



# Software Testing Foundation Level

Lecture 5 – Test Management

Uwe Gühl



# Contents

- 5.1 Test Organization
- 5.2 Test Planning and Estimation
- 5.3 Test Monitoring and Control
- 5.4 Configuration Management
- 5.5 Risks and Testing
- 5.6 Defect Management

# Contents

- 5.1 Test Organization
- 5.2 Test Planning and Estimation
- 5.3 Test Monitoring and Control
- 5.4 Configuration Management
- 5.5 Risks and Testing
- 5.6 Defect Management

# Independent Testing

- Testing tasks may be done by people in
  - a specific testing role,
  - another role like customers
- A certain degree of independence often makes the tester more effective at finding defects due to differences between the author's and the tester's cognitive biases.
- Independence is not a replacement for familiarity
- Developers can efficiently find many defects in their own code.

1.5



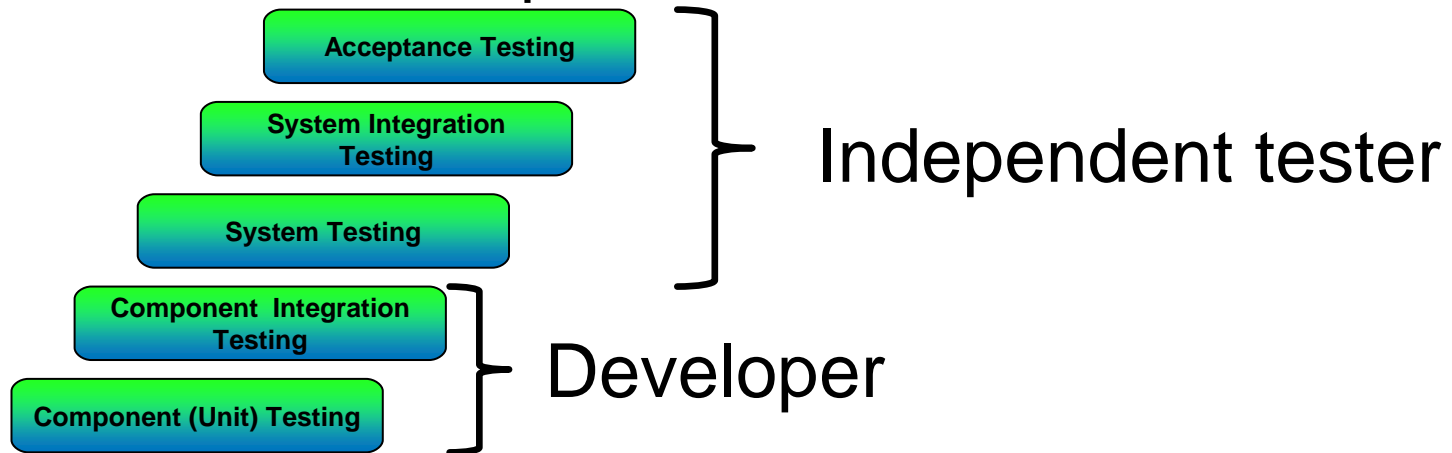
# Independent Testing

- Degrees of independence
  - No independent testers  
Developers testing their own code
  - Independent developers or testers within the development teams or the project team  
Developers testing their colleagues' products
  - Independent test team or group within the organization, reporting to project management or executive management
  - Independent testers, for example
    - from the business organization or user community
    - with specializations in specific test types
  - Independent testers external to the organization  
Working on-site (in-house) or off-site (outsourcing)



# Independent Testing

- Recommended responsibilities for test levels



- Example agile development
  - testers as part of a development team
  - testers as part of an independent test team
  - product owners/requirements engineers perform acceptance testing to validate user stories

# Independent Testing

- Potential benefits of test independence
  - Independent testers find different kinds of failures compared to developers because of their different
    - backgrounds,
    - technical perspectives, and
    - biases
  - An independent tester can verify, challenge, or disprove assumptions made by stakeholders during
    - specification of the system
    - implementation of the system
  - Independent testers of a vendor can report in an upright and objective manner about the system under test without (political) pressure of the company that hired them

# Independent Testing

- Potential drawbacks of test independence include:
  - Isolation from the development team, resulting in
    - lack of collaboration,
    - delays in providing feedback to the development team,
    - adversarial relationship with the development team
  - Developers may lose a sense of responsibility for quality
  - Independent testers may be seen as a bottleneck
  - Independent testers may lack some important information, e.g., about the test object



# Tasks of a Test Manager and Tester

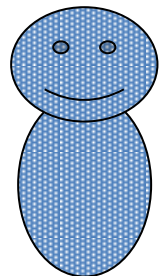
- **Test manager:** The person responsible for project management of testing activities, resources, and evaluation of a test object.
  - Also known as test coach, test coordinator
- **Test leader** (***Synonym: lead tester***):  
On large projects, the person who reports to the test manager and is responsible for project management of a particular test level or a particular set of testing activities.
- **Tester:** A person who performs testing.

# Tasks of a Test Manager and Tester

- Test manager
  - overall responsibility for the test process
  - leadership of the test activities.
  - could be performed by
    - professional test manager,
    - project manager,
    - development manager,
    - quality assurance manager.
- Larger projects or organizations with several test teams
  - Every team headed by a test leader
  - Every team reports to a test manager
- In agile projects test management tasks might be done by testers

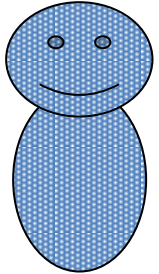
# Tasks of a Test Manager and Tester

- Test manager tasks
  - Develop or review a test policy and test strategy for the organization
  - Plan the test activities by
    - considering the context,
    - understanding the test objectives and risks.
  - Write and update the test plan(s)
    - Selecting test approaches
    - Estimating test time, effort and cost
    - Acquiring resources
    - Defining test levels and test cycles
    - Planning defect management
  - Coordinate the test plan(s) with project managers, product owners, and other stakeholder



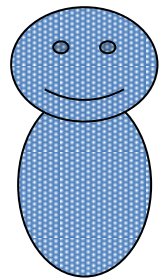
Test  
manager

# Tasks of a Test Manager and Tester

- Test manager tasks
    - Share testing perspectives with other project activities, such as integration planning
    - Initiate test analysis, test design, test implementation, test execution.
    - Monitor test progress and results
    - Check the status of exit criteria (or definition of done)
    - Facilitate test completion activities
    - Prepare and deliver
      - test progress reports
      - test summary reports
    - Adapt planning based on test results and progress
- 
- Test manager

# Tasks of a Test Manager and Tester

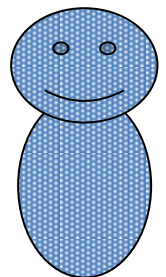
- Test manager tasks
  - Take any actions necessary for test control
  - Support setting up of
    - defect management system
    - configuration management of testware
  - Introduce suitable metrics for
    - measuring test progress
    - evaluating the quality of the testing and the product
  - Tools to support the test process
    - Support the selection and implementation
    - Recommend the budget for tool selection (and possibly purchase and/or support),
    - Allocate time and effort for pilot projects,
    - Provide continuing support in the use of the tool(s)



Test  
manager

# Tasks of a Test Manager and Tester

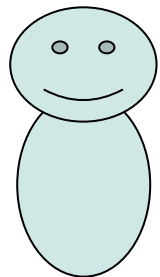
- Test manager tasks
  - Decide about the implementation of test environment(s)
  - Promote and advocate
    - testers,
    - test team,
    - test profession within the organization.
  - Develop the skills and careers of testers, e.g., through
    - training plans,
    - performance evaluations,
    - coaching.



