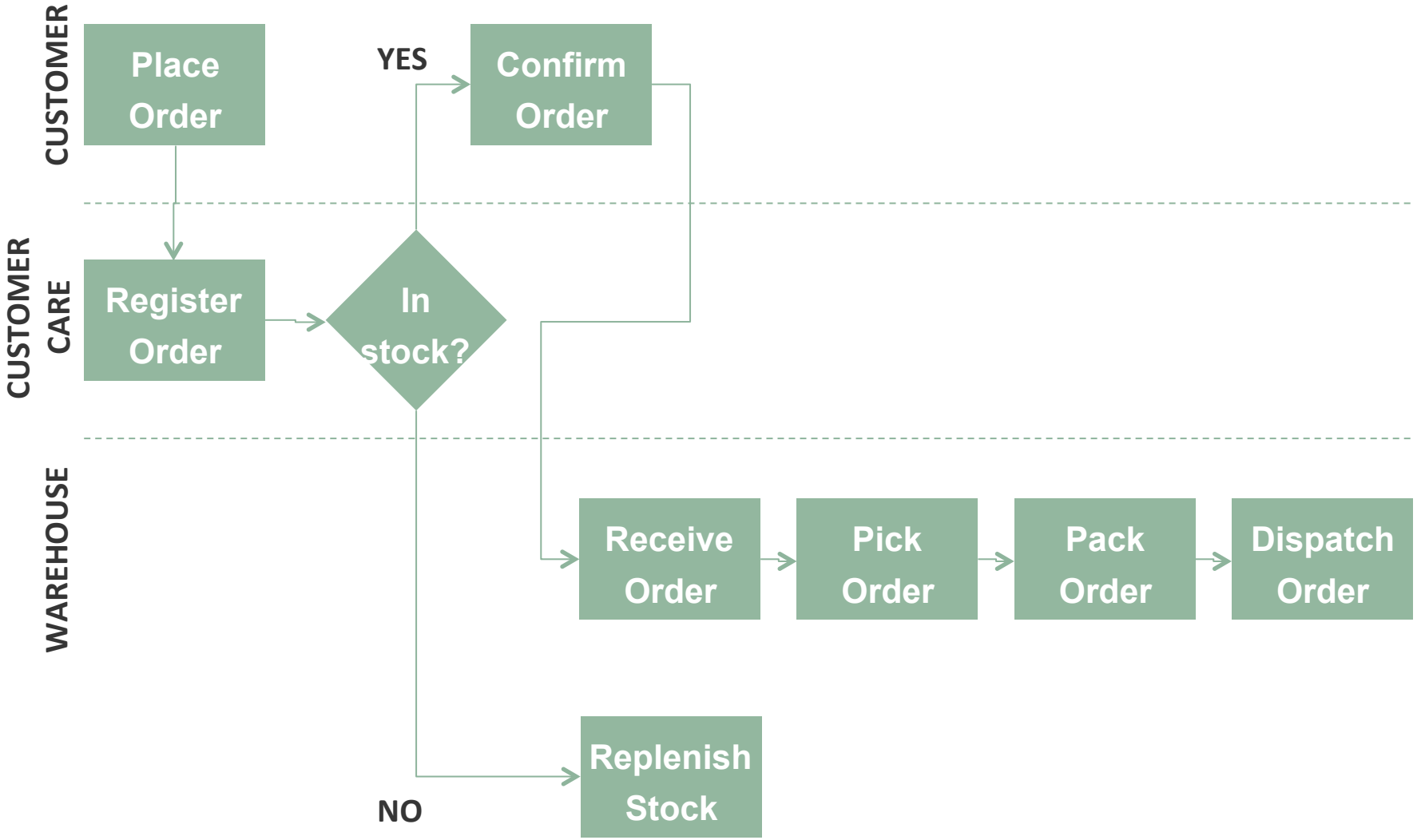


- Automated workflow
- Customer self-service

- Integrated order process
- Efficient warehouse management
- Online self-service

- To be detailed

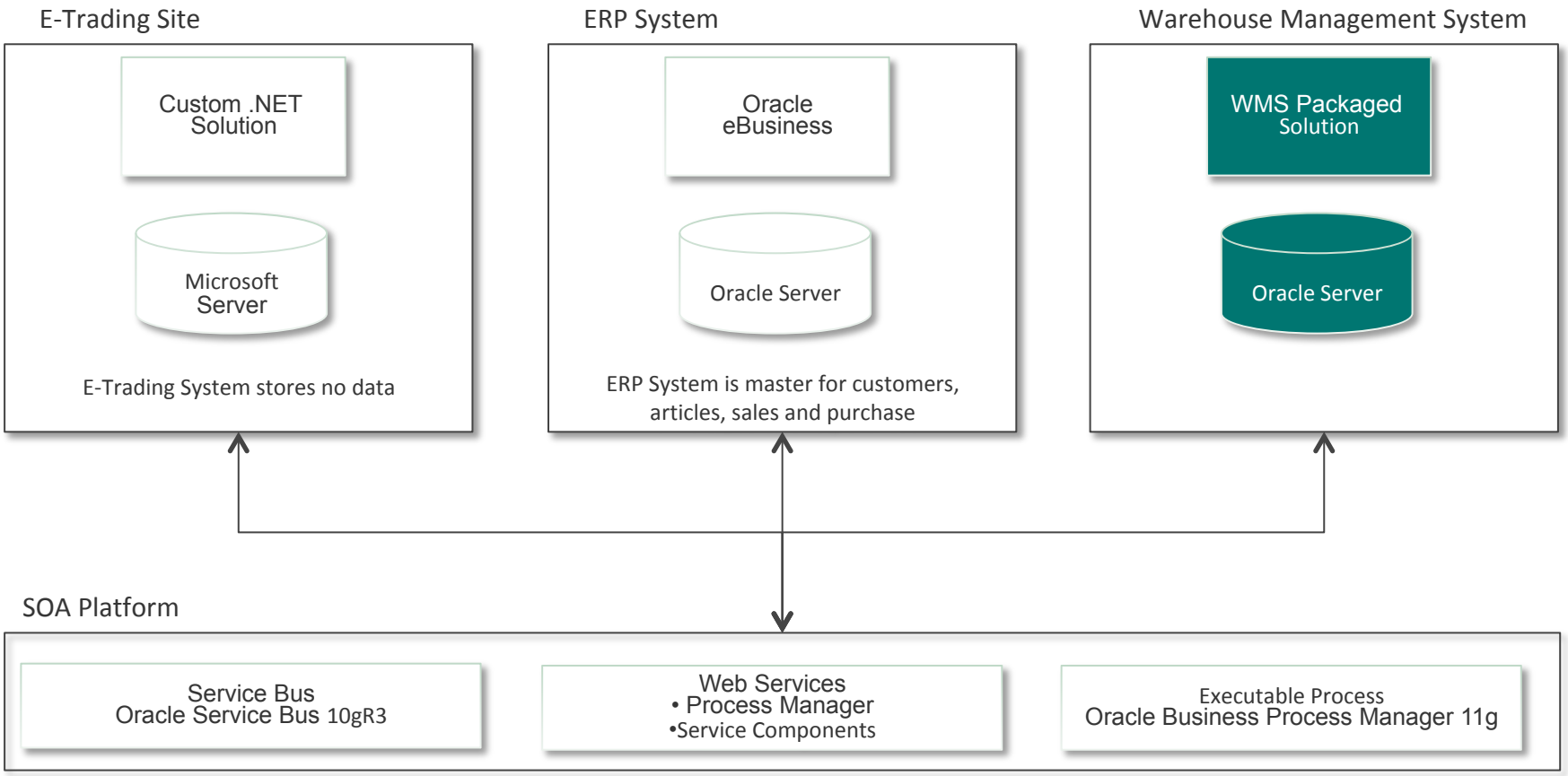
Case Work Task 1 – Business Process (End User etc.)



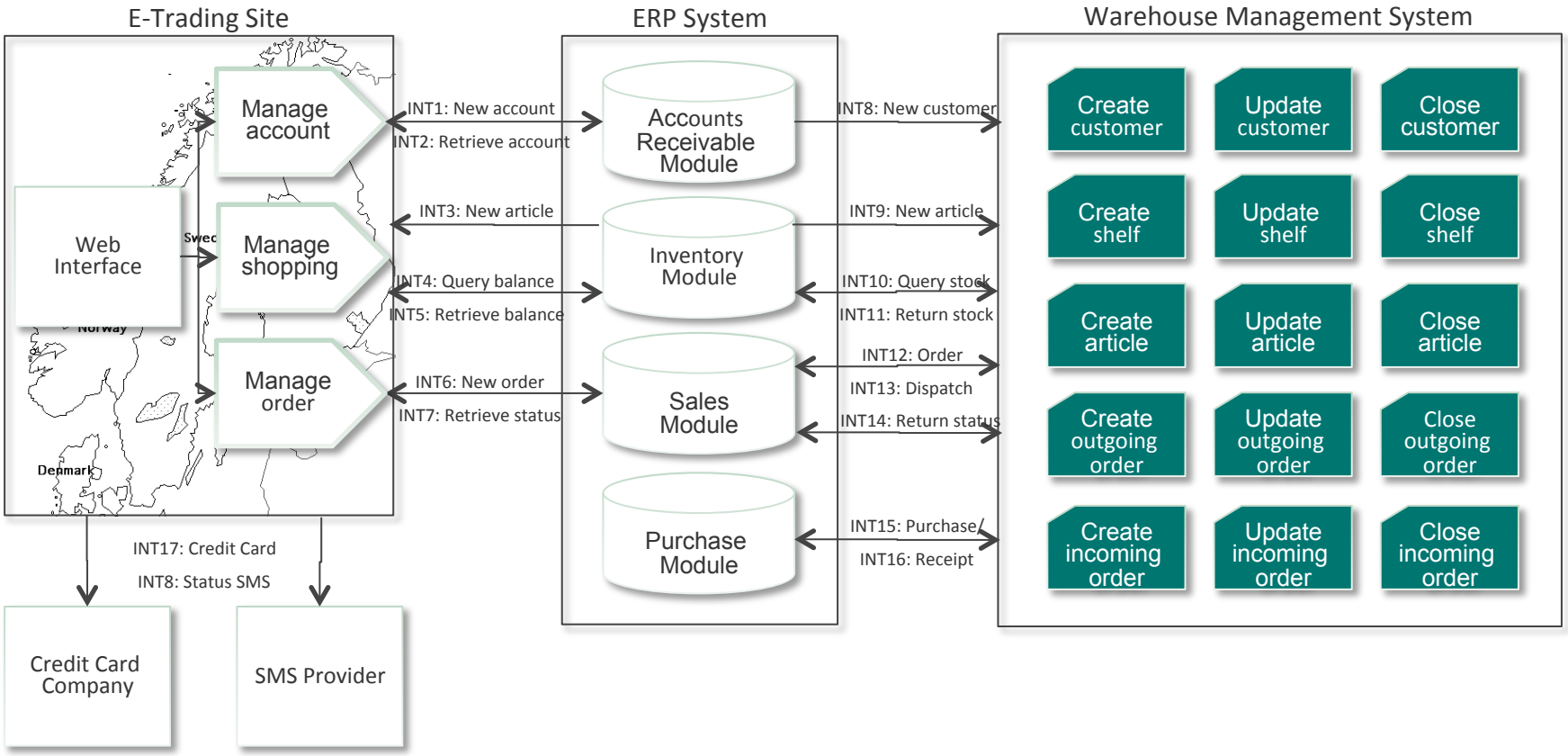
Case Work Task 1 – Business Requirements (Requirement Lead)

Id	Requirement Group
R1	The system must allow staff to set up shelves and connect articles to them
R2	The system must return stock levels by article
R3	The system must receive customer details
R4	The system must receive customer orders
R5	The system must support the physical pick, pack and dispatch of outgoing orders
R6	The system must return order status
R7	The system must support return orders
R8	The system must support the physical reception of return orders
R9	The system must support optimized inventory planning
R10	The system must receive incoming orders
R11	The system must support the physical reception of incoming orders
R12	The system must print invoices and reports

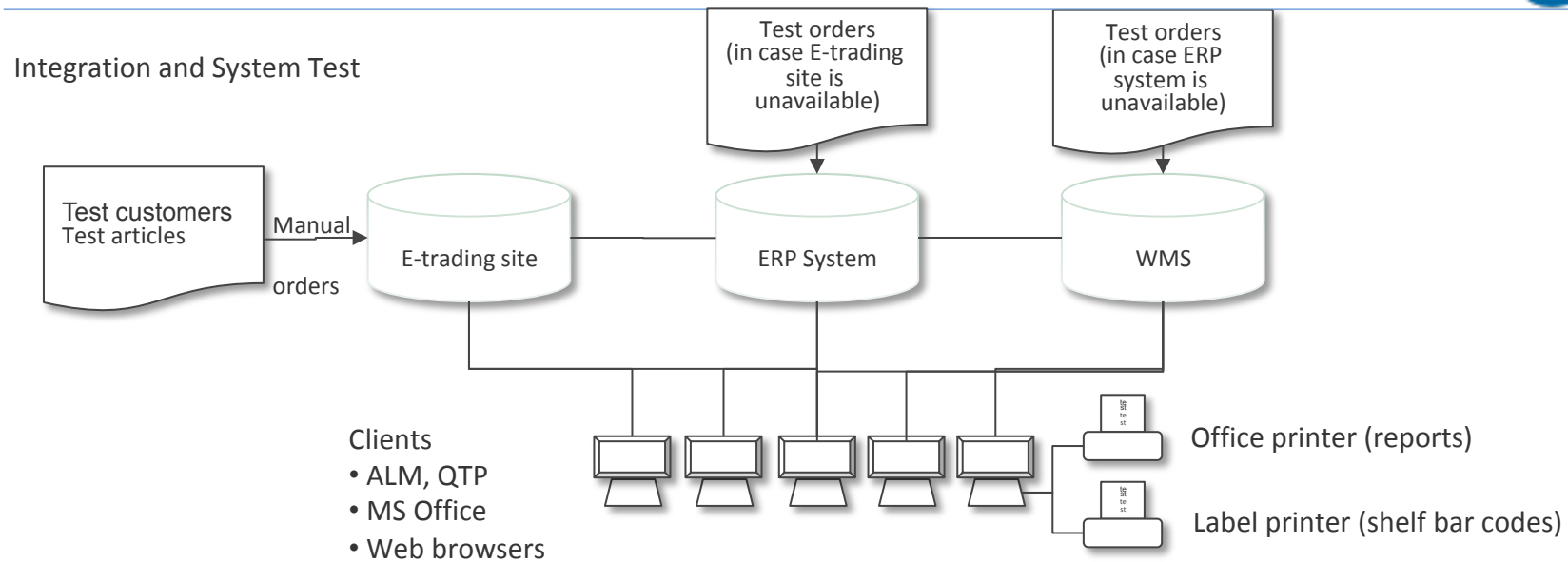
Case Work Task 1 – Technical Design (Architect etc.)

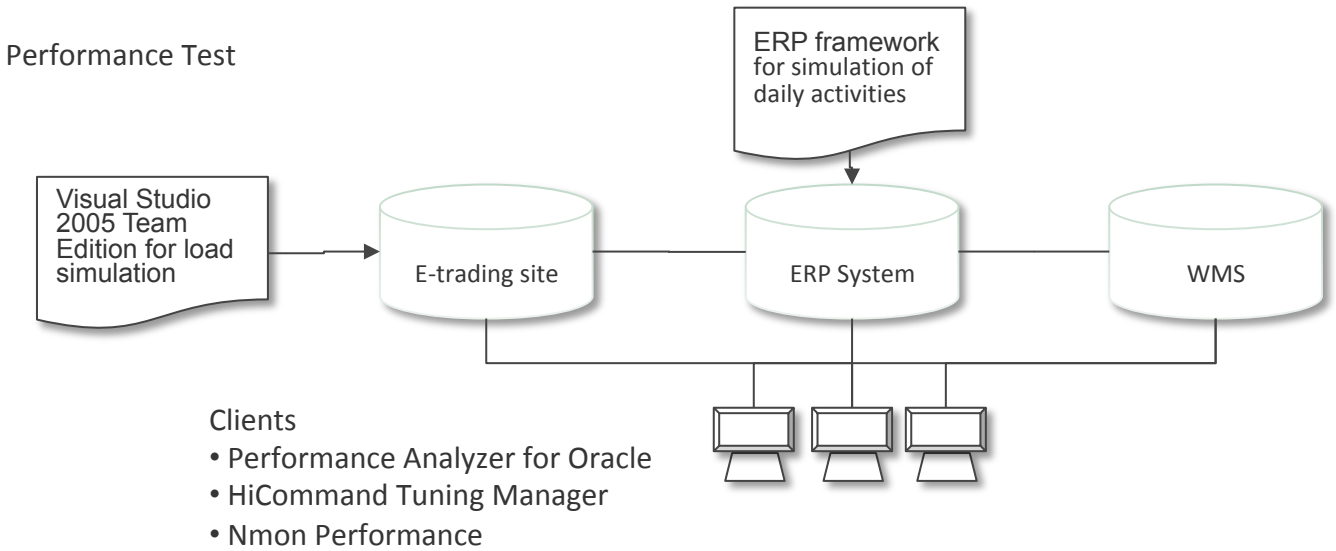


Case Work Task 1 – Functional Design (Designer etc.)

