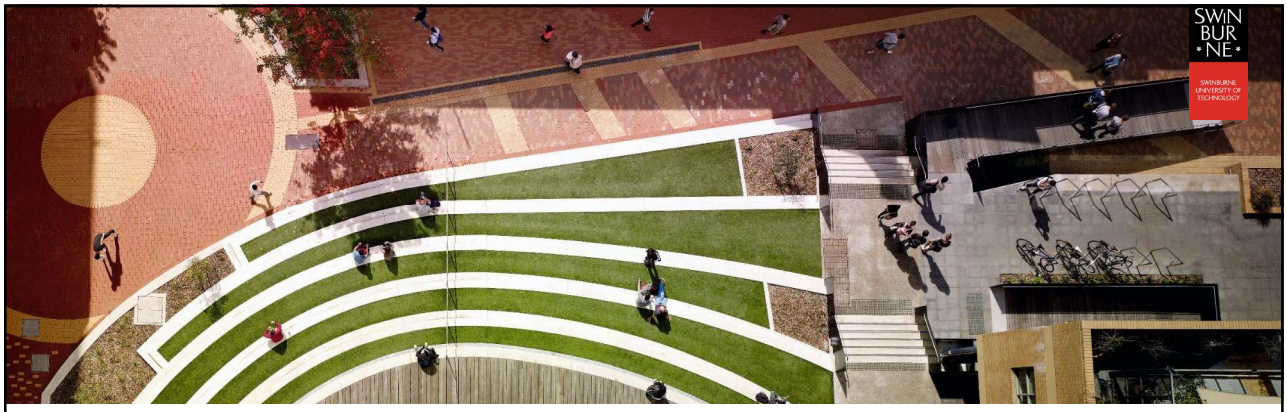


COS80022 – Software Quality and Testing

Week 6 – Testing Management and Tools

Outline

- Test Planning
- Test Monitoring and Control
- Test Tools

Test Organisation

Test Organisation

- Test management is often critical to success or failure of a software development project
- Testing infrastructure needs to be established

Independent Testing

- Very difficult to review your own material
- Testing tasks may be done by people in a specific testing role
- A certain degree of independence often makes the tester more effective at finding defects
- The more complex and safety critical a system is, the more important it is to have independence
- Varies depending on the SDLC

Levels of Test Independence

- No independent testers
- Independent developers or testers within the development team or the project team
- Independent test team or group within the organisation
- Independent testers from the business organisation or user community, or with specialisation in specific test types
- Independent testers external to the organisation, either working on-site or off-site

Benefits of Independence

- Independent testers are likely to recognise 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 and implementation of the system

Drawbacks of Independence

- Isolation from the development team, leading to a lack of collaboration, delays in providing feedback to the development team, or an adversarial relationship with the development team
- Developers may lose a sense of responsibility of quality
- Independent testers may be seen as a bottleneck or may be blamed for delays in the release of the software
- Independent testers may lack some important information

Role of Test Manager

- Tasked with overall responsibility for the test process and successful leadership of the test activities
- Might be performed by different resources
  - In large projects or organisations, several test teams may report to a test manager, test coach, or test coordinator, each team being headed by a test leader or lead tester
- Test manager role varies with the SDLC, e.g., in agile:
  - Team handles some of the tasks
  - Test managers outside team are sometimes called test coaches

Tasks for Test Manager

- Develop or review a test policy and strategy
- Plan the test activities by considering the context, the test objectives and risks
- Write and update the test plan(s)
- Co-ordinate the test plan(s) with project managers, project owners and others
- Share testing perspectives (such as integration planning)

Tasks of Test Manager (cont'd)

- Initiate the analysis, design, implementation and execution of tests, monitor test progress and results, and check the status of exit criteria
- Prepare and deliver test progress reports and test summary reports
- Adapt planning, based on test results and progress
- Support setting up the defect management system and adequate configuration management of testware

Tasks of Test Manager (cont'd)

- Introduce metric for measuring test progress and evaluating the quality
- Support the selection and implementation of tool to support the test process
- Promote and advocate the testers, the test team, and the test profession
- Develop the skills and careers of testers

Role of Testers

- Different people; resources for specific test types or test automation
  - Component testing & component integration testing: Developers often carry out the tester's role
  - System testing & system integration testing: An independent test team often does the role of a tester
  - Acceptance testing: Business analysts, subject matter experts, and users often do the role of a tester
  - Operation acceptance testing: operations and/or systems administration staff often do the role of a tester

Tasks of Tester

- Review and contribute to test plans
- Analyse, review and assess the test basis for testability
- Identify and document test conditions, and capture traceability
- Design, set up, and verify test environment(s)
- Design and implement test cases and test procedures
- Prepare and acquire test data