Test  
manager

# Tasks of a Test Manager and Tester

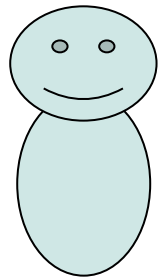
- Tester tasks
  - Review and contribute to test plans
  - Analyze, review, and assess
    - requirements,
    - user stories and acceptance criteria,
    - specifications,
    - models for testability (i.e., the test basis).
  - Identify and document test conditions
  - Capture traceability between
    - test cases,
    - test conditions,
    - test basis.



Tester

# Tasks of a Test Manager and Tester

- Tester tasks
  - Test environment(s)
    - Design
    - Set up
    - Verify
    - Coordinate with system administration and network management
  - Design and implement
    - test cases,
    - test procedures.

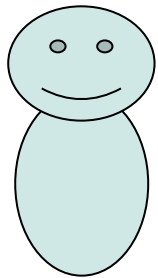


Tester



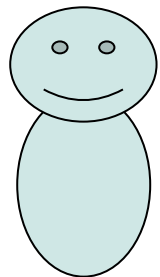
# Tasks of a Test Manager and Tester

- Tester tasks
  - Prepare and acquire test data
  - Create the detailed test execution schedule
  - Execute tests, evaluate the results, and document deviations from expected results
  - Use appropriate tools to facilitate the test process
  - Automate tests as needed, may be supported by
    - developer,
    - test automation expert.
  - Evaluate non-functional characteristics
- Tester– Review tests developed by others



# Tasks of a Test Manager and Tester

- Specialists in the tester role, for example:
  - Component testing level and the component integration testing level: developers.
  - Acceptance test level:
    - business analysts,
    - subject matter experts,
    - users.
  - System test level and the system integration test level: independent test team.
  - Operational acceptance test level:
    - operations
    - systems administration staff.

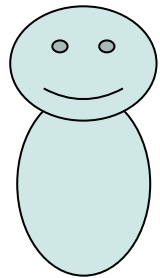


Tester

# Tasks of a Test Manager and Tester

## Excurses

- Best people should test
- Software Testers
  - are real experts after end of the test activities
  - know the software:  
strengths and weaknesses
  - could support
    - as multiplier,
    - for introducing,
    - for training.

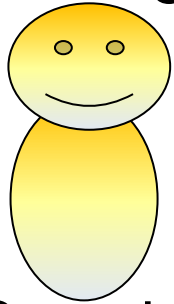


Tester

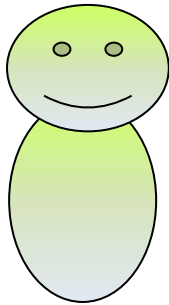
# Tasks of a Test Manager and Tester

## Excurses

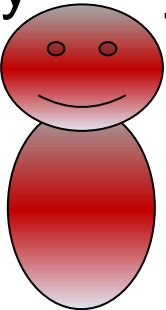
- Possible roles related to test in larger projects or organizations



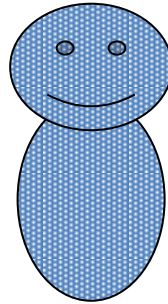
Security tester



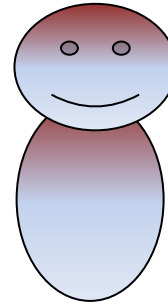
Test data manager



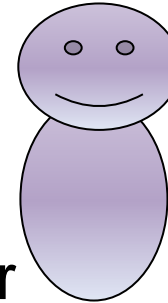
Load and performance tester



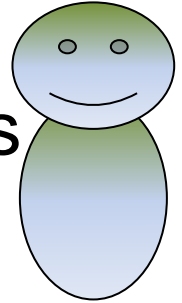
Test manager



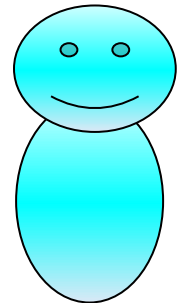
Defect manager



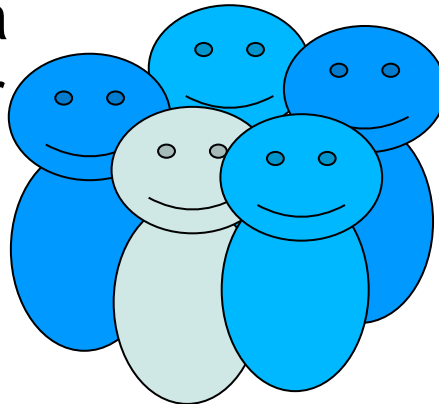
Test environment manager



Quality manager



Test automation expert



Tester

# Summary



- Degrees of independence
  - From very low – No independent testers
  - To very high – Independent testers from an external organization
- Lower test level → Tests by developer  
Higher test level → Tests by independent testers
- Typical roles in testing
  - Test manager
  - Tester