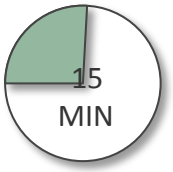


Case Work Task 1 – Test Environment (Test Data & Environment etc.)



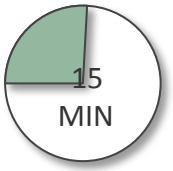


Case Work Task 1 – Sample Solution



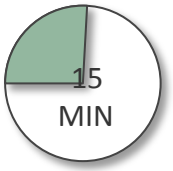
- Test Strategy Considerations
  - Test Methodology:
    - Client methodology should be used but ISTQB may still be used for guidance
    - Identify appropriate test stages
      - Identify integrations and systems for integration test
      - Identify peak volumes/users for performance test
      - Consider coordinating UAT with training of new users
      - Plan for security and usability test for WMS
      - Plan for embedded test for WMS (handheld devices etc.)
      - Plan for smoke test to verify that all systems are ready for test
    - Clarify boundaries between and responsibilities for test stages
  - Integration Procedures:
    - Identify release schedule and plan test cycles accordingly
  - Test Specification Techniques:
    - Specify dependencies, particularly on different teams
    - Advocate change control process
    - Prepare test tools for requirements traceability
  - Test Environments:
    - Include all integrated systems in test environment
    - Include not only systems but also hardware (handheld devices etc.)

Case Work Task 1 – Sample Solution cont.



- Test Strategy Considerations cont.
  - Test Automation:
    - Determine when the system is stable enough for automation
    - Determine need for emulating other systems
  - Confirmation Testing and Regression Testing:
    - Specify when to use emulated orders and when to create ”real” orders
    - Document steps for end-to-end tests
    - Document steps for regression tests
  - Test Control and Reporting:
    - Use estimator tools and peer reviews when estimating test activities
    - Check estimations against availability of test resources, environment etc.
    - Compare estimates with actuals and update estimates accordingly
  - Test Measurement and Metrics:
    - Propose suitable test metrics
    - Agree with the client which metrics to use and how to report them
    - Prepare test tools for capturing test metrics

Case Work Task 1 – Sample Solution cont.



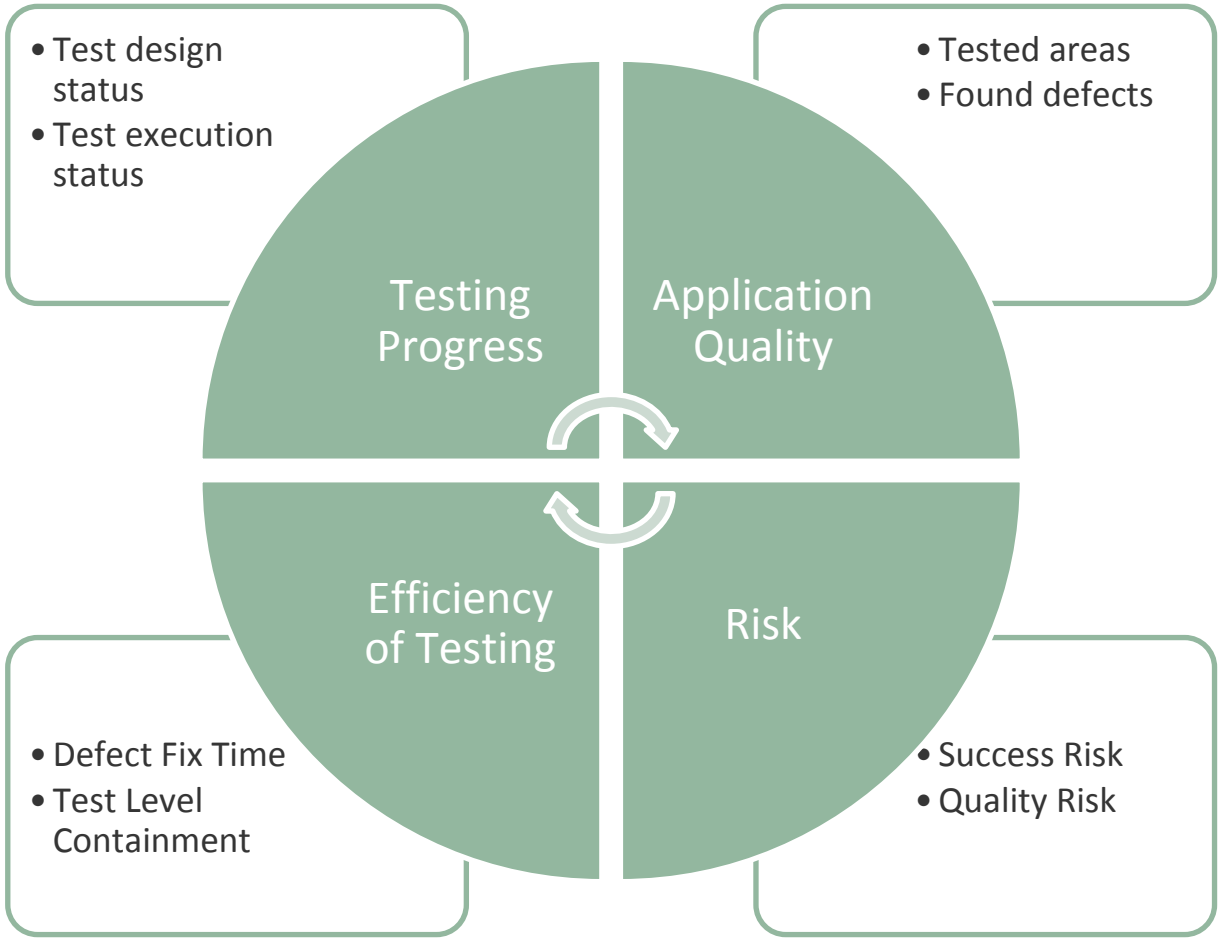
- Test Strategy Considerations cont.
  - Defect Management:
    - Clarify communication paths, particularly between different teams
    - Clarify responsibilities
    - Plan for defect meetings to avoid defects falling between chairs
    - Take into account fix times and retest in plan
  - Configuration Management:
    - Ensure coordinated code migrations to all systems
    - Identify lead times for migrations
    - Plan for smoke test and regression test
  - Roles and Responsibilities:
    - Create responsibility matrix and team interaction chart, paying special attention to interaction with external resources
    - Ensure governance from program and/or project management
    - Specify timelines, milestones and transition points between teams

Lunch!

Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

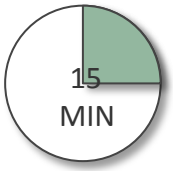
Test Metrics



Test Metrics Definition and Purpose

	Metric	Definition	Purpose
Test Progress	Actual vs Planned Test Cases Designed	Designed Test Cases / Planned Test Cases	Track test case design progress
	Actual vs Planned Test Cases Executed	Executed Test Cases / Planned Test Cases	Track test case execution progress
System Quality	Passed vs Planned Test Cases	Passed Test Cases / Planned Test Cases	Track overall quality of solution
	Failed vs Planned Test Cases	Failed Test Cases / Planned Test Cases	
	Passed vs Executed Test Cases	Passed Test Cases / Executed Test Cases	Assess how many of the defects have been found and how many remain to be found
	Failed vs Executed Test Cases	Failed Test Cases / Executed Test Cases	
Risks	Blocked test cases by area	Blocked Test Cases / Planned Test Cases	Assess which areas can be tested and not
	Failed test cases by area	Failed Test Cases / Planned Test Cases	Assess which areas have an acceptable quality and not
	Defects found by area	Number of defects found in area / Total number of defects found	Assess which areas are more prone to defects than others
Test efficiency	Defects found by test level	Number of defects found in test level / Total number of defects found	Assess whether defects are found early
	Defect leakage	Number of defects found at current level / Number of defects found at previous level	Assess testing in previous levels

Test Metrics Activity



- Work in pairs
- You will be given a some test statistics
- Calculate the expected test metrics
- Compare your results with the faculty
  - Was additional statistics required for any metrics?
  - Do the metrics give a fair picture of the test progress?

Test Metrics: Input Values

Input values	Pass 1 (1st week)	Pass 2 (2nd week)	Pass 3 (3rd week)
Total number of test cases	200	200	200
• Of which designed	190	195	200
• Of which executed	140	170	200
• Of which passed	100	160	195
• Of which failed	40	10	5
• Of which not executed	60	30	0
Total number of test cases Area A	40	40	40
• Of which blocked	10	5	0
• Of which failed	7	3	0
Total number of defects	40	50	55
Total number of defects Area A	9	17	32
Total number of defects in previous level			400

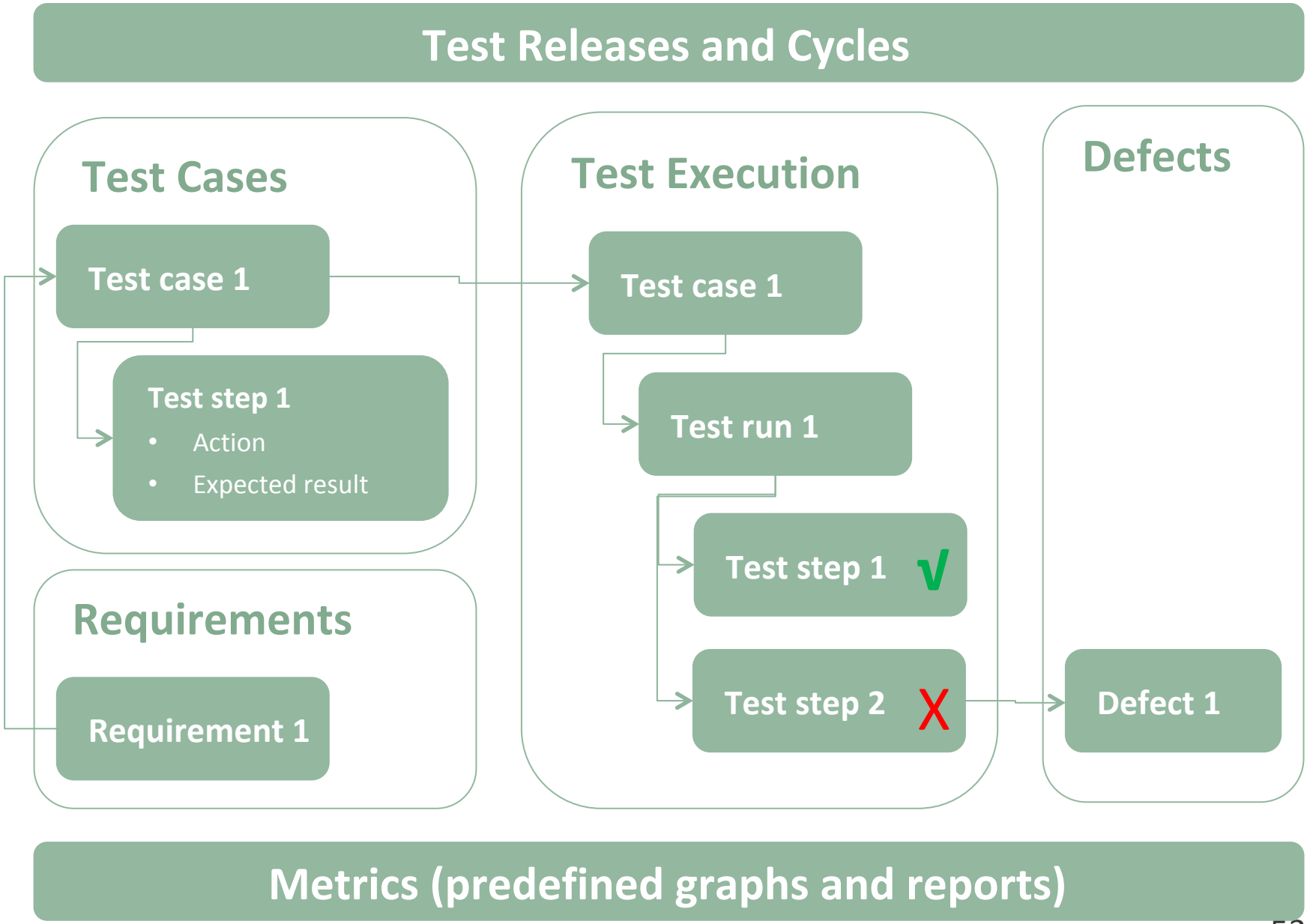
Test Metrics: Output Fields

Metrics	Pass 1 (1st week)	Pass 2 (2nd week)	Pass 3 (3rd week)
Actual vs Planned Test Cases Designed			
Actual vs Planned Test Cases Executed			
Passed vs Planned Test Cases			
Failed vs Planned Test Cases			
Passed vs Executed Test Cases			
Failed vs Executed Test Cases			
Blocked test cases by area			
Failed test cases by area			
Defects found by area			
Defects found by test level			
Defect leakage			

Test Metrics: Solution

Output values	Pass 1 (1st week)	Pass 2 (2nd week)	Pass 3 (3rd week)
Actual vs Planned Test Cases Designed	95%	98%	100%
Actual vs Planned Test Cases Executed	70%	85%	100%
Passed vs Planned Test Cases	50%	80%	98%
Failed vs Planned Test Cases	20%	5%	2%
Passed vs Executed Test Cases	71%	94%	98%
Failed vs Executed Test Cases	29%	6%	2%
Blocked test cases by area	25%	13%	0%
Failed test cases by area	18%	8%	0%
Defects found by area	23%	34%	58%
Defects found by test level	9%	11%	12%
Defect leakage	10%	13%	14%

Test Tools support Test Management

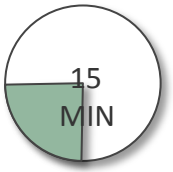


Course Agenda

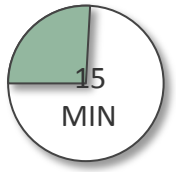
	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Case Work Task 2 –Plan Test

- **In this task you will practice preparing a test plan**
- **Build on the information from the Case Work Introduction**
- **Use the information to prepare a test plan for the system test**
  - Use the IEEE template
  - Focus on headlines and key bullet points rather than full text
- **Share your conclusions with the class**
  - Present your test plan
  - Which assumptions/considerations is it based on?
  - Which risks and issues do you see?

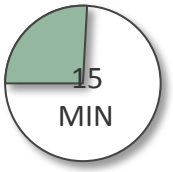


Case Work Task 2 – Sample Solution



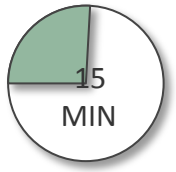
- Test Items, Features to be tested:
  - The objective is to test the WMS, both internal functions and end-to-end order processes
  - The test will be performed against requirements and use cases, tracked by a requirements traceability matrix
- Test Approach:
  - Refer to Test Strategy for Change Requests and Defect Management
  - Clarify responsibilities in RACI matrix and interaction in interaction chart
  - Severity 1 defects should be fixed in 1 day, severity 2 in 3 days and severity 3 in 5 days
  - Status should be reported after each delivery
- Item Pass/Fail Criteria:
  - Entry criteria:
    - Frozen requirements
    - Test plan, test cycles and test cases completed
    - Closed integration test
    - Prepared test environment
  - Exit critera:
    - No open defects of severity 1, workaround for open defects of severity 2, action plan for all open defects
    - Review meeting and test report completed

Case Work Task 2 – Sample Solution



- Test Deliverables:
  - Test Plan, Test Schedule and Test Cases
  - Test Report to be delivered
  - TestMetrics:
    - Provide planned rates for all metrics specified in the Test Strategy
    - Explain how to use metrics and whom to report to
- Testing Tasks:
  - Specify test management tool (i.e. HP ALM)
  - Specify how to use the test management tool
    - Links to requirements
    - Structure of test scripts
    - Templates for test scripts
    - Rules for logging defects
  - Specify test automation tool (i.e. QTP)
- Environmental Needs:
  - Specify need for systems (E-Trading Site, ERP and WMS)
  - Specify need for hardware (desktops, handheld devices etc.)
  - Specify rules for sharing test environment with other tests (i.e. integration test)
  - Specify how to verify the test environment (i.e. smoke test)

Case Work Task 2 – Sample Solution

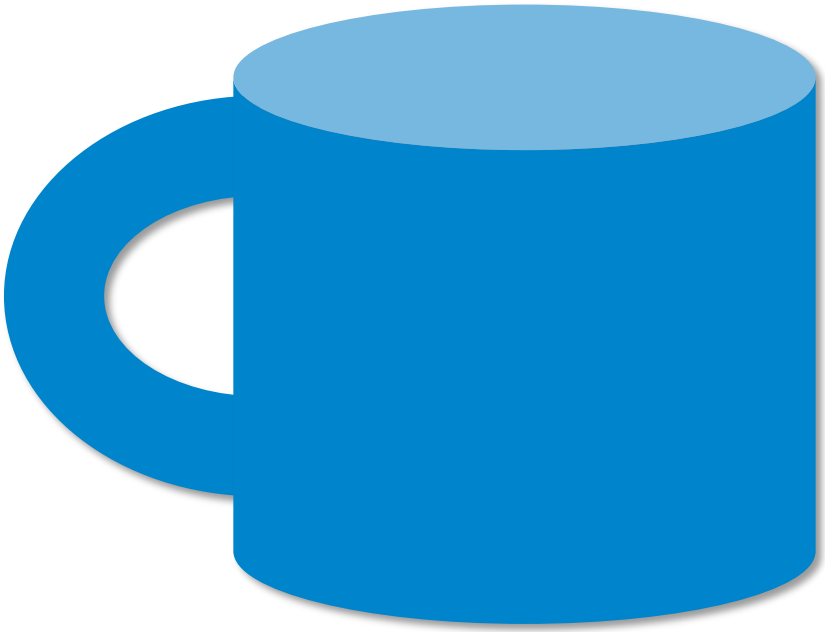


- Responsibililties:
  - Refer to Test Strategy for Configuration Management
  - Explain how to interact with Configuration Management
- Staffing and Training Needs:
  - List names,scheduled times and scheduled activities of testers
  - List names and expected need of support resources (super users for other systems to create/ verify test data, test environment support etc.)
  - List training in test tools, WMS and other systems
- Test Schedule:
  - Delivery 1 to be tested in September
  - Delivery 2 to be tested in October
  - Delivery 3 to be tested in November
- Risks and Contingencies:
  - Assumptions: List input from rest of project
  - Issues: List issues, such as requirements not being frozen yet
  - Risks: List risks, such as shared test environment
  - Contingencies: List contingencies, such as coordinating testing with integration test team

Case Work Task 2 – Test Plan Check List

	Team 1	Team 2	Team 3	Team 4
1. Test Items, Features to be tested/not tested	Comm Not ERP, e-	WMS,Int		
2. Test Approach	High prio	Tech, showst		
3. Pass/Fail Criteria, Suspension/ Resumption Critera	Ent, Exit formalize d	Env, tools, data, code,cov		
4. Test Deliverables	Plan, Doc	Case,des,		
5. Testing Tasks		Sta, dyn		
6. Environmental Needs	Comp	WMS, Tool? Data		
7. Responsibilities		Proj roles		
8. Staffing and Training Needs	Proj tr	Week		
	3 mon.	Des. Sep		

Coffee Break!



Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

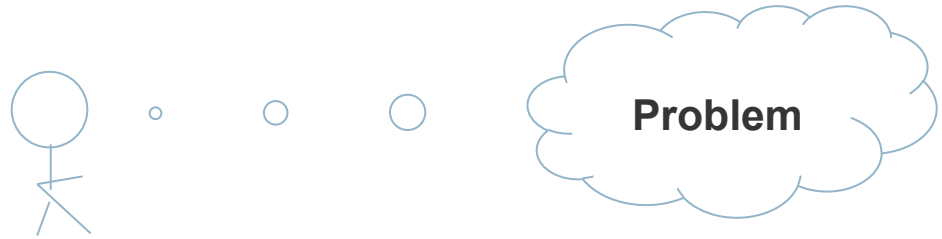
Requirement Definition

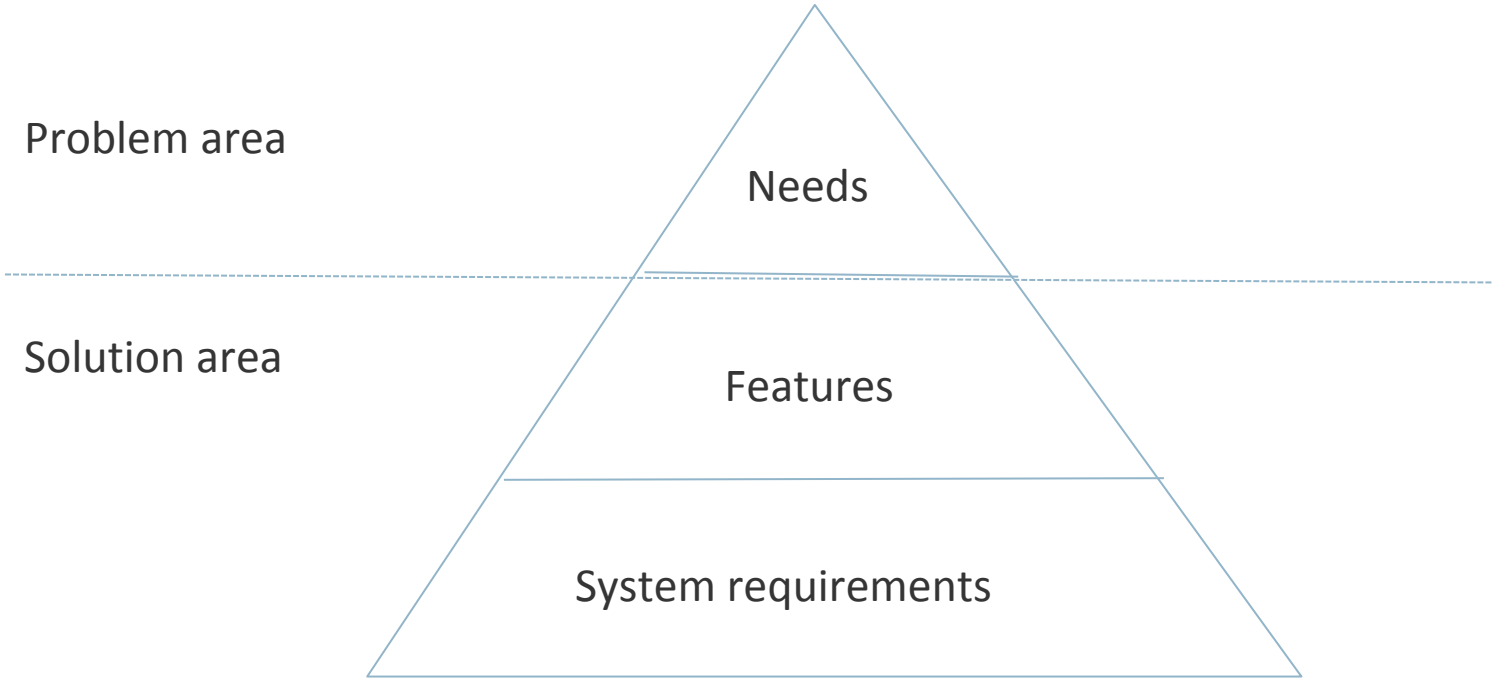
A requirement is defined as "a condition or capability to which a system must conform".

It can be:

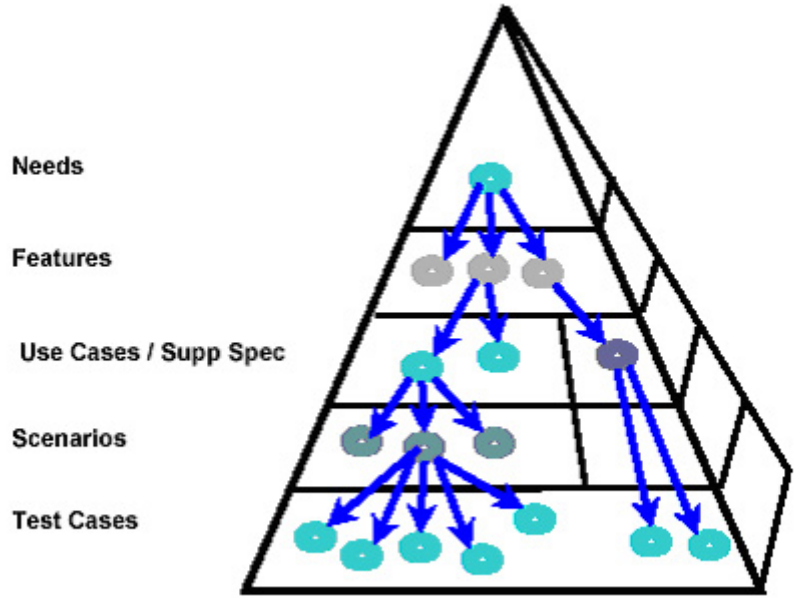
- A capability needed by a customer or user to solve a problem or achieve an objective
- A capability that must be met or possessed by a system to satisfy a contract, standard, specification, regulation, or other formally imposed document
- A restriction imposed by a stakeholder

Requirements Pyramid





Traceability Requirements Pyramid



Traceability plays several important roles:

- Verify that an implementation fulfills all requirements
  - Everything that the customer requested was implemented
- Verify that the application does only what was requested
  - Don't implement something that the customer never asked for
- Help with change management
  - When some requirements change, we want to know which test cases should
  - be redone to test this change

Where do we find the Requirements?

In different documents

Threat

Test reports

Environment

The users/Business

Processes/activities

Helpdesk/ bug reports

Existing hardware/applications

Notes

What is it?

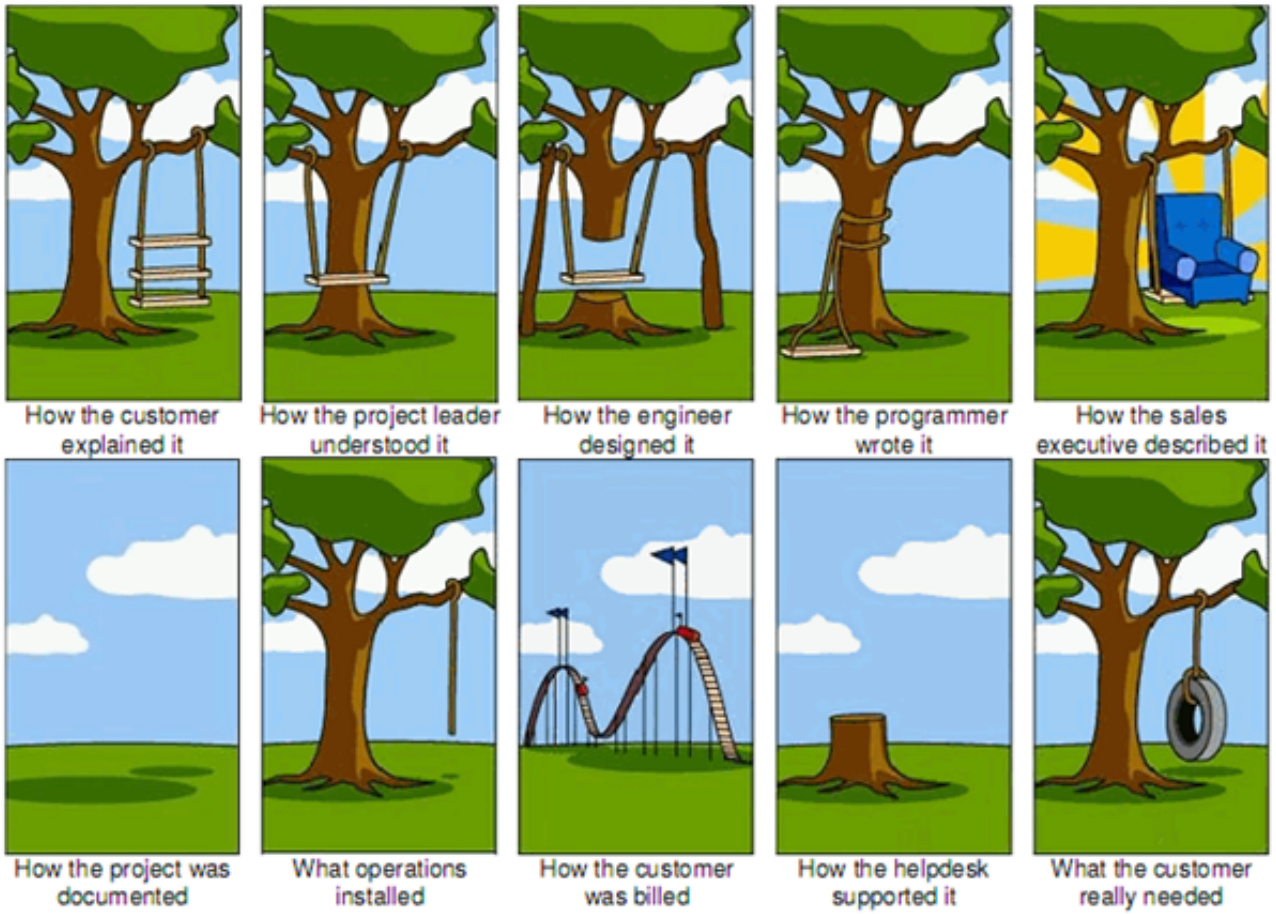
- Some creativity is needed…

Requirements:

1. Suspended with multiple ropes
2. Shall be able to bear load
3. Can be moved freely
4. Made of wood
5. Weatherproof

- Interpret the requirements

Fuzzy requirements => interpretations

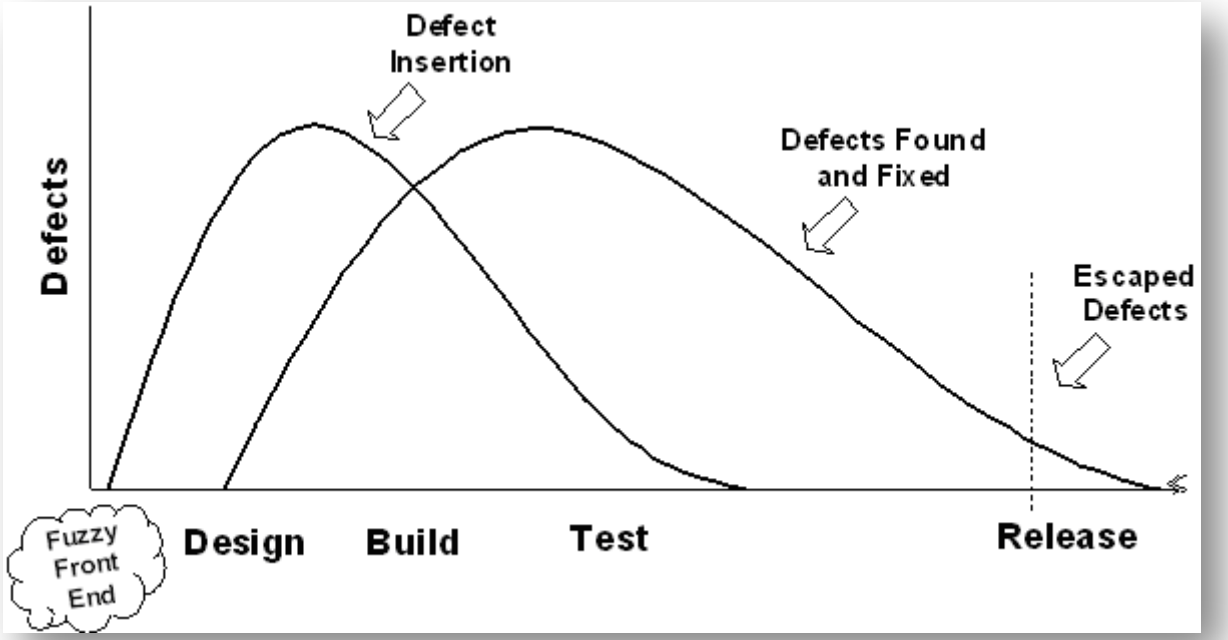


The challenge to any software development project

- **How to remove defects introduced?**
  - Early
  - Quickly
  - Cheaply
- **How to lower the risks they impose on the functioning of the product?**
  - Review and test requirements very early-on in the project lifecycle

Time, Cost and Quality

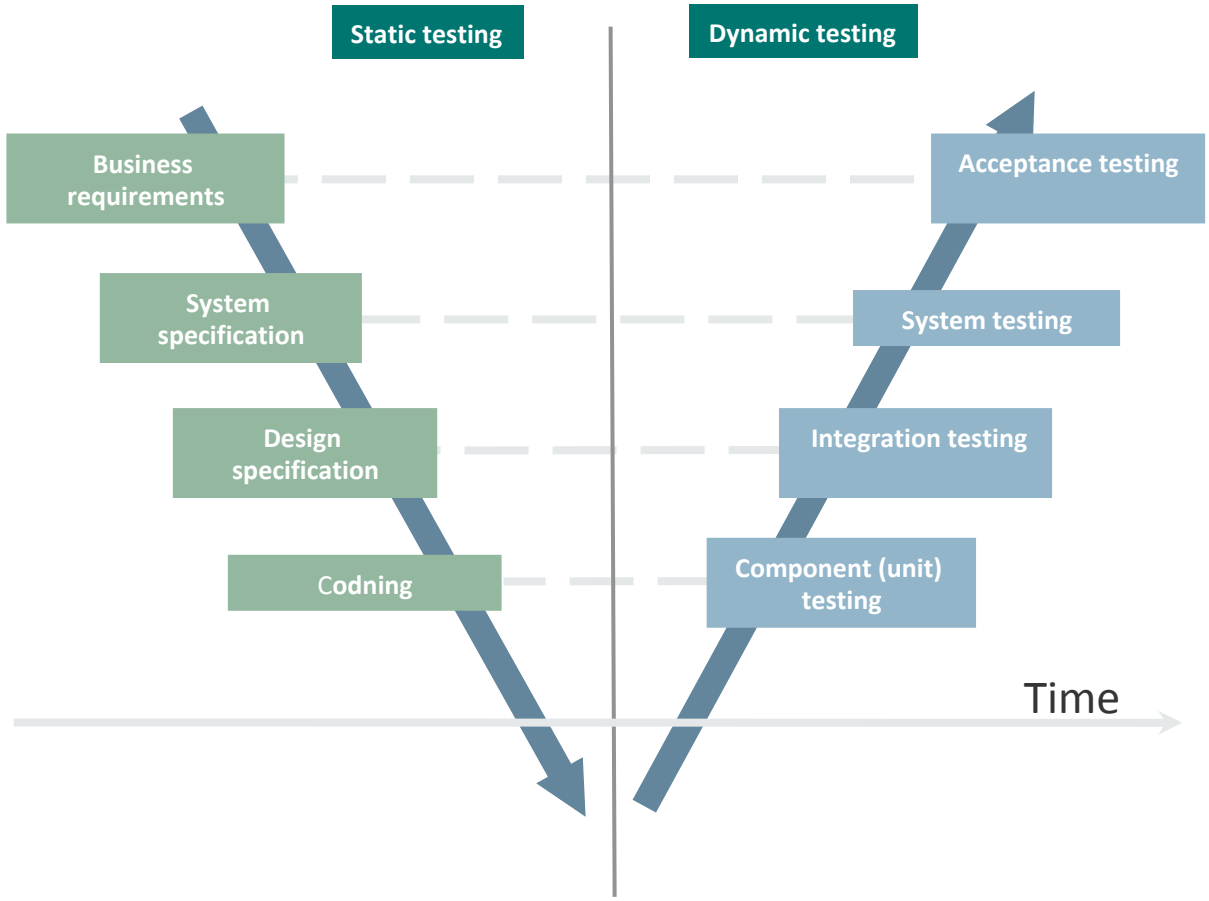
Defect Insertion and Find-and-Fix Dynamics