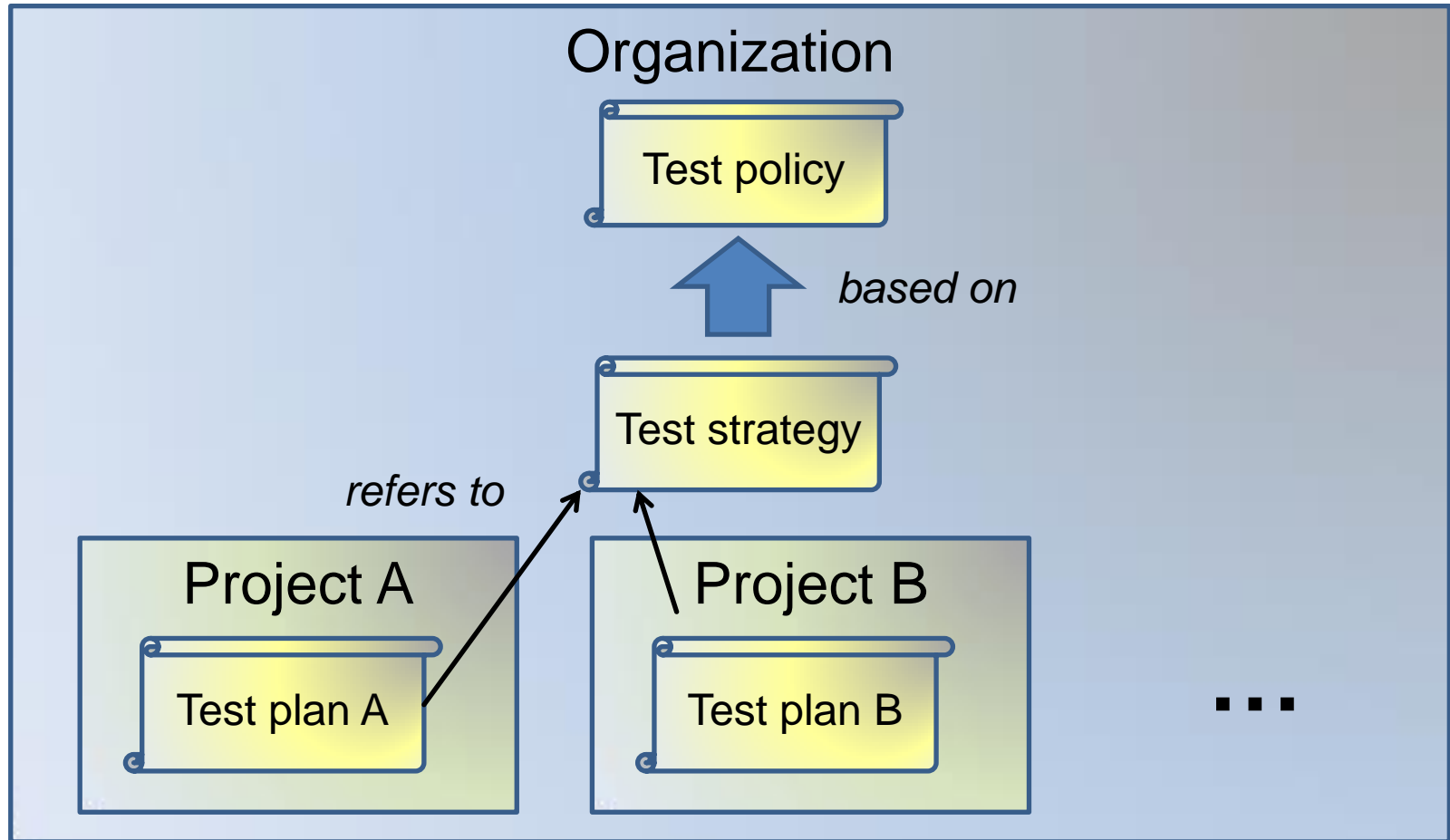
# Contents

- 5.1 Test Organization
- 5.2 Test Planning and Estimation
- 5.3 Test Monitoring and Control
- 5.4 Configuration Management
- 5.5 Risks and Testing
- 5.6 Defect Management

# Purpose and Content of a Test Plan

- **Test policy**  
(**Synonym:** *organizational test policy*):  
A high-level document describing the principles, approach and major objectives of the organization regarding testing.
- **Test strategy**  
(**Synonym:** *organizational test strategy*):  
Documentation aligned with the test policy that describes the generic requirements for testing and details how to perform testing within an organization

# Purpose and Content of a Test Plan





# Purpose and Content of a Test Plan

- Test planning activities
  - Determining
    - scope,
    - objectives,
    - risks of testing.
  - Defining the overall approach of testing
  - Making decisions about
    - what to test,
    - the people,
    - other resources required to perform the various test activities,
    - how test activities will be carried out.

# Purpose and Content of a Test Plan

- Test planning activities
  - Integrating and coordinating the test activities into the software lifecycle activities
  - Scheduling of
    - test analysis,
    - design,
    - implementation,
    - execution,
    - evaluation activities.
  - Scheduling depends on software lifecycle model
    - Sequential development: particular dates
    - Iterative development: context of each iteration

# Purpose and Content of a Test Plan

- Test planning activities
  - Selecting metrics for test monitoring and control
  - Budgeting for the test activities
  - Determining the level of detail and structure for test documentation (e.g., by providing templates or example documents)
- Working products: Test plan/Master test plan
  - The content of test plans vary, and can extend beyond the topics identified above.
  - ISO/IEC/IEEE 29119-3 shows a sample test plan

# Purpose and Content of a Test Plan

## Excurses

Headlines of a test plan following  
ISO/IEC/IEEE 29119-3:2013

1. Overview
2. Document specific information
3. Introduction
4. Context of the testing
5. Testing communication
6. Risk register
7. Test strategy
8. Testing activities and estimates
9. Staffing
10. Schedule

# Test Strategy and Test Approach

- Test strategy
  - generalized description of the test process,
  - product or organizational level.
- **Test approach**: The implementation of the test strategy for a specific project.
- Types of test strategies
  - Analytical:
    - Test team analyzes the test basis to identify the test conditions to cover, analyzes typically based on
      - ❖ requirements,
      - ❖ risks.
    - Example: Risk-based testing

# Test Strategy and Test Approach

- Types of test strategies
  - Model-based
    - **Model-based testing (MBT):**  
Testing based on or involving models.
    - Tests are considering models related to e.g.
      - ❖ a function,
      - ❖ a business process,
      - ❖ an internal structure,
      - ❖ non-functional characteristic.

# Test Strategy and Test Approach

- Types of test strategies
  - Model-based
    - Examples
      - ❖ Business process models,
      - ❖ State models,
      - ❖ **Reliability growth models:**  
A model that shows the growth in reliability over time of a component or system as a result of the defect removal.

# Test Strategy and Test Approach

- Types of test strategies
  - Methodical
    - Systematic use of some predefined set of test conditions
      - ❖ collection of common or likely types of failures,
      - ❖ list of important quality characteristics,
      - ❖ company-wide look-and-feel standards for mobile apps or web pages.
  - Process-compliant (or standard-compliant)
    - Test activities based on external rules and standards like
      - ❖ specified by industry-specific standards,
      - ❖ process documentation,
      - ❖ rigorous identification and use of the test basis,
      - ❖ any process or standard imposed on or by the organization.



# Test Strategy and Test Approach

- Types of test strategies
  - Directed (or consultative)
    - driven primarily by the advice, guidance, or instructions of
      - ❖ business domain experts,
      - ❖ technology experts,
      - ❖ other stakeholders.
    - Stakeholders may be outside the test team or outside the organization itself.
  - Regression-averse
    - desire to avoid regression of existing capabilities.
    - includes
      - ❖ reuse of existing testware (especially test cases and test data),
      - ❖ extensive automation of regression tests,
      - ❖ standard test suites.

# Test Strategy and Test Approach

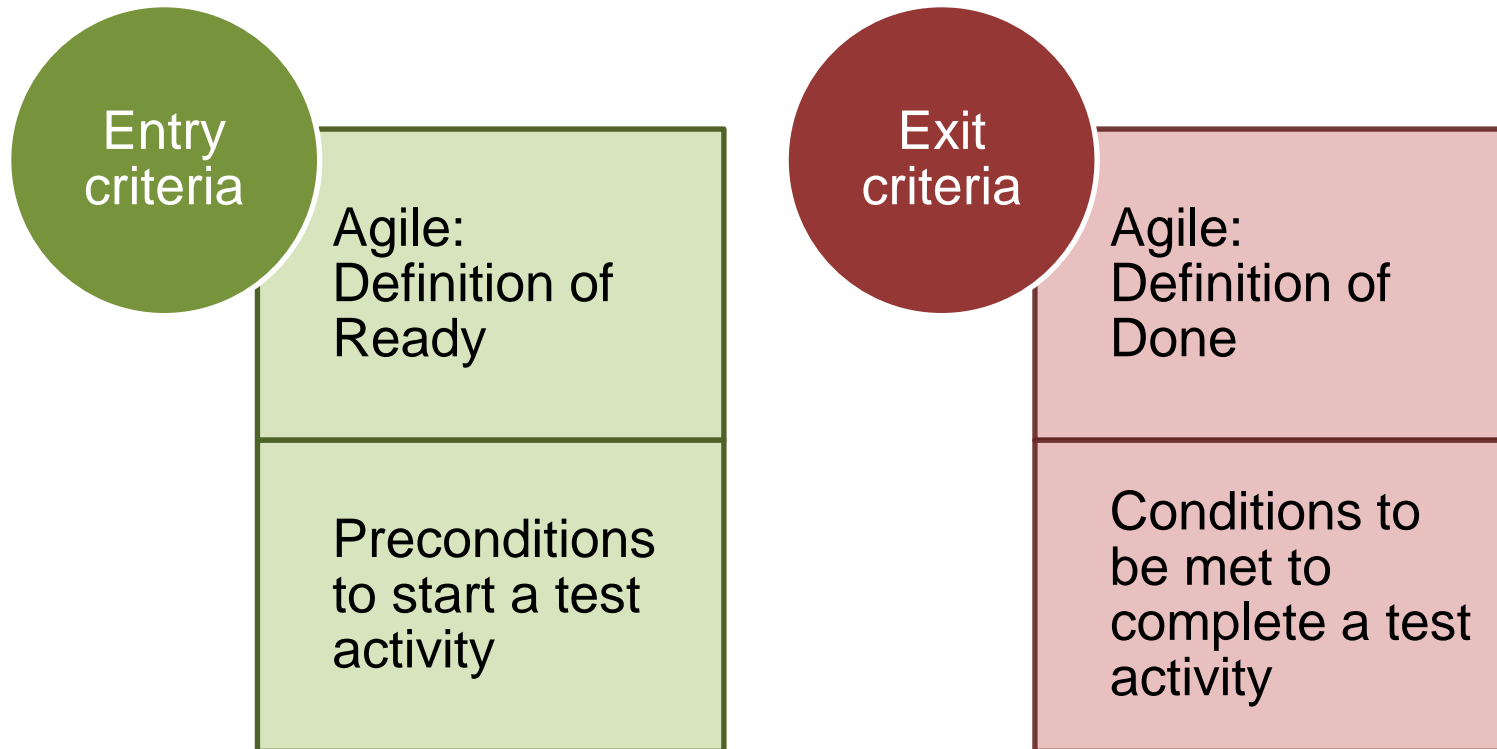
- Types of test strategies
  - Reactive:
    - testing is not pre-planned as preceding strategies but reactive to
      - ❖ component or system being tested,
      - ❖ events occurring during test execution.
    - Tests are designed and implemented, and may immediately be executed in response to knowledge gained from prior test results.
    - Common reactive strategy: exploratory testing