1) Source: Six Sigma Software Metrics Part 2 by David L. Hallowell

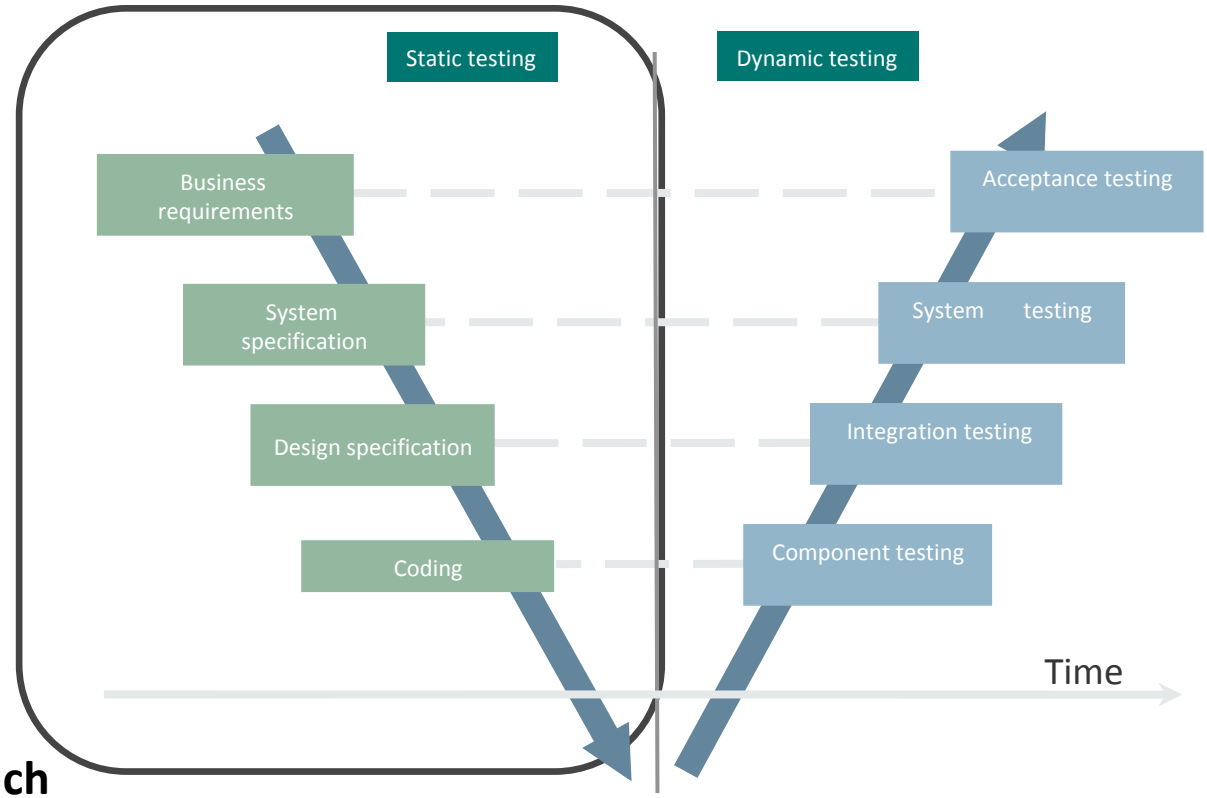
The V-model including Static and Dynamic testing

- Static testing
  - Review
  - Static code analysis
- Dynamic testing
  - Execution of software



Static testing

- **Review techniques**
  - Informal
  - Walkthrough
  - Technical
  - Inspection
- **Static analysis**
  - Code
  - Automated by tools
- **Requirements testing**
- **Early test design approach**



What is a good requirement? – Briefly

- A good requirement expresses something that is:
  - Necessary
  - Verifiable
  - Achievable
  - Clearly described

- Can also be expressed as:
  - **S**pecific
  - **M**easurable
  - **A**ttainable
  - **R**ealistic
  - **T**estable

Characteristics of good requirements

- IEEE standard 830 states the following characteristics for good requirements:
  - Correct
  - Unambiguous
  - Complete
  - Consistent
  - Verifiable
  - Modifiable
  - Traceable

In the next slides examples of good and bad will be shown

Examples of wrong and correct requirements

Correctness (Adequacy)

Wrong

- The system shall store user information in a database

Correct

- The system shall store user information in a retrievable and updateable manner

Checks

- Is the requirement within the scope of the release?
- Is the requirement feasible from a technology point of view?
- Does the requirement refrain from being unnecessarily over-stringent?
- Is it a requirement (what) and not a description of implementation (how)?
- Is it a requirement and not a statement or a goal?
  - Requirements use shall or must
  - Statements of fact use will
  - Goals use should

Examples of wrong and correct requirements, cont.

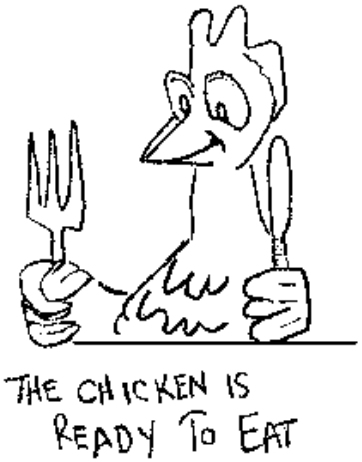
Non-Ambiguity

Wrong

- The system shall be user-friendly

Correct

- The system's front end shall be developed in accordance with the Microsoft Windows Logo GUI guidelines



Checks

- Does the requirement refrain from using words and phrases open to misinterpretation?
  - minimise, maximise, user-friendly, rapid, fast, quick, easy, sufficient, adequate
- Does the requirement refrain from using non-specific phrases referring to time?
  - after, before, until, when, earlier, instantaneous, simultaneous, timely
- Is the requirement simple and clear to understand?
  - no unnecessary adjectives, adverbs, comparisons

Examples of wrong and correct requirements, cont.

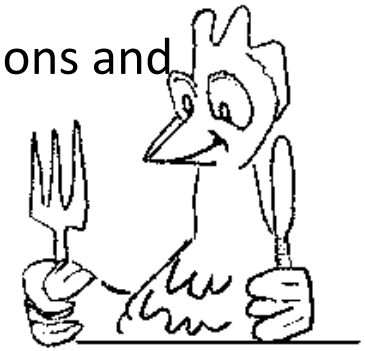
Non-Ambiguity

Checks, cont.

- Does the requirement refrain from making assumptions about the reader's knowledge of external systems or business processes?
  - If so, are these assumptions explained elsewhere?

- Does the requirement refrain from using indefinite pronouns?
  - all; another; any; anybody; anything; both; each; either; every; everybody; everyone; everything; few; many; most; much; neither; no one; nobody; none; one; several; some; somebody; someone; something; such

- Does the requirement refrain from using non-specific abbreviations and words?
  - If so, must be included in the glossary



Examples of wrong and correct requirements, cont.

Completeness

Wrong

- R1.1 The system shall monitor widgets

Correct

- R1.1 The system shall monitor 200 widgets
  for conditions X,Y,Z
- R1.2 The system shall update the status of widgets
  (refer to R1.1) every 1 second and display it in the administrator window

Checks

- Does the requirement contain enough information within itself to be of value without a dependency on another requirement at the same level?
- Does the requirement consider all possible paths and scenarios and contains information on how to deal with them?
- Does the requirement avoid the use of incomplete terms?
  - etc, TBD, unknown at this point, but not limited to, and/or

Examples of wrong and correct requirements, cont.

Consistency

Wrong

- R1.1 The system shall perform action X
- R1.2 In addition to R 1.1 it should allow the user to perform action Y
- The system shall be expected to perform action Z (1.3)

Correct

- R1.1 The system shall perform action X
- R1.2 The system shall allow the user to perform action Y
- R1.3 The system shall perform action Z

Checks

- Does the wording of the requirement follow a common set of conventions and is it consistent with the rest of the document?
- Does the formatting of the requirement follow a common set of conventions and is it consistent with the rest of the document?
- Is the requirement written in an active voice?
  - Active voice: the system shall calculate Z
  - Passive voice: Z shall be calculated

Examples of wrong and correct requirements, cont.

Verifiability

Wrong

- R2.3 The system shall minimise user input to complete procedure X

Correct

- R2.3 The system shall require 2 user inputs to complete procedure X

Checks

- Can the requirement be tested to prove that it has been satisfied?
- Does the requirement contain enough information to be tested and verified?
- Does the requirement contain/specify quantified data to be used as input for test scenarios?

Examples of wrong and correct requirements, cont.

Modifiability/Maintainability

Correct

- R2.3 The system shall require 2 user inputs to complete procedure X
- R2.3 The system shall require 3 user inputs to complete procedure Y

Checks

- Can the requirement be rewritten to change the content without changing the style and structure?
  - The system shall be able to perform transaction Y in time A
  - The system shall be able to perform transaction X in time B. "If", "when" and "where" must be failed

Examples of wrong and correct requirements, cont.

Traceability

Wrong

- R2.3 The system shall allow the user to change today's date as specified previously

Correct

- R2.3 The system shall allow the user to change today's date as specified in R2.1

Checks

- Does the requirement have a unique identifier?
- If it is a low level requirement can be it be traced to a high level requirement(s) from which it spawned?
- Does the requirement have a priority assigned?

Summary – why focus on the requirements early

- Find the errors early in the process
- Know that we have documented the requirements to a ”good enough” level
- Know that we are heading in the right direction
- Save time and money

Good practise tip!

Early test design

- Create test cases parallel with the requirement analysis
- Cooperation between requirement analyst and test manager
- The tester will understand the purpose with the requirements
- The requirement analyst understand errors in the requirements
- Identifies non-testable requirements when the tester must:
  - Understand the requirements in order to identify the tests that are required
  - Create activities to carry out tests
  - Define the expected results

Good practise tip!

Acceptance criteria for requirements -

- Clear for the requirements analys what it is they should create and at what quality criterias
- Reduce the amount of re-work since the requirements are of good quality to start with
- Should be agreed with business and developers

Any further questions?

Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Suggestion

When to Use What when Stating Requirements

- **Use Case (UC)**
  - When developing new functionality
  - When doing significant changes to existing flow
- **User Story (US)**
  - When doing minor changes to flow
  - When doing minor changes to user interface
  - When developing short and simple usage
- **Non functional requirement (NFR)**
  - When stating quality attributes
- **Software Requirement Specification (SRS)**
  - When neither UC nor US are used alone
  - Contains both functional and non functional requirements

What is a Use Case?

- A ***use case*** is a list of action or event steps, typically defining the interactions between a role (actor) and a system, to achieve a goal. The actor can be a human, an external system, or time
- Organizes functional requirements
- Records paths (called *scenarios*) from trigger events to goals
- Describes one main flow of events and one or more alternate flows
- Is multi-level, so that one use case can use the functionality of another one

Wikipedia

UC example

Show an example on paper or the coming slides

UC example cont.

Use Case: Edit an article

Primary Actor: Member *(Registered User)*

Brief description: *(equivalent to a User Story or an epic, see coming slides)*

The member edits any part (the entire article or just a section) of an article he/she is reading. Preview and changes comparison are allowed during the editing.

Wikipedia

UC example cont.

Preconditions:

The article with editing enabled is presented to the member

Postconditions

Success Guarantees:

The article is saved and an updated view is shown.

An edit record for the article is created by the system, so watchers of the article can be informed of the update in a while later.

Triggers:

The member invokes an edit request (for the full article or just one section) on the article

UC example cont.

Basic flow:

1. The system provides a new editor area/box filled with all the article's relevant content with an informative edit summary for the member to edit. If the member just wants to edit a section of the article, only the original content of the section is shown, with the section title automatically filled out in the edit summary.
2. The member modifies the article's content till satisfied.
3. The member fills out the edit summary, tells the system if he/she wants to watch this article, and submits the edit.
4. The system saves the article, logs the edit event and finishes any necessary post processing.
5. The system presents the updated view of the article to the member

Wikipedia

UC example cont.

Alternative flows/Extensions:

a. Show preview:

1. The member selects *Show preview* which submits the modified content.
2. The system reruns step 1 with addition of the rendered updated content for preview, and informs the member that his/her edits have not been saved yet, then continues.

b. Show changes:

1. The member selects *Show changes* which submits the modified content.
2. The system reruns step 1 with addition of showing the results of comparing the differences between the current edits by the member and the most recent saved version of the article, then continues.

c. Cancel the edit:

1. The member selects *Cancel*.
2. The system discards any change the member has made, then goes to step 5.

Wikipedia

What is a User Story (US)?

- A ***User Story*** is a short, simple description of a property (feature, characteristic) told from the perspective of the person who desires the new capability.
- Should be completed ("done") within one iteration
- Pattern: As a <type of user>, I want <some goal> so that <some reason>
- Must be complemented with stakeholder interactions or a collection of documents
- Containing enough information to be able to provide a reasonable estimate
- Is complemented with conditions of satisfaction (confirmation/ acceptance criteria) for verification that the goal is fulfilled
  - Do not confuse with acceptance test of the system
- Usually, a collection of work items is identified to bring a user story to "done"
- ***Epic*** is a concept which often represents a larger user story which is too big to be implemented and verified in one iteration. Typically, an epic is defined at the same level as a use case and a scenario

User Story card example

A User Story Card has three main parts:

- Card (i.e. “as a user, I want…”)
- Conversation (tasks and/or small wireframe to remind people about the feature)
- Confirmation (the tests that will show the feature is complete)

Ex of wireframe

Story ID: #15	Story: Scan Book
As a Borrower, I want to scan book, so that I can borrow it.	
Tasks: •Start session and initialize scanner •Read bar code of book •Lookup book •Display book info	
Responsible Person:	
Initial Estimate	Work Done / Work Left

Confirmation

1. Success – the book is scanned

a. The book is identified
b. The log is updated

2. Failure – display message:

a) ”Scanning interrupted – try again”

b) ”Book not known – contact Librarian”

c) ”Barcode not found – try again with front up”

Use Case – User Story
What's the difference?

User Story

- Brief description
- One or two sentences
- One flow at the time
- Used for planning
- Emerge faster
- More readable
- Verbal documentation
- Transient (Card, Conversation, Confirmation)
- Implemented and tested in one iteration
- Written by customer
- Contains acceptance test cases
- Need complementary UI specification
- From eXtreme Programming

Use Case

- Interaction actor-system
- Goal, summary, actor, precondition, trigger event, main success flow and alternative flows
- Main flow and alternative flows
- Not used for planning
- Requires analysis and writing
- Large document
- Textual model associated with diagrams
- Persistent (archived)
- Implemented and tested in several iterations
- Written by user proxy
- Separated from test cases
- Holistic views
- From Unified process

What is a Non functional requirement (NFR)?

- A ***non-functional requirement*** is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors
- Often called **qualities** of a system.
- Other terms:
  - constraints
  - quality attributes
  - quality goals
  - quality of service requirements
  - non-behavioral requirements

Quality Attributes - Functional

- Accuracy
  - adherence to specified or implied requirements.
- Suitability
  - appropriateness of functions for specified tasks.
- Interoperability
  - Functionality in all intended environments
- Functional security
  - Access to functions and data
- Usability
  - Suitability of system for all intended users

Quality Attributes - Technical

- Accessibility
  - Accessibility of software to those with particular requirements or restrictions
- Technical security
  - Security from use by unauthorized persons
- Reliability
  - Robustness and recoverability
- Efficiency
  - Performance, load, stress and scalability testing
- Maintainability
  - Ease of analyzing, changing and testing
- Portability
  - Ease of transfer to intended environment

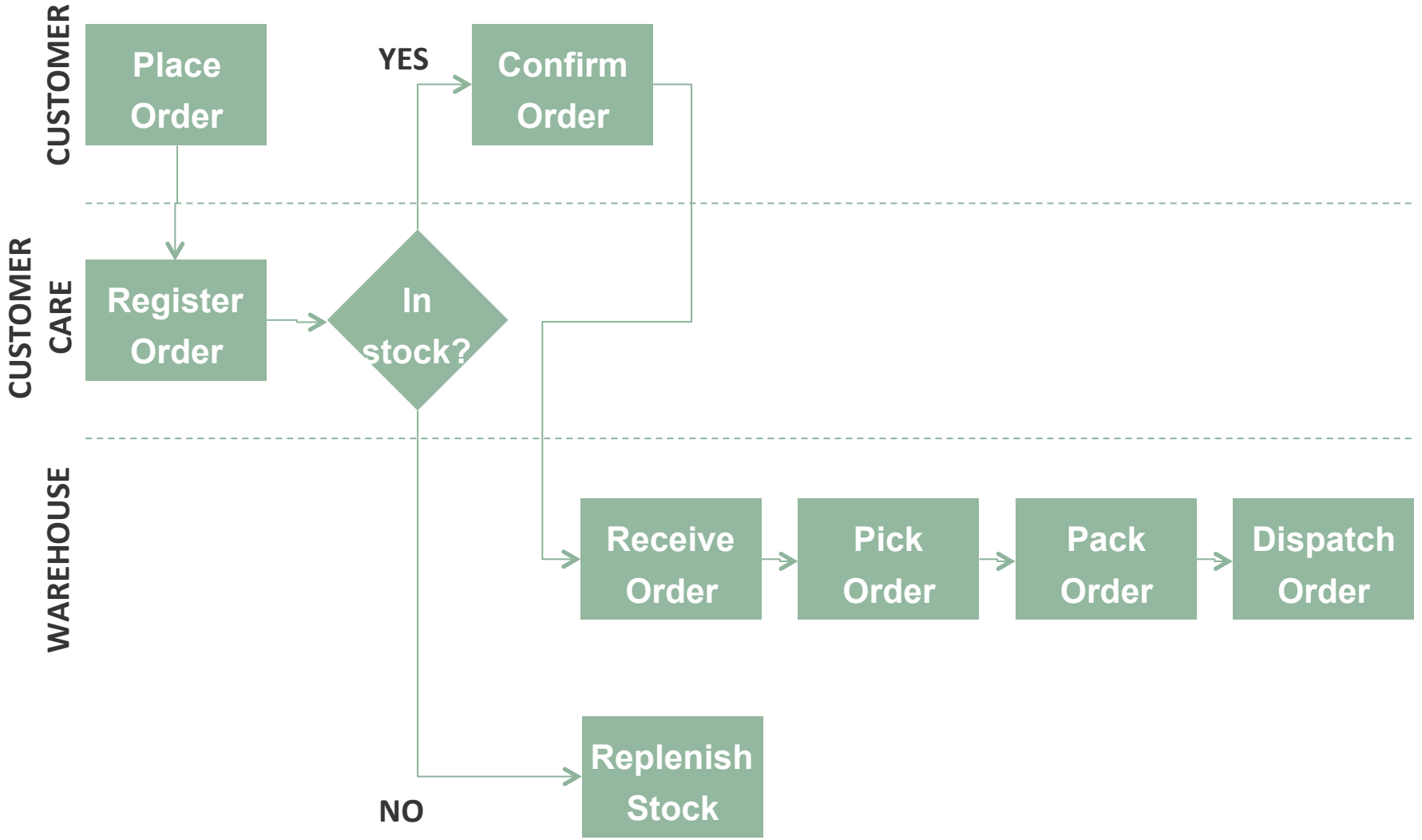
What is a Software Requirement Specification (SRS)?

- A ***Software Requirement Specification (SRS)*** is a description of a software system to be developed
  - laying out both functional and non functional requirements
  - may include a set of Use Cases that describe interactions the users will have with the software
- Establishes the basis for an agreement between customers and contractors/ suppliers on what the software product is to do as well as what it is not expected to do

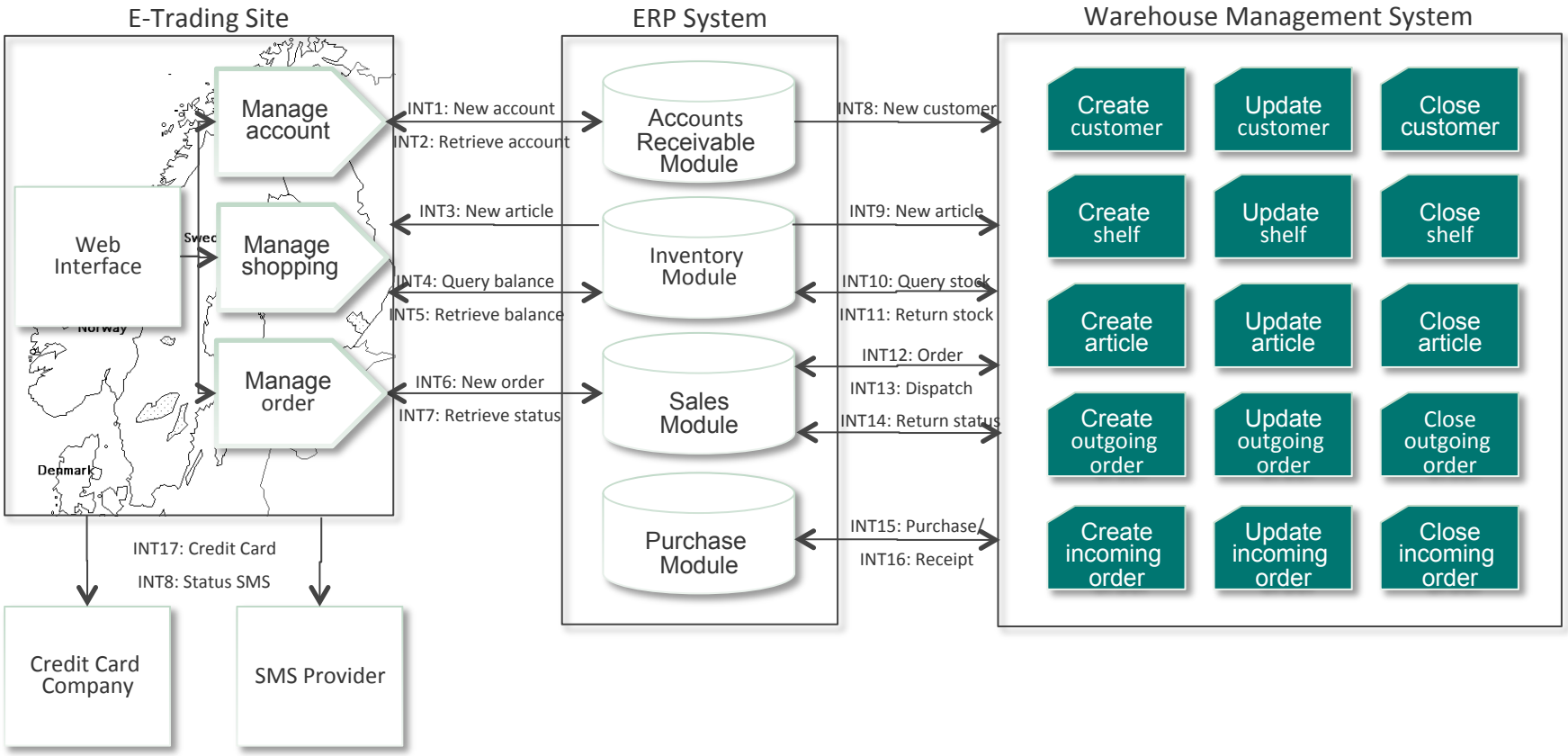
Case Work Task 3 – Create Requirements

- **In this task you will practice creating requirements**
- **Build on the information from the Case Work Introduction**
- **Review information from requirement lead**
- **Create requirements for Requirement Group 3-5**
  - Teachers will moderate the discussion
  - Think about the different scenarios
  - Think about the information the system must have
  - Also think about what might go wrong

Case Work Task 3 – Business Process (End User etc.)



Case Work Task 3 – Functional Design (Designer etc.)



Case Work Task 3 – Requirements (Requirements Lead)

Id	Requirement Group
R1	The system must allow staff to set up shelves and connect articles to them
R2	The system must return stock levels by article
R3	The system must receive customer details
R4	The system must receive customer orders
R5	The system must support the physical pick, pack and dispatch of outgoing orders
R6	The system must return order status
R7	The system must support return orders
R8	The system must support the physical reception of return orders
R9	The system must support optimized inventory planning
R10	The system must receive incoming orders
R11	The system must support the physical reception of incoming orders
R12	The system must print invoices and reports

Case Work Task 3 – Sample Solution

Id	Description
1	Customer orders must be sent to the WMS detailing article number, article description, number of items, item price, total price, shelf number and current stock level.
2	Users must be able to print pick lists detailing article number, article description, number of items, shelf number and current stock level.
3	Users must be able to check in and out items to and from the shelves.
4	The system must update the stock level when items are checked in and out from the shelf.
5	Users must be able to print address labels detailing the customer name and customer address.
6	Users must be able to print customer invoices detailing article number, article description, number of items, item price and total price.
7	Users must be able to register packages dispatched to the courier.
8	The system must be able to update the order status.
9	Are quality attributes/non-functional requirements necessary?
10	
11	
12	

Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Day Wrap Up

- Topics covered today
  - Test Project
  - Test Management
  - Test Strategy
  - Test Metrics
  - Test Plan
  - Requirements
  - Test Analysis
- Which additional learnings have you done?
- Which learnings are you still missing?
- Tomorrow we will practice more on a test lead's daily challenges

Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

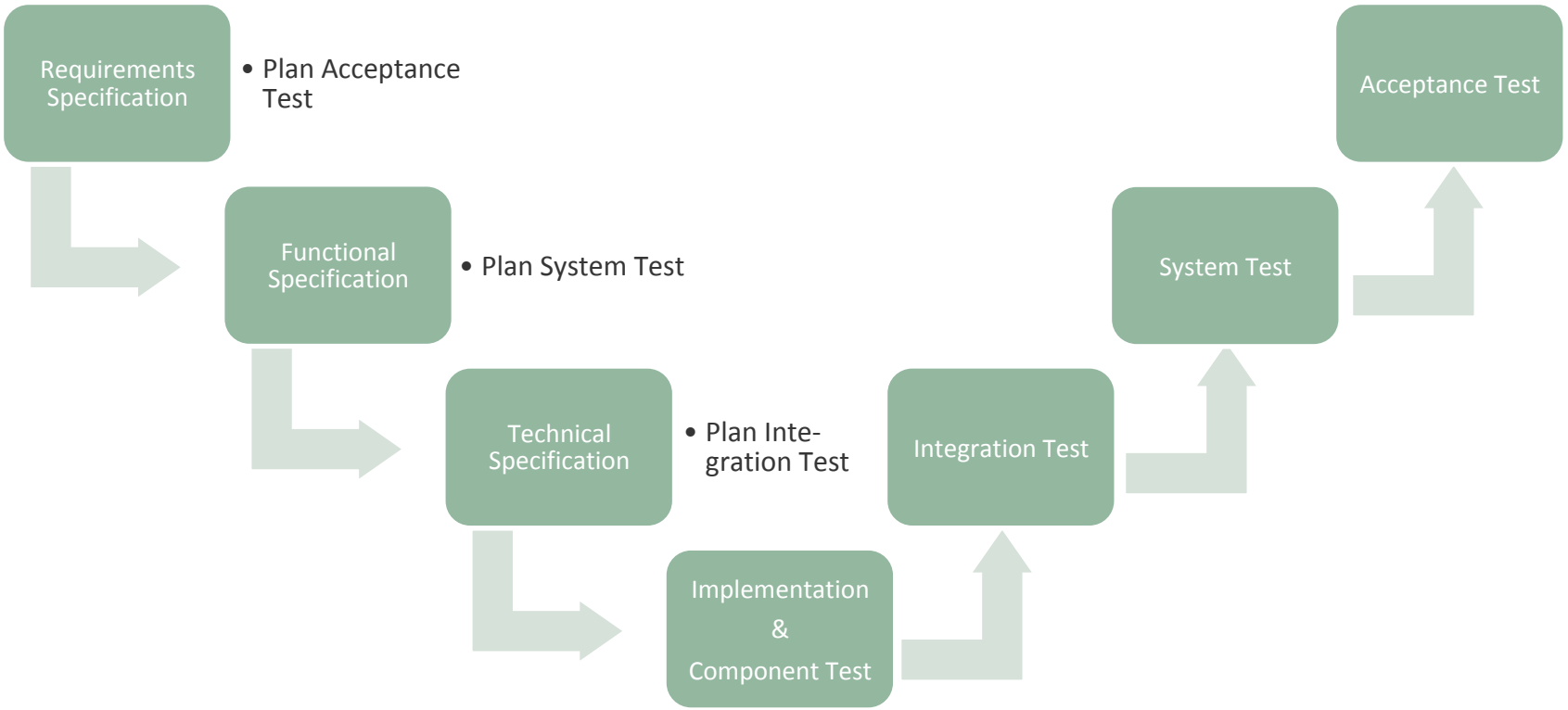
Checkpoint Rules

- Now it's time to test what you've learnt!
- The checkpoint is a quiz in Jeopardy style
  - Faculty gives the answers
  - You give the questions
- Group in pairs and select categories in turn
- No penalties if you get it wrong so if you don't know – guess!
- After the question – right or wrong – the turn goes to next pair
- The quiz ends with a bonus question
  - Write down your bet before faculty gives the answer
  - Write down your question
  - If you get it right you get twice the amount back (no penalties if you get it wrong)
- The winner is the pair with the most money left

Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Test in a Project Lifecycle



Test Levels

- Component Test (Unit Test): Tests the individual units of an entire solution. The test ensures that the component (or module) is built and behaves as detailed in the specifications.
- Component integration test (String test): Tests the assembly and combined operation of related components. It ensures that the interactions between the components function correctly.
- Integration Test: An end-to-end test of the business requirements across all applications and platforms.
- System Integration Test: Tests the integrations with existing or 3rd party applications and platforms not in the scope of the project.
- System test (Product test): Tests that all functional and business requirements have been met by the system.
- Acceptance Test (UAT): Ensures that the users and stakeholders are satisfied with the solution.

Test Case Activity

Test Level		Test Case
Component test		Send ”close article” message from ERP to WMS
Component integration test		Create book order, create gift certificate order, create partially available order
Integration test		Send customer details from order to post label printer
System integration test		Place an order as a new customer and receive a package
System test		Join article id from incoming order with shelf id from WMS
Acceptance test		Create order in ERP and check that order is recieved in WMS

Test Case Activity

Test Level		Test Case
Component test		Send ”close article” message from ERP to WMS
Component integration test		Create book order, create gift certificate order, create partially available order
Integration test		Send customer details from order to post label printer
System integration test		Place an order as a new customer and receive a package
System test		Join article id from incoming order with shelf id from WMS
Acceptance test		Create order in ERP and check that order is recieved in WMS

Test Design Techniques

- Specification-based (black box)
  - Equivalence partitioning: Input with similar expected output are identified and grouped
  - Boundary value analysis: Minimum, maximum and invalid values are identified
  - Decision table testing: Unique combinations of conditions are identified
  - State transition testing: Different responses to conditions (states) are identified
  - Classification tree method: Options of variables of interest are identifed
  - Use case testing: User/system interactions (including pre- and postconditions) are identified
- Structure-based (white box)
  - Statement testing: Executable statements are identified
  - Decision testing: Decision statements are identified (if/else, for, while etc.)
  - Branch testing: Branches are identified (if/else, for, while etc.)
  - Condition testing: True/false labels are identified
  - Multiple condition testing: Combinations of true/false conditions are identified
  - Condition determination testing: Combinations of true/false conditions that can affect branches are identified
  - Loop testing: Conditions that control loop iteration are identified
  - Path testing: Unique sequences of paths are identifed

Test Design Techniques

- Defect-based:
  - Taxonomies: Potential problem is used as basis for test design
- Experience-based
  - Error-guessing: Use of experience to guess potential errors
  - Checklist-based: Use of set of criteria against which a product has to be verified
  - Exploratory: Simultaneous learning, planning and execution (”free testing”)
  - Attacks: Focused attempts to provoke specific failures
- Static Analysis (without executing the code)
  - Static analysis of code
  - Static analysis of architecture
- Dynamic analysis (while executing the code)
  - Detect memory leaks (memory allocated to program is not released)
  - Detect wild pointers (pointing to wrong objects, functions, memory etc.)
  - Analysis of performance (identify bottlenecks etc.)

Non-functional tests

- Accuracy test: Tests application's adherence to specified or implied requirements.
- Suitability test: Evalueates and validates appropriateness of functions for specified tasks.
- Interoperability test: Tests whether an application functions in all targeted environments.
- Functional security test: Tests ability to prevent unauthorized access.
- Usability test: Measures suitability of application for its users.
- Accessibility test: Accessibility of software to those with particular requirements or restrictions.
- Technical security test: Prevent unintended use.
- Reliability test: Tests robustness and recoverability.
- Efficiency test: Tests performance, load, stress and scalability.
- Maintainability test: Tests the ease with which software can be maintained.
- Portability test: Tests ease with which software can be transferred.