# Test Strategy and Test Approach

- Types of test strategies could be combined to achieve more effective testing, for example
  - risk-based testing (an analytical strategy) with
  - exploratory testing (a reactive strategy).
- Test approach tailors the test strategy for a particular project or release, it should consider the context, e.g.,
  - risks,
  - safety,
  - available resources and skills,
  - technology,
  - nature of the system (e.g., custom-built versus COTS),
  - test objectives,
  - regulations.

# Entry Criteria and Exit Criteria

## (Definition of Ready and Definition of Done)



- to be defined for each test level and test type,
- differ based on the test objectives.

# Entry Criteria and Exit Criteria

## (Definition of Ready and Definition of Done)

- If entry criteria are not met, it is likely that the activity will prove
  - more difficult,
  - more time-consuming,
  - more costly,
  - more risky.

# Entry Criteria and Exit Criteria

## (Definition of Ready and Definition of Done)

- Typical entry criteria:
  - Availability of
    - testable requirements,
    - user stories,
    - models (e.g., when following a model-based testing strategy).
  - Availability of test items that have met the exit criteria for any prior test levels
  - Availability of test environment
  - Availability of necessary test tools
  - Availability of test data and other necessary resources

# Entry Criteria and Exit Criteria

## (Definition of Ready and Definition of Done)

- Typical exit criteria:
  - Planned tests have been executed
  - A defined level of coverage has been achieved, e.g., of
    - requirements,
    - user stories,
    - acceptance criteria,
    - risks,
    - code.
  - The number of unresolved defects is within an agreed limit
  - The number of estimated remaining defects is sufficiently low
  - The evaluated levels of specified quality characteristics are sufficient

# Entry Criteria and Exit Criteria

## (Definition of Ready and Definition of Done)

- Finishing testing without meeting exit criteria might be possible:
  - budget being expended,
  - scheduled time being completed,
  - pressure to bring the product to market.
- Recommended: Project stakeholders and business owners have reviewed and accepted the risk to go live without further testing



# Test Execution Schedule

- defines the order in which test suites are to be run.
- should consider
  - prioritization,
  - dependencies,
  - confirmation tests,
  - regression tests,
  - the most efficient sequence for executing the tests.

# Test Execution Schedule

- Usually: Executing test cases with the highest priority first.
- Challenges
  - Test cases with a higher priority could be dependent on test cases with a lower priority  
→ Lower priority test case must be executed first.
  - Dependencies across test cases must be considered regardless of their relative priorities.
  - Confirmation and regression tests must be prioritized as well, based on the importance of rapid feedback on changes, but here again dependencies may apply.

# Test Execution Schedule

- Example: prioritized test execution schedule after a sprint delivery in an agile project, aligned with product owner

1. Manual and automated execution of smoke tests
2. Retest of fixed critical defects
3. Test preparation of user stories delivered with current sprint
4. Test of user stories delivered with current sprint
5. Support of system integration tests
6. Retest of fixed defects with priority lower than critical
7. Test of delivered user stories, where testing is not completed yet
8. Regression tests

# Factors Influencing the Test Effort

- Product characteristics
  - Risks associated with the product
  - Quality of the test basis
  - Size of the product
  - Complexity of the product domain
  - Requirements for quality characteristics
  - Required level of detail for test documentation
  - Requirements for legal and regulatory compliance

# Factors Influencing the Test Effort

- Development process characteristics
  - Stability and maturity of the organization
  - Development model in use
  - Test approach
  - Tools used
  - Test process
  - Time pressure

# Factors Influencing the Test Effort

- People characteristics
  - Skills and experience of the people involved, especially with similar projects and products (e.g., domain knowledge)
  - Team cohesion and leadership
- Test results
  - Number and severity of defects found
  - Amount of rework required

# Test Estimation Techniques

- **Test estimation**: An approximation related to various aspects of testing.
- Most commonly used techniques are:
  - Metrics-based technique:  
estimating the test effort based on metrics of former similar projects, or based on typical values
  - Expert-based technique:  
estimating the test effort based on the experience of the owners of the testing tasks or by experts