Specification-Based Test Design

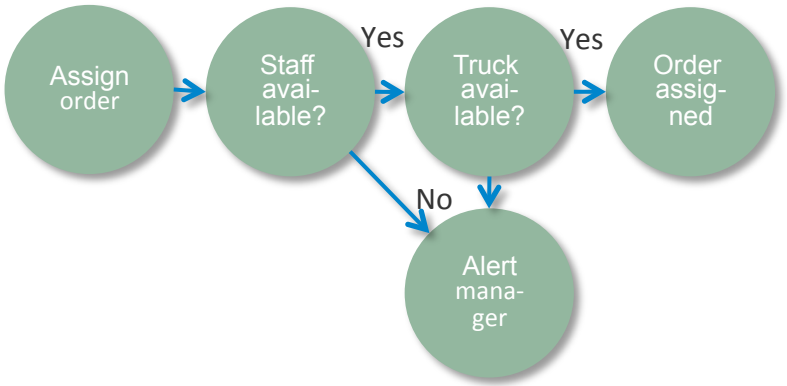
Requirement Group 5: Staff picks, packs and dispatches outgoing order (maximum 99 books)

Equivalence partitioning	
Invalid partition 1	-0
Valid partition 1	1-99
Invalid partition 2	100+

Boundary value analysis	
Boundary 1	0, 1
Boundary 2	99, 100

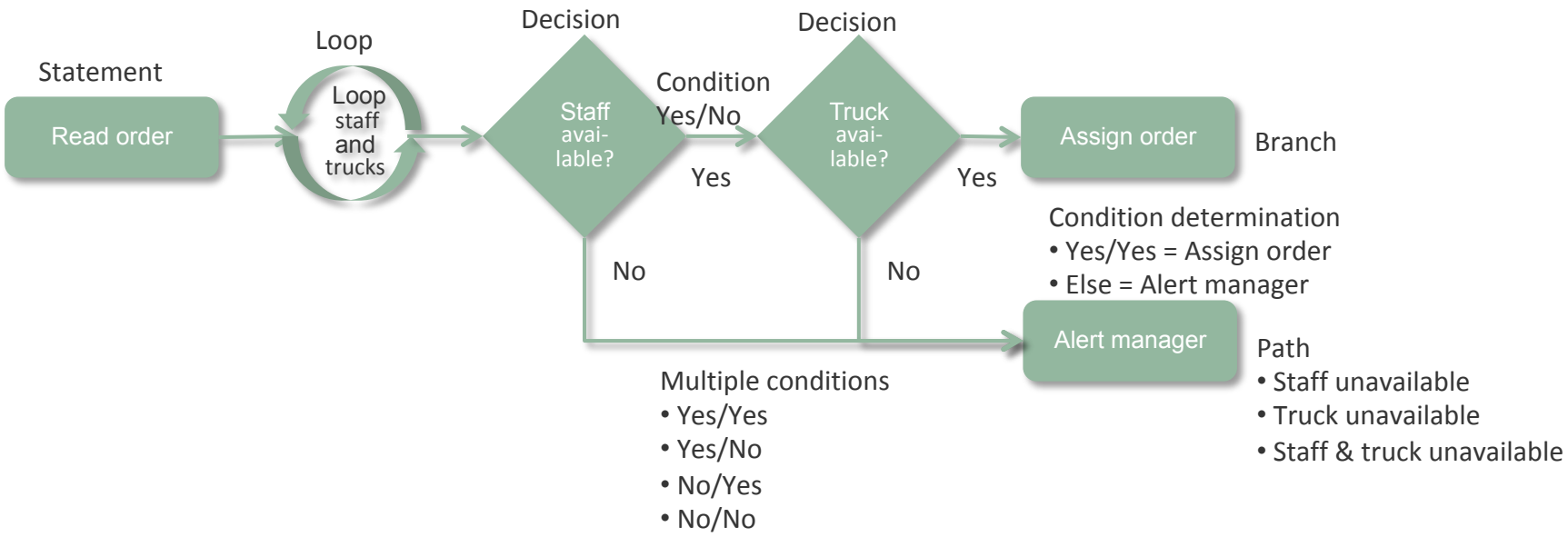
Decision table					
Condition	Staff available	Y	Y	N	N
	Truck available	Y	N	Y	N
Action	Assign order	X			
	Wait for staff			X	X
	Wait for truck		X		X
	Alert manager		X	X	X

Classification tree



State transition			
State	Assign order	Pick, pack, dispatch	Interrupt order
S1 Order unassigned	S2	-	S1
S2 Order in progress	S2	S3	S1
S3 Order closed	-	-	-

Structure-Based Test Design



A typical 3 pass test schedule

Day 0: New code is migrated and smoke test is executed

Day 1: Test team execute pass 1, development team fix defects

- If system is not stable, a mid-pass code migration may be required and a pass restart required
- If not all functionality is delivered, consider adding passes (ideally all functionality should be executed during at least two passes)
- Each pass is executed in 3 cycles
  - Cycle 1: Test of all major business scenarios
  - Cycle 2: Test of alternative business scenarios
  - Cycle 3: Series of exception processing tests

Day 5: Code freeze, test scripts executed, new code migrated and test environment restored

Day 6: Test team execute pass 2, development team fix defects

Day 10: Code freeze, test scripts executed, new code migrated and test environment restored

Day 11: Test team execute pass 3, development team DOES NOT fix defects (as long as the severity is low) but rather begins to work on next release

Day 15: Test team check metrics against exit criteria

- If failed a fourth pass is required
- If passed the project may proceed to next test stage

Test Closure Activities

- Test closure activites fall into four main groups
  1. Ensure that all test work is concluded
     - Planned tests run or deliberately skipped
     - Defects closed or managed
  2. Deliver valuable work products
     - Known defects and workaround communicated to users
     - Tests and test and test environments given to maintenance
  3. Participate in retrospective meetings (lessons learned)
     - Improve estimates
     - Identify root causes for defects
     - Identify process improvement opportunities
  4. Archive work products
- These tasks should be included in the exit criteria
- In addition, a test summary report should be prepared and reported

Test Summary Report (IEEE)

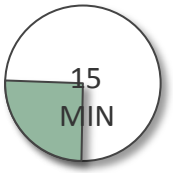
- Test Summary Report Identifier: Test level and software level
- Summary: Information about what was tested
  - Test Items: What was tested?
  - Test Environment: Where was it tested?
  - References: Supporting documents
- Variances: Deviations from agreed scope and quality
  - Specification: What deviated?
  - Reason: Why did it deviate?
  - Support material: Change requests, incident reports etc.
- Assessment:
  - Test coverage: How well was test completed?
  - Test improvements: How was the test changed or how should the test be changed to improve?
- Results:
  - Incidents: By priority, severity, impact
  - Defect patterns: Where were the defects detected?
  - Open incidents: With owner and action
- Evaluation: Quality of software
  - Limitations: Incomplete functions or dropped features
  - Failure likelihood: Risk areas, good quality areas
- Activities: Planned and actual time and costs
- Approvals: Formal management approval (test lead does not approve, only recommends)

Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Case Work Task 4 –Design Test

- **In this task you will practice designing a test case**
- **Build on the information from the Case Work Introduction**
- **Design test cases for the requirements created in the previous task**
  - Assume that the separate function tests have been designed and that end-to-end tests remain to be designed
  - Focus on the main flow and leave exception flows
  - Think of features not covered by the requirements that you may want to test
- **Share your conclusions with the class**
  - Present your test case
  - Which assumptions/considerations is it based on?



IEEE Test Case Standard

- **Test case identifier.** A unique label so you can refer to that document. Point to the test plan/design.
- **Test items.** List the items and features you will check. Point to their documentation.
- **Input specifications.** Describe all the information passed to the test item for this test. [Either point to files, or include the information in such a way that it can be automatically extracted.]
- **Output specifications.** Describe all the behaviours required, including non-functional requirements like time, memory use, network traffic. Provide exact values if you can.
- **Test environment needs.** What hardware, software, and other stuff do you need?
- **Special procedural requirements.** Any special setup, user interaction, or tear-down actions?
- **Inter-case dependencies.** What other test cases must be done first? Point to them. Why must they be done first?

User Interface Specification

- User interface specification (UI specification)
  - Document that captures the details of the software user interface into a written document.
  - Covers all possible actions that an end user may perform and all visual, auditory and other interaction elements
- Graphical user interface
  - Elements used by graphical user interfaces (GUIs) to offer a consistent visual language to represent information stored in computers.
  - Make it easier for people with few computer skills to work with and use computer software.
  - Example: Windows, buttons, drop-down lists

Case Work Task 3 – Sample Solution

Id	Description
1	Customer orders must be sent to the WMS detailing article number, article description, number of items, item price, total price, shelf number and current stock level.
2	Users must be able to print pick lists detailing article number, article description, number of items, shelf number and current stock level.
3	Users must be able to check in and out items to and from the shelves.
4	The system must update the stock level when items are checked in and out from the shelf.
5	Users must be able to print address labels detailing the customer name and customer address.
6	Users must be able to print customer invoices detailing article number, article description, number of items, item price and total price.
7	Users must be able to register packages dispatched to the courier.
8	The system must be able to update the order status.
9	Are quality attributes/non-functional requirements necessary?
10	
11	
12	

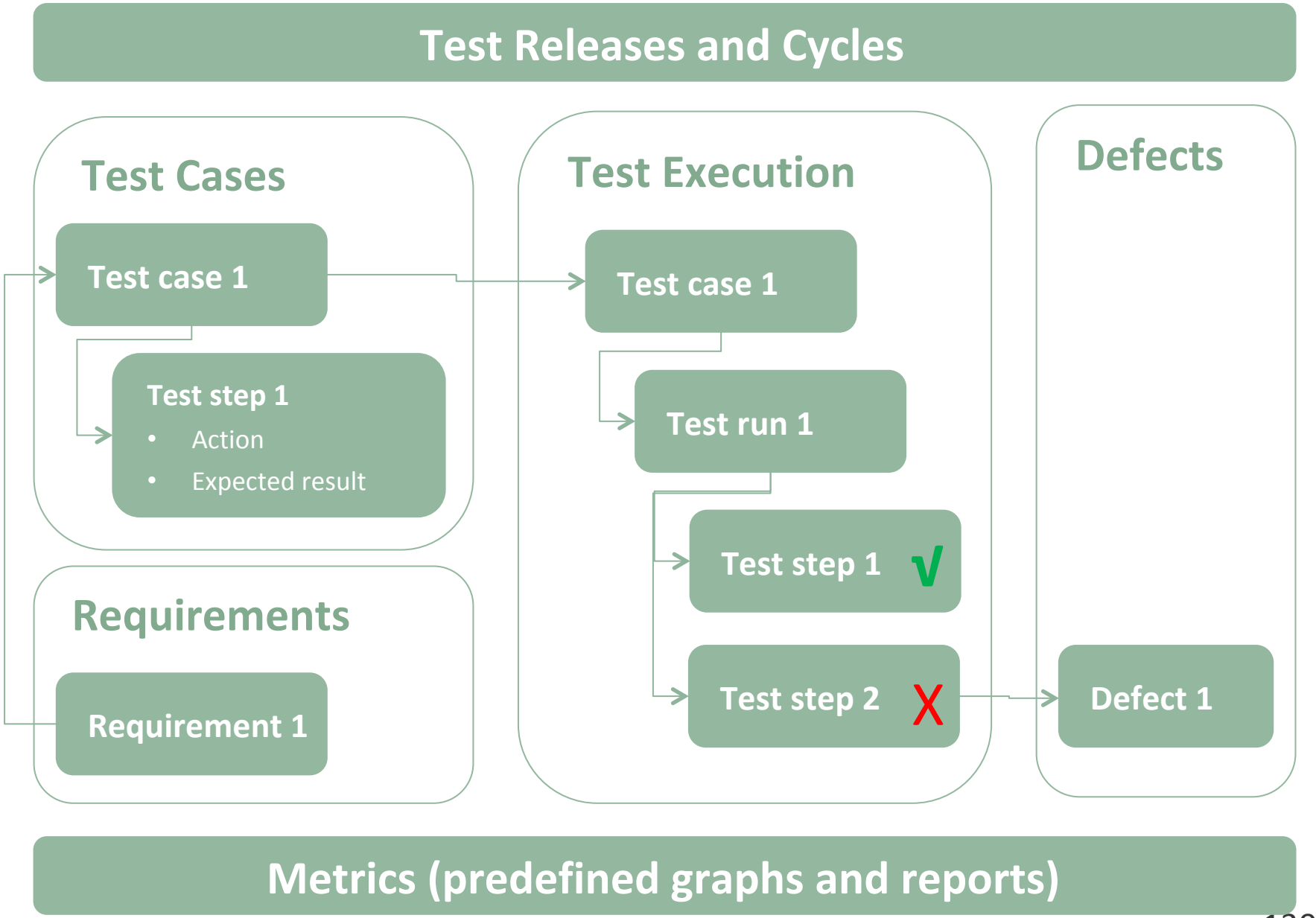
Case Work Task 4 – UI Specification (Requirements Lead)

Id	Handheld device for warehouse pick, pack and dispatch process		
1	Logon Screen	Input field: Name Input field: Password Buttons: OK/Cancel	Correct credentials: Welcome Screen Incorrect credentials: Error Screen
2	Error Screen	Input field: Name Input field: Password Buttons: OK/Cancel	Error message
3	Welcome Screen	Button: Order Button: Pack Button: Dispatch	Menu for warehouse options
4	Order Screen	List of Orders Buttons: OK/Cancel	Order selection Cancel: Welcome Screen OK: Pick Screen
5	Pick Screen	List of Articles, Shelves and Number to pick Buttons: OK/Cancel	Pick instructions Cancel: Undo pick OK: Register pick
6	Pack Screen	List of Picked Orders Buttons: OK/Cancel	Pack confirmation Cancel: Undo pack OK: Register pack
7	Dispatch Screen	List of Packed Orders Buttons: OK/Cancel	Dispatch confirmation Cacnel: Undo dispatch OK: Register dispatch

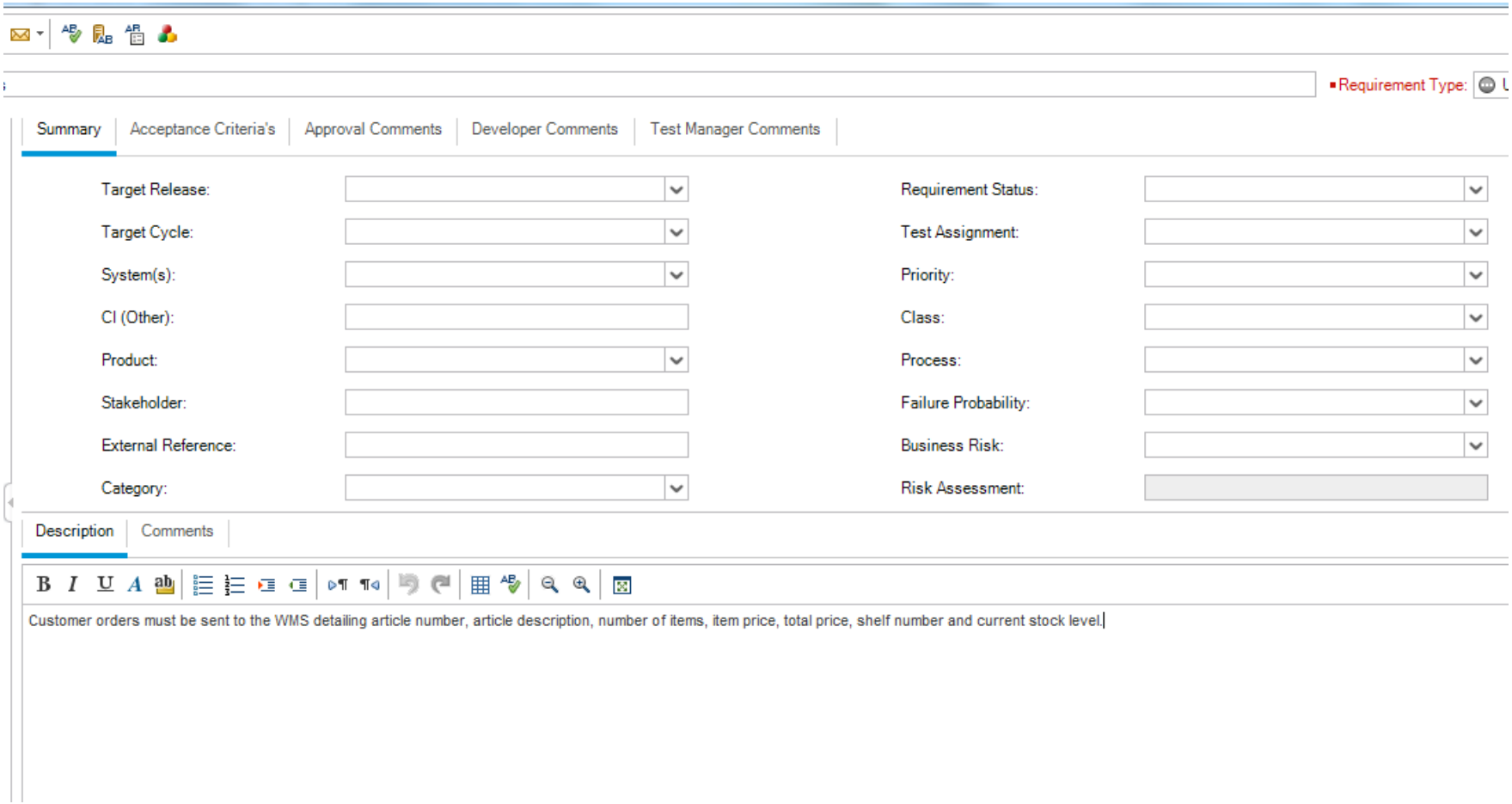
Case Work Task 4 – Sample Solution

Id	Description	Expected Results
1	Log on to the handheld device with [parameter: name] [parameter: password][1)]	Handheld device starts and welcome screen displayed
2	Press button ”Order”	Orders [parameter: order_no] displayed sorted by order date ascending[2)]
3	Select the first order and press button ”OK”	Order locked in system[3)] Pick route displayed showing [parameter: article_no] [parameter: article_name] [parameter: shelf_no] [parameter: items]
4	Select the first article and press button ”OK”	Stock level updated in system[3)] Article gets marked ”picked”
5	Select the second article and press button ”OK”	Stock level updated in system Article gets marked ”picked” Pack station screen displayed
6	Press button ”Pack” [4)]	Picked order appears
7	Select order and press button ”OK”	Order status updated to ”Packed” in system
8	Press button ”Dispatch”	Packed order displayed
9	Select order and press button ”OK”	Order status updated to ”Dispatched” in system
10	Exploratory testing[5)]	What happens if you press other button?
11	Non-functional testing[6)]	Does the device have contact with the system everywhere in the warehouse?

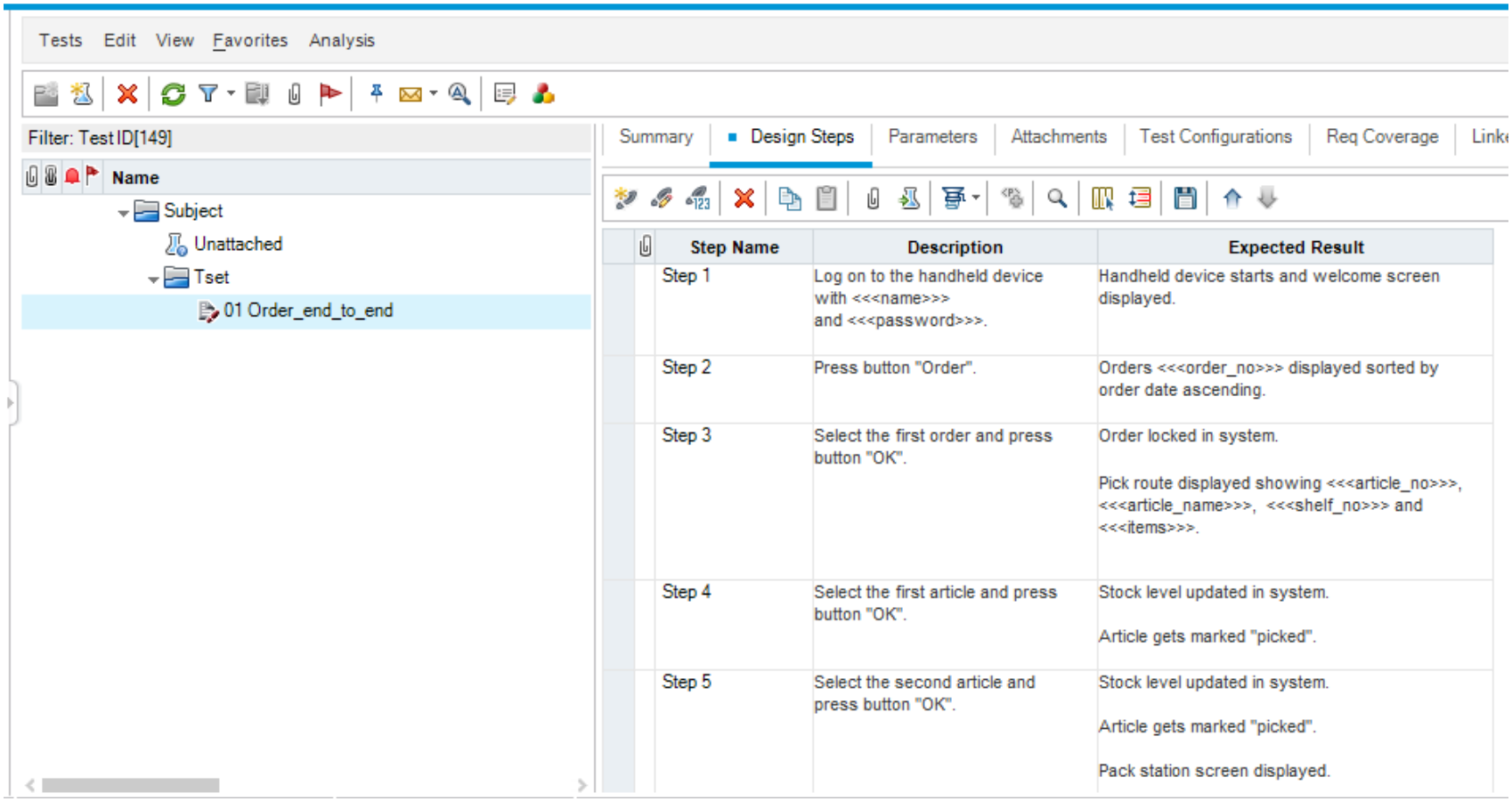
Test Tools support Test Management



Case Work Task 4 - Solution

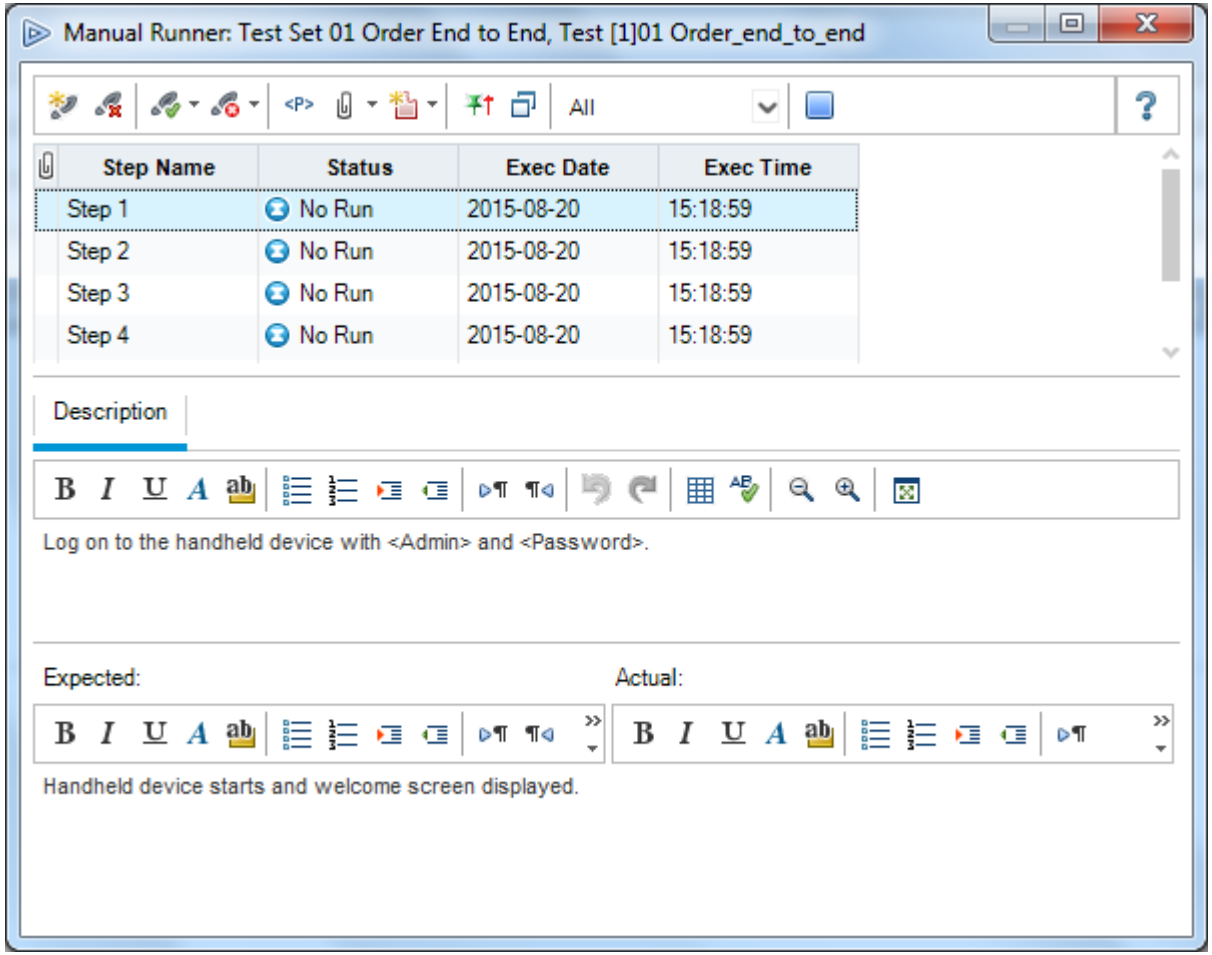


Case Work Task 4 - Solution



Step Name	Description	Expected Result
Step 1	Log on to the handheld device with <<<name>>> and <<<password>>>.	Handheld device starts and welcome screen displayed.
Step 2	Press button "Order".	Orders <<<order_no>>> displayed sorted by order date ascending.
Step 3	Select the first order and press button "OK".	Order locked in system. Pick route displayed showing <<<article_no>>>, <<<article_name>>>, <<<shelf_no>>> and <<<items>>>.
Step 4	Select the first article and press button "OK".	Stock level updated in system. Article gets marked "picked".
Step 5	Select the second article and press button "OK".	Stock level updated in system. Article gets marked "picked". Pack station screen displayed.

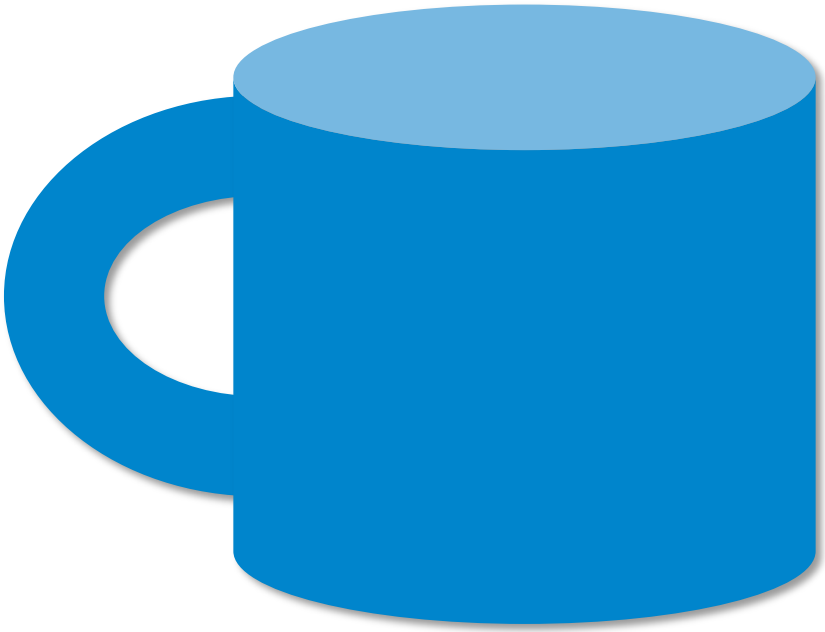
Case Work Task 4 - Solution



Ticket Monster

- Practice test activities online at (http://www.jboss.org/ticket-monster/)
  - Manual Test
  - Automated Test
  - Performance Test
- To be used later this training

Coffee Break!



Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Case Work Task 5a –Implement Test

- **In this task you will practice planning test cycles**
- **Build on the information from the Case Work Introduction**
- **Review information about solution content and releases**
- **Use the information to prepare a test schedule for the system test**
- For simplicity, base your test cycles on the requirement groups and determine the following:
  - Which delivery is it?
  - What is the functional complexity?
  - What is the business criticality?

Case Work Task 5a – Form

Id	Requirement Group	Delivery (1-3)	Functional Complexity (1-3)	Business Criticality (1-3)
R1	The system must allow staff to set up shelves and connect articles to them			
R2	The system must returns stock levels by article			
R3	The system must receive customer details			
R4	The system must receive customer orders			
R5	The system must support the physical pick, pack and dispatch of outgoing orders			
R6	The system must return order status			
R7	The system must support return orders			
R8	The system must support the physical reception of return orders			
R9	The system must support optimized inventory planning			
R10	The system must receive incoming orders			
R11	The system must support the physical reception of incoming orders			
R12	The system must print invoices and reports			

Case Work Task 5 – Delivery Plan (Release Manager, Configuration Manager)

System	Delivery 1	Delivery 2	Delivery 3
E-Trading Site	Articles Shopping (surfing only)	Account Order Return	Secure payment Status notification Reporting
ERP Integrations	Inventory Module Purchase Module	Sales Module A/R Module	Credit Card Company SMS Provider Reporting
WMS	Shelves Articles Incoming order	Outgoing order Customer Stock level	Inventory planning Reporting

- Release 1 offers the end customer basic shopping functionality
  - Delivery 1 and 2 are critical for go-live
  - Delivery 3 may be postponed to Release 2 if severe defects
- Release 2 offers the end customer a personalized shopping experience
  - Purchase recommendations
  - Flexible delivery
  - Extended assortment

  (to be planned in detail, i.e. out of scope for this case work)

Case Work Task 5a – Complex areas (Development Lead etc.)

Area	Integrations	Functional complexity (1=top)
Receive incoming order	INT15	1
Receive outgoing order	INT6, INT12	1
Receive return order	INT6, INT12	1
Reporting	N/A	1
Secure payment	INT17	1
Manage return order	INT7, INT13	2
Dispatch outgoing order	INT7, INT13	2
Customers	INT1, INT2, INT8	2
Order status	INT7, INT14	2
Place incoming order on shelf	INT16	2
Order status SMS	INT18	3
Shelves	N/A	3
Articles	INT3, INT9	3
Stock	INT4, INT5, INT10, INT11	3
Inventory planning	N/A	3
Shopping (surfing only)	N/A	3

Case Work Task 5a – Critical Areas (Client Manager etc.)