# Test Estimation Techniques

- Metrics-based technique:
  - Burndown chart

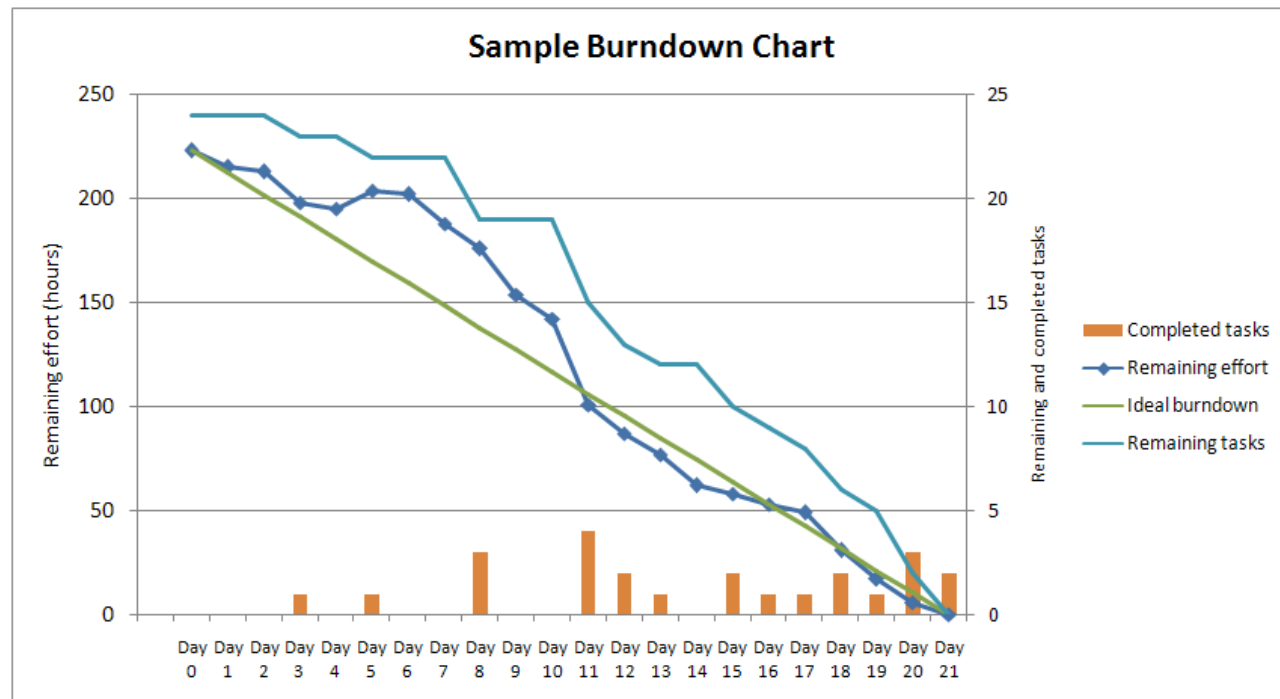


Image source: <https://commons.wikimedia.org/wiki/File:SampleBurndownChart.png>

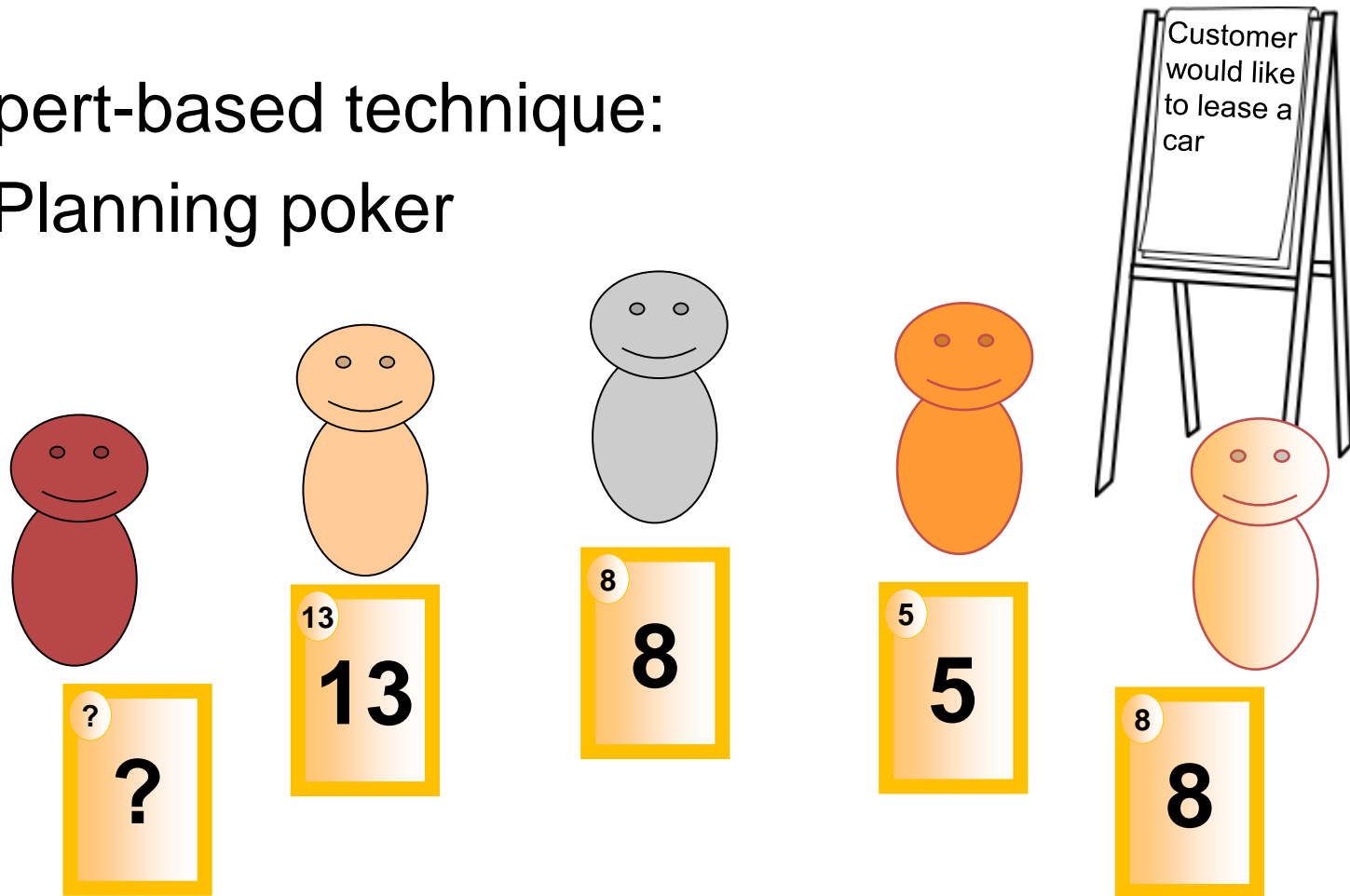


# Test Estimation Techniques

- Metrics-based technique:
  - **Burndown charts**: A publicly displayed chart that depicts the outstanding effort versus time in an iteration. It shows the status and trend of completing the tasks of the iteration.  
The X-axis typically represents days in the sprint, while the Y-axis is the remaining effort (usually either in ideal engineering hours or story points).
    - To determine the amount of work the team can do in the next iteration
  - Defect removal models
    - volumes of defects and time to remove them are captured and reported
    - basis for estimating future projects of a similar nature

# Test Estimation Techniques

- Expert-based technique:
  - Planning poker



# Test Estimation Techniques

- Expert-based technique:
  - **Planning poker**: A consensus-based estimation technique, mostly used to estimate effort or relative size of user stories in Agile software development. It is a variation of the Wideband Delphi method using a deck of cards with values representing the units in which the team estimates.
  - **Wideband Delphi**: An expert-based test estimation technique that aims at making an accurate estimation using the collective wisdom of the team members.

# Summary



- Test plan for a project
  - depends on test strategy and test policy of an organization
  - describes the objective and planned test activities
- Types of test strategies
  - Analytical
  - Model-based
  - Methodical
  - Process-compliant (or standard-compliant)
  - Directed (or consultative)
  - Regression-averse
  - Reactive

# Summary



- Entry criteria:
  - preconditions to start a test activity
  - Agile: Definition of Ready
- Exit criteria:
  - conditions to be met to complete a test activity
  - Agile: Definition of Done
- Test execution schedule defines the order in which test suites are to be run
  - to be prioritized

# Summary



- Factors influencing the test effort
  - characteristics of the product,
  - characteristics of the development process,
  - characteristics of the people,
  - test results
- Test estimation techniques
  - Metrics-based: burndown charts, defect removal models
  - Expert-based: planning poker, wideband delphi.



# Contents

- 5.1 Test Organization
- 5.2 Test Planning and Estimation
- 5.3 Test Monitoring and Control
- 5.4 Configuration Management
- 5.5 Risks and Testing
- 5.6 Defect Management

# Test Monitoring and Control

- Test Monitoring
  - Purpose: gather information and provide feedback and visibility about test activities, collected automatically or manually
  - assess test progress
  - measure if test exit criteria (definition of done) are met, like the achievements of targets for coverage of
    - product risks,
    - requirements,
    - acceptance criteria.