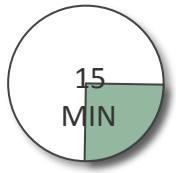
Business Process	Business Criticality (1-3)
”All our work is driven by customer orders.”	1
”We need to keep trac of which shelves we have and which articles we have on them.”	1
”The warehouse staff needs help to work effectively.”	1
”When an item runs low, we order new items.”	2
”Incoming items are received on a weekly basis.”	2
”Sometimes, we must handle returned orders.”	2
”Returned items must be returned to the correct shelf.”	2
”Address labels are printed by us.”	2
”Invoices are also printed by us.”	2
”Our staff are usually observant on shelves getting empty.”	3
”Our customers appreciate to get to know when an order is on its way.”	3
”My experience helps me know how many items to keep on the shelf.”	3

Case Work Task 5a – Solution

Id	Requirement Group	Delivery (1-3)	Functional Complexity 1-3)	Business Criticality (1-3)
R1	The system must allow staff to set up shelves and connect articles to them	1	3	1
R2	The system must returns stock levels by article	2	3	3
R3	The system must receive customer details	2	2	2
R4	The system must receive customer orders	2	1	1
R5	The system must support the physical pick, pack and dispatch of outgoing orders	2	2	1
R6	The system must return order status	3	2	3
R7	The system must support return orders	1	1	2
R8	The system must support the physical reception of return orders	1	2	2
R9	The system must support optimized inventory planning	3	3	3
R10	The system must receive incoming orders	1	1	2
R11	The system must support the physical reception of incoming orders	1	2	2
R12	The system must print invoices and reports	3	1	2

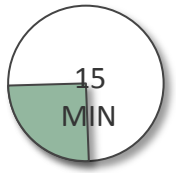
Case Work Task 5b – Implement Test

- **In this task you will practice risk-based quality management**
  - The risk category is composed of two factors: business criticality and failure probability
  - The functional complexity category indicates the complexity of the requirement's implementation
  - The business criticality indicates how important the requirement is for the business processes.
- **Risk-based quality management is an approach to risk mitigation in the end-to-end quality effort which has a critical linkage to requirements and requirements traceability**



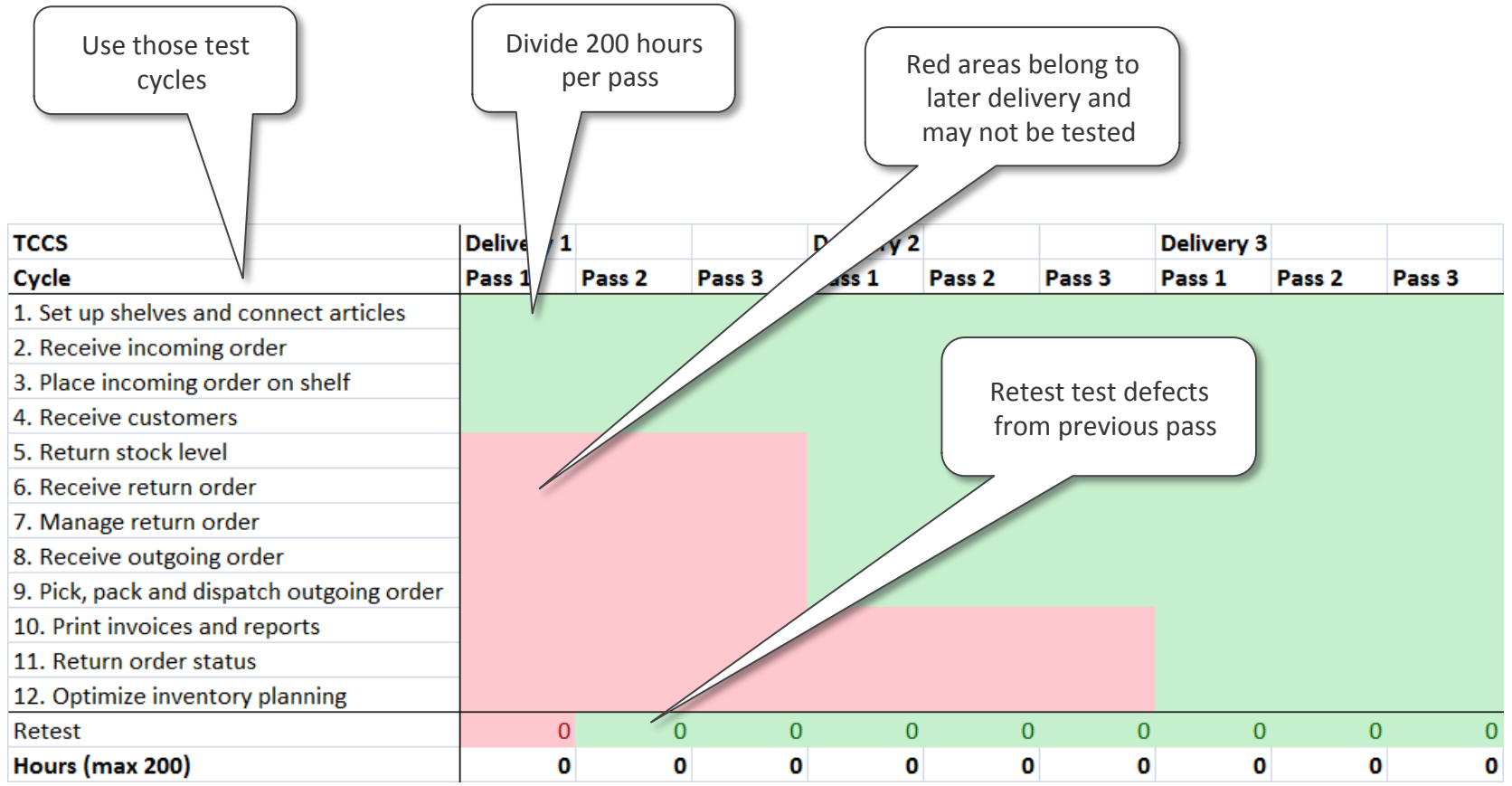
Case Work Task 5b – Implement Test

- **Build on the information from the Case Work Task 5b**
- **Use the information to prepare a test schedule for the system test**
  - Use the schedule template and cycles sent out by faculty
  - Schedule your test by dividing hours among the weeks
  - Apply risk-based quality management
  - You may divide no more than 200 hours per pass (5 testers * 8 hours * 5 days)
  - 1 retest hour in one pass closes 1 defect from previous passes
  - Return your test schedule to faculty
  - In later activities your scheduling will determine how many defects you find and the winner is the one who closes most severe defects!
  - Bonuses are scored for best test approach and test report



- DO NOT add, delete or move around any lines in the schedule

Case Work Task 5c – Schedule template



TCCS	Delive 1			D y 2			Delivery 3		
Cycle	**Pass 1**	**Pass 2**	**Pass 3**	**ss 1**	**Pass 2**	**Pass 3**	**Pass 1**	**Pass 2**	**Pass 3**
1. Set up shelves and connect articles									
2. Receive incoming order									
3. Place incoming order on shelf									
4. Receive customers									
5. Return stock level									
6. Receive return order									
7. Manage return order									
8. Receive outgoing order									
9. Pick, pack and dispatch outgoing order									
10. Print invoices and reports									
11. Return order status									
12. Optimize inventory planning									
Retest	0	0	0	0	0	0	0	0	0
Hours (max 200)	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Case Work Task 6 – Execute Test, Delivery 1

- **In this task you will practice test control**
- **Test control is an ongoing activity, involving the following:**
  - Comparing actual progress against plan
  - Reporting status, including deviations from plan
  - Guiding testing to fulfill objectives
  - Revisiting test planning activites as needed
  - Responding to information generated by testing
  - Responding to changing project conditions
  - Reprioritizing and reallocating test effort as necessary
- Metrics to monitor test planning and control include
  - Risk and test coverage
  - Defect discovery and information
  - Planned versus actual hours to design and execute test

Case Work Task 6 – Execute Test, Delivery 1

- **Your testers will** ”execute” the first delivery according to your test schedule
- **After each pass faculty will inform you of the number of defects found**
  - Review the results and update the sheet if necessary
  - Return the sheet to faculty for next pass

TCCS	Delivery 1		
Cycle	Pass 1	Pass 2	Pass 3
1. Set up shelves and connect articles	50	50	50
2. Receive incoming order	50	50	50
3. Place incoming order on shelf	50	50	50
4. Receive customers			
5. Return stock level			
6. Receive return order			
7. Manage return order			
8. Receive outgoing orde			
9. Pick, pack and dispatch			
10. Print invoices and rep			
11. Return order status			
12. Optimize inventory planning			
Retest	0	0	0
Hours (max 200)	**200**	**200**	**200**

DEFECTS DETECTED TOTAL	Delivery 1		
Cycle	Pass 1	Pass 2	Pass 3
1. Set up shelves and connect articles	3		
2. Receive incoming order	6		
3. Place incoming order on shelf	5		
4. Receive customers	5		
5. Return stock level			
6. Receive return order			
7. Manage return order			
8. Receive outgoing orde			
9. Pick, pack and dispatch			
10. Print invoices and rep			
11. Return order status			
12. Optimize inventory planning			
Sum	**19**	**0**	**0**
Retest	**0**	**0**	**0**
Total Open	**19**	**19**	**19**

Example:

1. Test lead allocates 50+50+50+50 hours in Pass 1
2. Testers (faculty) discovers 3+6+5+5 defects in Pass 2
3. Test lead may now decide to reallocate hours in Pass 2

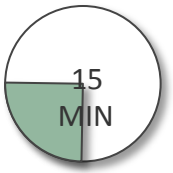
Lunch!

Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Case Work Task 7 –Status Report

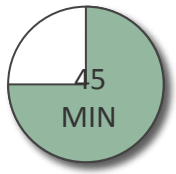
- **In this task you will practice reporting status**
- **With the first delivery tested**, you will report status to the project management
  - Use the test schedule from previous task
  - Prepare a status report according to your test approach (test metrics etc.)
  - Assume that 1 hour = 1 test case
- **Present your status report to the class**
  - Adapt your presentation to the audience (project management)
  - Focus on results, risks and issues – not on process (the client is not interested in how you work but on the result of your work)



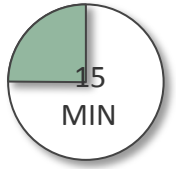
Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Case Work Task 9 – Execute Test, Delivery 2



- In this task you will practice more test control
- In addition to previous delivery, you will now have to deal with ”unexpected events”
  - You will receive ”test lead cards” based on your test approach
  - Faculty will inform you of the unexpected events
  - Each unexpected event may be countered by playing a test lead card (but mitigation cards work as ”wild cards”)
  - Only one test lead card may be played per event and may not be reused so select carefully
  - Faculty will inform you of the effect and you should update your schedule accordingly
- **Your testers will** ”execute” the second delivery according to your test schedule
- **After each pass faculty will inform you of the number of defects found**
  - Review the results and update the schedule if necessary
  - Return the schedule to faculty for next pass



Case Work Task 9 – Example of Unexpected Events

” No plan survives its collision with reality - Susan Scott”

An unexpected event…

… is countered by good planning and flexibility

Unexpected Event 0

Test cases are accidentally deleted. You must rewrite the test scripts and lose 10 testing hours in Delivery 2, Pass 1.

Test Lead Card: Test Tools

Test scripts are stored in a central repository with version control and backup functionality

TCCS	Delivery 2		
Cycle	Pass 1	Pass 2	Pass 3
1. Set up shelves and connect articles	5		
2. Receive incoming order	5		
3. Place incoming order on shelf	5		
4. Receive customers	5		
5. Return stock level	30		
6. Receive return order	35		
7. Manage return order	35		
8. Receive outgoing order	35		
9. Pick, pack and dispatch outgoing order	35		
10. Print invoices and reports			
11. Return order status			
12. Optimize inventory planning			
	0	0	0
Hours (max 200)	**190**	**0**	**0**

Effect of Card:

Test scripts are restored and no testing hours are lost

”You never expect the Spanish Inquisition” - Monty Python

Case Work Task 9 – Unexpected Events Delivery 2

Unexpected event 1

Your testers complain that no stock levels are returned to the ERP system. When investigating, the integration test lead admits that INT11 failed the test.

You may not test test cycle 5. Return stock level until Delivery 3. In addition, you lose 20 testing hours during Delivery 2, Pass 1.

Unexpected event 3

The financial director has just returned from vacation and she's furious. "We can't go live if invoices don't work", she says.

Thanks to hard work from the development team, the reporting functionality is moved from Delivery 3 to Delivery 2.

You struggle to identify which test scripts that belongs to the reporting requirements and may not test Cycle 10. Print invoices and reports until Delivery 2, Pass 2.

Unexpected event 2

From Delivery 2, Pass 2, your testers start receiving bug fixes for retest from the 3rd party developers.

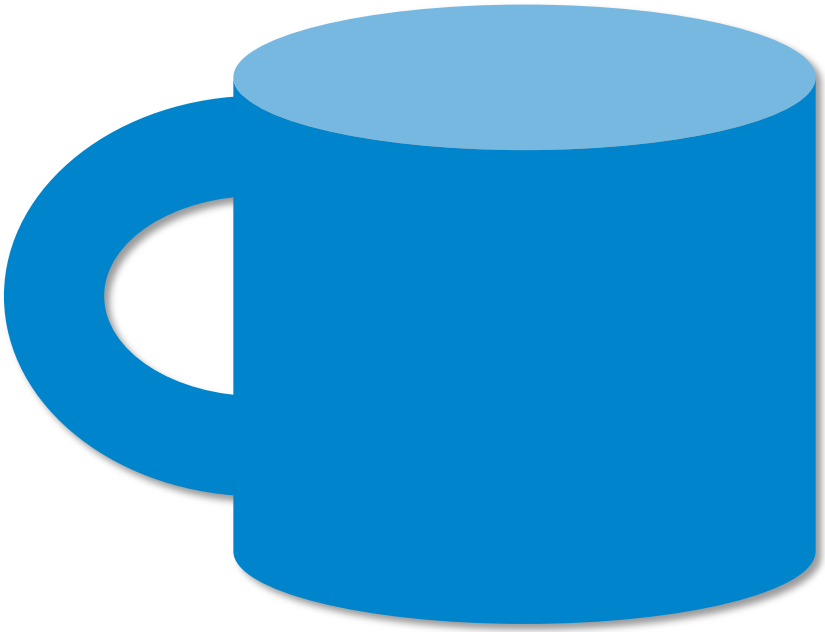
You draw their attention to the defect management process. (You do have one, right?)

Unexpected event 4

The test environment is down on the first testing day of Pass 3. The test environment manager is notified and he will have it up again after lunch.

You lose 20 testing hours during Delivery 2, Pass 3.

Coffee Break!

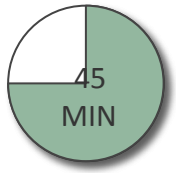


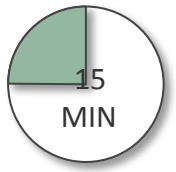
Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Case Work Task 9 – Execute Test, Delivery 3

- In this task you will practice more test control
- **As in previous delivery**, you will have to deal with unexpected events
- **Your testers will** ”execute” the third delivery according to your test schedule as in previous task
- **In the next activity you will present your results in a test report**





Case Work Task 9 – Unexpected Events Delivery 3

Unexpected event 5

Due to a misunderstanding, INT14 Return Status has not been migrated to the test environment.

A new migration is planned for next pass and you may not test Cycle 11. Return order status until Delivery 3, Pass 2.

Unexpected event 7

The solution is now stable enough for test automation.
You review your opportunities for test automation.

Unexpected event 8

The solution contained more defects than expected.

You review your possibilities to identify defect prone areas and prioritizing defects.

Unexpected event 6

One of your tester unfortunately falls sick.

It is not serious but she cannot be replaced and you lose 40 hours during Delivery 3, Pass 2.

Unexpected event 9

Due to a planning mistake, UAT will start before your test is ready.

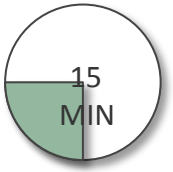
Your test is shortened by one week and only retest may take place in Delivery 3, Pass 3.

Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Case Work Task 10 –Test Report

- **In this task you will practice preparing a test report**
- **With the testing completed**, you will prepare and report the test report
  - Use the test schedule from previous task
  - Report according to your test approach (test metrics etc.)
  - Structure your test report according to IEEE
    - Identifier
    - Summary
    - Variances
    - Assessment
    - Results
    - Evaluation
    - Activities
    - Approvals
- **Present your test report to the class**
  - Adapt your presentation to the audience (project management)
  - Focus on results, risks and issues – not on process (the client is not interested in how you work but on the result of your work)



Test Summary Report

- Test Summary Report Identifier: Test level and software level
- Summary: Information about what was tested
  - Test Items: What was tested?
  - Test Environment: Where was it tested?
  - References: Supporting documents
- Variances: Deviations from agreed scope and quality
  - Specification: What deviated?
  - Reason: Why did it deviate?
  - Support material: Change requests, incident reports etc.
- Assessment:
  - Test coverage: How well was test completed?
  - Test improvements: How was the test changed or how should the test be changed to improve?
- Results:
  - Incidents: By priority, severity, impact
  - Defect patterns: Where were the defects detected?
  - Open incidents: With owner and action
- Evaluation: Quality of software
  - Limitations: Incomplete functions or dropped features
  - Failure likelihood: Risk areas, good quality areas
- Activities: Planned and actual time and costs
- Approvals

Course Agenda

	Day 1	Day 2
08.00-08.30	Introduction	Checkpoint
08.30-09.15	Test Project	Test Techniques
09.15-10.00	Test Management	Case Work 4: Design
10.00-10.15		
10.15-11.00	Case Work: Introduction	Case Work 5: Implementation
11.00-12.00	Case Work 1: Strategy	Case Work 6: Execution
12.00-13.00		
13.00-13.45	Test Metrics	Case Work 7: Status
13.45-14.45	Case Work 2: Plan	Case Work 8: Execution
14.45-15.00		
15.00-15.45	Requirements	Case Work 9: Execution
15.45-16.45	Case Work 3: Analysis	Case Work 10: Closure
16.45-17.00	Day wrap up	Course wrap up

Course Wrap Up

- Topics covered in the course
  - Test Project
  - Test Management
  - Test Strategy
  - Test Plan
  - Test Analysis
  - Test Design
  - Test Implementation
  - Test Execution
  - Test Closure
- Which additional learnings have you done?
- Which learnings are you still missing?

Recommended Training

ISTQB Advanced

- Test Manager
- Test Analyst
- Technical Test Analyst

TMMI (Test Process Assessment)

- TMMI Professional
- TMMI Assessor
- TMMI Lead-Assessor

IREB (Requirements)

- Certified Professional for Requirements Engineering

Leadership Training

- Managing for Growth

Ohter Interests?

- Estimation
- Metrics
- RBT
- Effective Test Case Writing
- Exploratory Testing

Some final words…

- As a test lead you will be in a responsible position
- For testers and test leads, ISTQB states the following code of ethics:
  - Public: Act consistently with the public interest
  - Client and employer: Act in a manner that is in the best interests of the client and employer
  - Judgement: Maintain integrity and independence
  - Management: Subscribe to and promote an ethical approach
  - Profession: Advance the integrity and reputation of the profession
  - Colleagues: Be fair and supportive to colleagues, promote cooperation with developers
  - Self: Participate in lifelong learning regarding your practice

I agree

I do not agree

CONGRATULATIONS
YOU ARE NOW A GREAT TEST LEAD!