# Test Monitoring and Control

- Test control
  - describes any action taken as a result of test monitoring information
  - Actions may cover any test activity and may affect any other software lifecycle activity.
  - Examples:
    - Re-prioritizing tests when an identified risk occurs (e.g., software delivered late)
    - Changing the test schedule due to availability or unavailability of a test environment or other resources
    - Re-evaluating whether a test item meets an entry or exit criterion due to rework

# Metrics Used in Testing

- Metrics can be collected during and at the end of test activities in order to assess:
  - Progress against the planned schedule and budget
  - Current quality of the test object
  - Adequacy of the test approach
  - Effectiveness of the test activities with respect to the objectives

# Metrics Used in Testing

- Common test metrics
  - Percentage of planned work done in test case preparation (or percentage of planned test cases implemented)
  - Percentage of planned work done in test environment preparation
  - Test case execution
    - number of test cases run/not run,
    - test cases passed/failed,
    - test conditions passed/failed.
  - Defect information
    - defect density,
    - defects found and fixed,
    - failure rate,
    - confirmation test results.

# Metrics Used in Testing

- Common test metrics
  - Test coverage of
    - requirements,
    - user stories,
    - acceptance criteria,
    - risks,
    - code
  - Task completion, resource allocation and usage, and effort
  - Cost of testing, including the cost compared to the benefit of finding the next defect or the cost compared to the benefit of running the next test

# Purposes, Contents, and Audiences for Test Reports

- Purpose of test reports: summarize and communicate test activity information
  - during a test activity
    - Test progress reports
  - at the end of a test activity, typically related to a test level, when exit criteria are reached
    - Test summary report
- ISO standard ISO/IEC/IEEE 29119-3
  - refers to both types of test reports\*
  - contains structures and examples for each type.

*\* The standard refers to “test completion reports” instead of “test summary reports”*

# Purposes, Contents, and Audiences for Test Reports

- Test progress reports may include:
  - The status of the test activities and progress against the test plan
  - Factors impeding progress
  - Testing planned for the next reporting period
  - The quality of the test object

# Purposes, Contents, and Audiences for Test Reports

- Test summary reports include:
  - Summary of testing performed
  - Information on what occurred during a test period
  - Deviations from plan, including deviations in schedule, duration, or effort of test activities
  - Status of testing and product quality with respect to the exit criteria or definition of done
  - Factors that have blocked or continue to block progress
  - Metrics of defects, test cases, test coverage, activity progress, and resource consumption.
  - Residual risks
  - Reusable test work products produced



# Purposes, Contents, and Audiences for Test Reports

- Contents of test summary reports vary depending on
  - project,
  - organizational requirements,
  - software development lifecycle.
- Examples
  - Complex project with many stakeholders or a regulated project compare to a quick software update.
  - Daily stand-up test progress reporting in agile development might be incorporated into
    - task boards,
    - defect summaries,
    - burndown charts.



# Purposes, Contents, and Audiences for Test Reports

- Test reports should be tailored based on
  - context of the project
  - report's audience, e.g.,
    - technical audience  
detailed information on defect types and trends
    - executive summary report  
high-level report with
      - ❖ test conditions passed/failed/not tested,
      - ❖ status summary of defects by priority,
      - ❖ budget,
      - ❖ schedule.

# Summary



- Test monitoring with focus on
  - Test case execution status
  - Defect overview
  - Coverage
- Test control takes action as a result of test monitoring information
- Test reports
  - summarize and communicate test activity information
  - Test progress reports and test summary report
  - Tailoring depending on context and audience required

# Contents

- 5.1 Test Organization
- 5.2 Test Planning and Estimation
- 5.3 Test Monitoring and Control
- 5.4 Configuration Management
- 5.5 Risks and Testing
- 5.6 Defect Management

# Configuration Management

- **Configuration**: The composition of a component or system as defined by the number, nature, and interconnections of its constituent parts.
- **Configuration item**: An aggregation of work products that is designated for configuration management and treated as a single entity in the configuration management process.
- **Configuration management**: A discipline applying technical and administrative direction and surveillance to identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify that it complies with specified requirements.

# Configuration Management

- Purpose of configuration management: establish and maintain the integrity of
  - the component or system,
  - the testware, and
  - their relationships to one anotherthrough the project and product lifecycle.

# Configuration Management

- To support testing with configuration management
  - All test items are uniquely identified, version controlled, tracked for changes, and related to each other
  - All items of testware are uniquely identified, version controlled, tracked for changes, related to each other and related to versions of the test item(s) so that traceability can be maintained throughout the test process
  - All identified documents and software items are referenced unambiguously in test documentation
- During test planning, configuration management procedures and infrastructure (tools) should be identified and implemented.

# Summary



- Use configuration management to establish and maintain the integrity of
  - the component or system,
  - the testware, and
  - their relationships to one anotherthrough the project and product lifecycle.

# Contents

- 5.1 Test Organization
- 5.2 Test Planning and Estimation
- 5.3 Test Monitoring and Control
- 5.4 Configuration Management
- 5.5 Risks and Testing
- 5.6 Defect Management



# Definition of Risk

- **Risk**: A factor that could result in future negative consequences.
  - **Product risk**:  
A risk impacting the quality of a product.
  - **Project risk**:  
A risk that impacts project success.
- **Risk level** (**Synonym**: *risk exposure*):  
The qualitative or quantitative measure of a risk defined by impact and likelihood.
- The level of risk **R** is determined by  $R = P * I$ 
  - **P** (probability/likelihood of the event)
  - **I** (estimated impact (the harm) from that event)

# Definition of Risk

- Risk level: Proposal how to quantify risk R
  - P (probability/likelihood of the event)
    - 3 = high
    - 2 = possible
    - 1 = low
  - I (estimated impact (the harm) from that event).
    - 3 = very critical
    - 2 = critical
    - 1 = less critical
  - $R = P * I$
  - Possible values for R:
    - **9 and 6 = high risks**
    - **4 and 3 = medium risks**
    - **2 and 1 = low risks**

# Product and Project Risks

- Product risk
  - possibility that a work product may fail to satisfy the legitimate needs of its users and/or stakeholders like
    - specification,
    - component,
    - system,
    - test.
  - Quality risk:  
A product risk related to a quality characteristic.

# Product and Project Risks

- Examples of product risks:
  - Software might not perform its intended functions according to the specification
  - Software might not perform its intended functions according to user, customer, and/or stakeholder needs
  - A system architecture may not adequately support some non-functional requirement(s)
  - A particular computation may be performed incorrectly in some circumstances
  - A loop control structure may be coded incorrectly
  - Response-times may be inadequate for a high-performance transaction processing system
  - User experience (UX) feedback might not meet product expectations

# Product and Project Risks

- Project risk involves situations that, should they occur, may have a negative effect on a project's ability to achieve its objectives
- Examples of project risks:
  - Project issues:
    - Delays in
      - ❖ delivery,
      - ❖ task completion,
      - ❖ satisfaction of exit criteria/definition of done.

# Product and Project Risks

- Examples of project risks:
  - Project issues:
    - inadequate funding because of
      - ❖ inaccurate estimates,
      - ❖ reallocation of funds to higher priority projects,
      - ❖ general cost-cutting across the organization.
    - Substantial re-work because of late changes

# Product and Project Risks

- Examples of project risks:
  - Organizational issues:
    - Skills, training, and staff may not be sufficient
    - Conflict and problems because of personnel issues
    - Missing resources, due to conflicting business priorities, like
      - ❖ users,
      - ❖ business staff,
      - ❖ subject matter experts.

# Product and Project Risks

- Examples of project risks:
  - Political issues:
    - Communication issues  
Testers may not communicate their needs and/or the test results adequately
    - Effects missing  
Developers and/or testers may fail to follow up on information found in testing and reviews (e.g., not improving development and testing practices)
    - Improper attitude  
toward, or expectations of, testing (e.g., not appreciating the value of finding defects during testing)



# Product and Project Risks

- Examples of project risks:
  - Technical issues:
    - Requirements
      - ❖ not defined well enough
      - ❖ not met, given existing constraints
    - Test environment not ready on time
    - Delay of migration topics like
      - ❖ data conversion,
      - ❖ migration planning,
      - ❖ tool support.

# Product and Project Risks

- Examples of project risks:
  - Technical issues:
    - Weaknesses in the development process may impact the consistency or quality of project work products like
      - ❖ design,
      - ❖ code,
      - ❖ configuration,
      - ❖ test data,
      - ❖ test cases.
    - Poor defect management and similar problems may result in accumulated defects and other technical debt

# Product and Project Risks

- Examples of project risks:
  - Supplier issues:
    - A third party may
      - ❖ fail to deliver a necessary product or service,
      - ❖ go bankrupt.
    - Contractual issues

# Risk-based Testing and Product Quality

- Test activities based on risk
  - to decide where and when to start testing
  - to identify areas that need more attention.
  - Risk mitigation to reduce
    - the probability of an adverse event occurring and/or
    - the impact of an adverse event.
- **Risk analysis:**  
The overall process of risk identification and risk assessment.
  - The resulting product risk information is used to guide
    - test planning,
    - the specification,
    - preparation and execution of test cases,
    - test monitoring and control.

# Risk-based Testing and Product Quality

- Based on product risk analysis
  - Determine the test techniques to be employed
  - Determine the particular levels and types of testing to be performed, such as
    - security testing,
    - accessibility testing
  - Determine the extent of testing to be carried out
  - Prioritize testing in an attempt to find the critical defects as early as possible
  - Determine about risk reducing activities  
Example: providing training to inexperienced designers

# Risk-based Testing and Product Quality

- Risk management activities to minimize the likelihood of a product failure
  - Analyze and re-evaluate on a regular basis what can go wrong
    - Update status of risks and risk levels
    - In case: identify of new risks
  - Determine which risks are important to deal with
  - Implement actions to mitigate those risks
  - Make contingency plans to deal with the risks should they become actual events

# Summary



- Differentiate between product risks and project risks
- Risk  $R = \text{probability } P * \text{estimated impact } I$
- Example for risk levels
  - high risk
  - medium risk
  - low risk
- Risk management activities
  - to minimize the likelihood of a product failure,
  - to mitigate the estimated impact,
  - should start with high risks.
- Risk-based testing as professional approach:  
All testing activities and resources are based on risk types and risk levels.

# Contents

- 5.1 Test Organization
- 5.2 Test Planning and Estimation
- 5.3 Test Monitoring and Control
- 5.4 Configuration Management
- 5.5 Risks and Testing
- 5.6 Defect Management



# Defect Management

- **Defect management**:  
The process of recognizing, recording, classifying, investigating, resolving and disposing of defects.
- **Defect report** (**Synonym**: *bug report*):  
Documentation of the occurrence, nature, and status of a defect.
- Standard ISO/IEC/IEEE 29119-3 shows an example of the contents of a defect report\*

\* *The standard refers to “incident report” instead of “defect report”*

# Defect Management

- **Anomaly** :  
Any condition that deviates from expectation based on requirements specifications, design documents, user documents, standards, etc., or from someone's perception or experience.
- Anomalies may be found during, but not limited to, reviewing, testing, analysis, compilation, or use of software products or applicable documentation.
- Anomaly as a result of a failing test  
→ needs to be investigated.
- Objectives of testing: finding defects  $\Rightarrow$  defects to be logged.
- Logging of defects depends on
  - context of the component or system being tested,
  - test level,
  - software development lifecycle model.

# Defect Management

- Defect lifecycle – defects to be tracked
    - Identification
    - Investigation
    - Classification
    - Resolution like
      - correction and successful confirmation testing of fix,
      - deferral to a subsequent release,
      - acceptance as a permanent product limitation.
- ⇒ Defect management process;  
stakeholders should agree

# Defect Management

- Defects may be reported during
  - coding,
  - static analysis,
  - reviews,
  - during dynamic testing,
  - use of a software product.
- Beware of false positives that are not result of a defect in the system under test, like network errors

# Defect Management

- Defects may be reported for issues in
  - code,
  - working systems,
  - in any type of documentation including
    - requirements,
    - user stories and acceptance criteria,
    - development documents,
    - test documents,
    - user manuals,
    - installation guides.
- In the defect management process, standards could be defined for
  - attributes,
  - classification,
  - workflow of defects.

Defects found during reviews, will normally be documented different, e.g., in review meeting notes

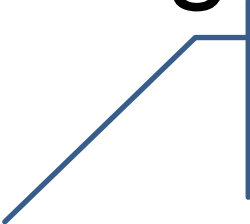
# Defect Management

- Objectives of a defect report
  - Provide developers information to enable them
    - to identify specific effects,
    - to isolate the problem with a minimal reproducing test,
    - to correct the potential defect(s), as needed
    - to otherwise resolve the problem.
  - Provide test managers a means of tracking the quality of the work product and the impact on the testing.  
Example: A lot of defects are reported,
    - Testers need time reporting them instead of running tests,
    - More confirmation testing needed.
  - Provide ideas for development and test process improvement

# Defect Management

- Contents of a defect report

- An identifier
- A title and a short summary of the defect being reported
- Date of the defect report, issuing organization, and author
- Identification of the test item (configuration item being tested) and environment
- The development lifecycle phase(s) in which the defect was observed
- A description of the defect to enable reproduction and resolution, including logs, database dumps, screenshots, or recordings (if found during test execution)
- Expected and actual results
- Scope or degree of impact (severity) of the defect on the interests of stakeholder(s)
- Urgency/priority to fix



Tools offer to automatically insert attributes like Id, author, history

# Defect Management

- Contents of a defect report
  - **Severity**: The degree of impact that a defect has on the development or operation of a component or system.
    - based on the interests of stakeholder(s)
  - **Priority**: The level of (business) importance assigned to an item, e.g., defect.
    - Here: Urgency/priority to fix



In practice severity and priority are often used similar, but originally they got difference meanings



# Defect Management

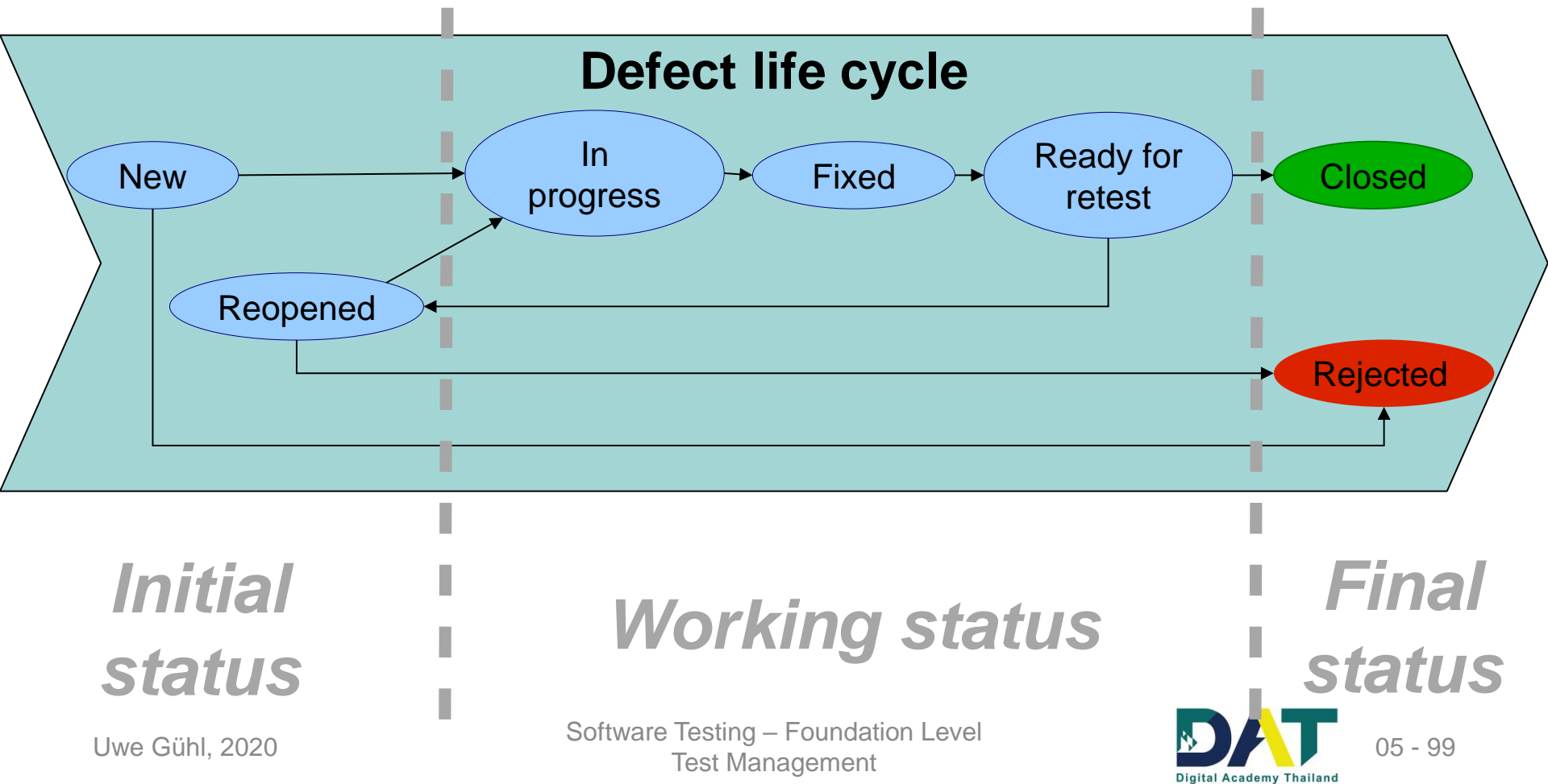
- Contents of a defect report
  - Example for severity classifications
    - Severity 1: Critical  
Total system outage; system upgrade failed  
(e.g. system does not boot); restore not possible
    - Severity 2: Major  
Data migration too slow; excessive number of alarms;  
sporadic system re-starts; loss of synchronization
    - Severity 3: Minor  
Incomplete list of commands; documentation issues
    - Severity 4: Trivial  
Cosmetic problems; not well structured printouts

# Defect Management

- Contents of a defect report
  - State of the defect report, for example
    - open,
    - deferred,
    - duplicate,
    - waiting to be fixed,
    - awaiting confirmation testing,
    - re-opened,
    - closed.

# Defect Management

- Example for status of defect reports



# Defect Management

- Contents of a defect report
  - Conclusions, recommendations and approvals
  - Global issues, such as other areas that may be affected by a change resulting from the defect
  - Change history, such as the sequence of actions taken by project team members with respect to the defect to isolate, repair, and confirm it as fixed
  - References, including the test case that revealed the problem

# Defect Management

- Rules for defect reports
  - **Bad** bug reports are reports
    - that say nothing ("It doesn't work!"),
    - that make no sense,
    - that don't give enough information,
    - that give wrong information,
    - of problems that turn out to be
      - ❖ user error,
      - ❖ the fault of somebody else's program,
      - ❖ network failures.
  - **Good** bug reports:  
Wonderfully clear, helpful, informative

*Source: Simon Tatham: How to Report Bugs Effectively, 1999,  
<http://www.chiark.greenend.org.uk/~sgtatham/bugs.html>*

# Defect Management

- Rules for defect reports
    - Show a defect directly to the developer
    - Describe a defect so it could be reproduced.  
Best: Step by step, use screenshots, videos
    - Describe what you expected and what you got  
What works and what went wrong?
    - Notice contents of error messages, esp. numbers
    - Report the symptoms
      - **Must:** What are actual facts  
"I was at the computer and this happened"
      - **Could:** What are speculations, your ideas as proposal  
"I think the problem might be this"
- Source: Simon Tatham: How to Report Bugs Effectively, 1999,  
<http://www.chiark.greenend.org.uk/~sgtatham/bugs.html>*

# Defect Management

- Rules for defect reports
  - Try to work around for intermittent faults and inform about version, operating system, etc.
    - Try other machines, web browsers, screen resolution
    - Does it depend on the size of files you use, other programs you use in parallel?
  - Try to help that the defect could be fixed
    - Provide extra information on request like version numbers
    - Special activities, so that developer could locate the defect

*Source: Simon Tatham: How to Report Bugs Effectively, 1999,  
<http://www.chiark.greenend.org.uk/~sgtatham/bugs.html>*

# Defect Management

- Rules for defect reports
  - Write clearly and as neutral as possible
  - Be specific.
    - Not:** "I selected Load"
    - Better:** "I clicked on Load", or "I pressed Alt-L"
  - Be verbose
    - If you write one sentence only, developer must ask and ask
  - Be careful of pronouns
    - Not:** "I started FooApp. It put up a warning window. I tried to close it and it crashed."
    - Better:** "I started FooApp, which put up a warning window. I tried to close the warning window, and FooApp crashed."
  - Read what you wrote
    - Try to reproduce a listed sequence of actions yourself
  - Don't joke

*Source: Simon Tatham: How to Report Bugs Effectively, 1999,  
<http://www.chiark.greenend.org.uk/~sgtatham/bugs.html>*



# Defect Management

- Example for a defect report

Defect ID	SDM-1104
Defect headline	Only 6 characters of enrollment number could be entered
Priority	major
Status	New
Reported:	16.02.2020 06:53
Modified	16.02.2020 10:31
Reported by:	Joe Doo
Assigned to	Sam Ying
Version	SDM Sprint 9
Component	Data entering module
Attachments	Screenshot1.jpg Screenshot2.jpg
Links	Test case SDM-0007 Create Student data set

# Defect Management

- Example for a defect report

<b>Description</b>	<p>Precondition: Enrollment number 4055-991 already given</p> <ol style="list-style-type: none"><li>1. Called SDM system</li><li>2. Called "Add student"</li><li>3. Entered last name "Daowang"</li><li>4. Entered first name "Rid"</li></ol> <p><b>Expected behavior:</b> Entering enrollment number 4055-991 is possible</p> <p><b>Actual behavior:</b> It is only possible to enter the first 6 characters: "4055-9".</p> <p>Hint: Press [submit] works, but number in system is "4055-9" instead of "4055-991"</p>
--------------------	--

# Summary



- Defect management is relevant for the project
- Defect reports offer information for
  - the developer to fix a defect,
  - the test manager concerning the quality of a system.
- Defect reports should focus on a neutral complete description to reproduce a defect
- Status of defect reports help to control the defect management process