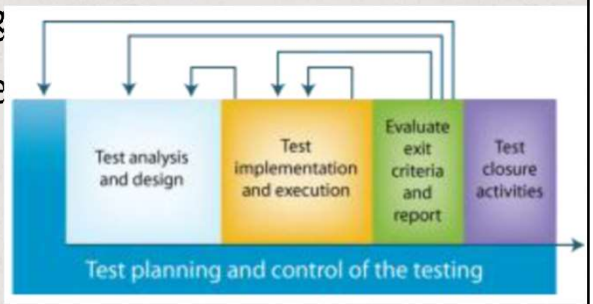
Tasks of Tester (cont'd)

- Create the detailed test execution schedule
- Execute tests, evaluate results, and document deviations from expected results
- Use appropriate tools to facilitate the test process
- Automate tests as needed
- Evaluate non-functional characteristics
- Review tests developed by others

Test Planning

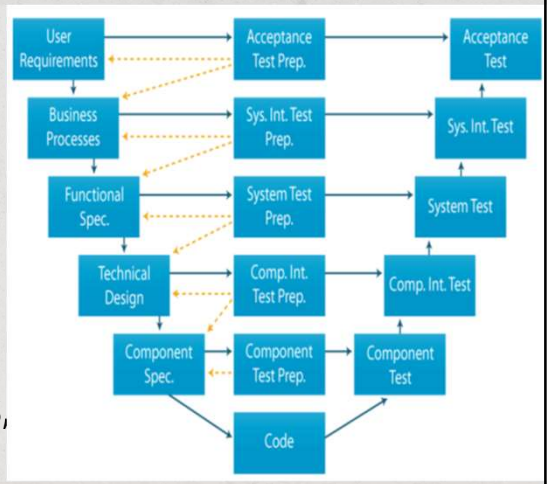
Test Planning

- Focuses the mind on what is required in order to test and which **approach** will be follows
- **Test strategy** will influence the test plan in terms of techniques and level of effort &
- **Risk-based test** strategy is use to inform planning process
- Greater the risk the more rigorous testing techniques



Test Planning

- A test plan outlines test activities and is influenced by the test policy and test strategy, the development lifecycles and methods being used
- Planning may be documented in a master test plan and in separate test plans for test levels, or for separate test types



Activities of Test Planning

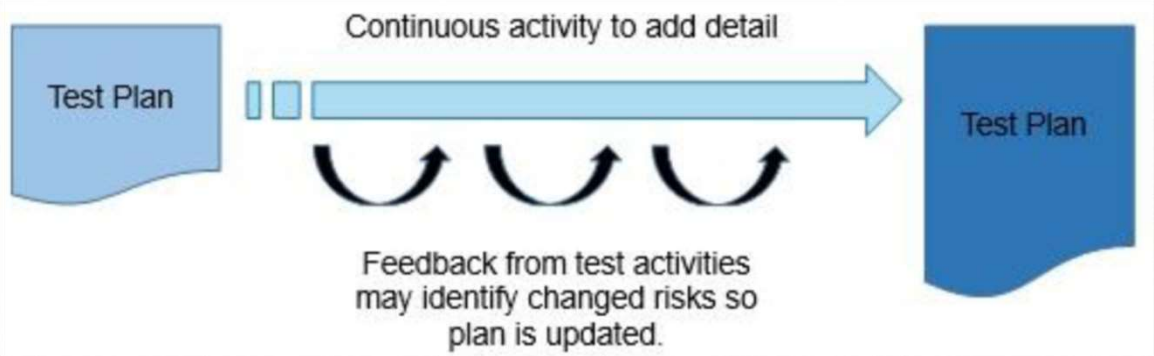
- Determining the scope, objectives, and risks of testing
- Defining the overall approach of testing
- Integrating and coordinating the test activities into the software lifecycle activities
- **Making decisions** about what to test, the people and other resources required and how test activities will be carried out

Activities of Test Planning (cont'd)

- **Scheduling** of test analysis, design, implementation, execution, and evaluation activities
- Selecting **metrics** for test monitoring and control
- **Budgeting** for the test activities
- Determining the level of detail and structure for test documentation

Test Planning

- A continuous activity and is performed throughout the product's lifecycle



Test Strategies

- Analytical: risk-based – test high risk functionality first
- Model-based: based on model of required aspect of product
- Methodical: systematic use of some predefined set of tests
- Process-compliant: external rules and standards
- Directed: advice, guidance, instructions of stakeholders/experts
- Regression averse: reuse of existing testware; automation
- Reactive: rather than pre-planned

Test Approach

- Tailor the test strategy for a particular project or release based on decisions made around:
  - Complexity and goals of the project
  - Type of product being developed
  - Product risk analysis
  - Context; Risks; Safety
  - Available resources & skills; Technology
  - Nature of system
  - Test objectives; Regulations

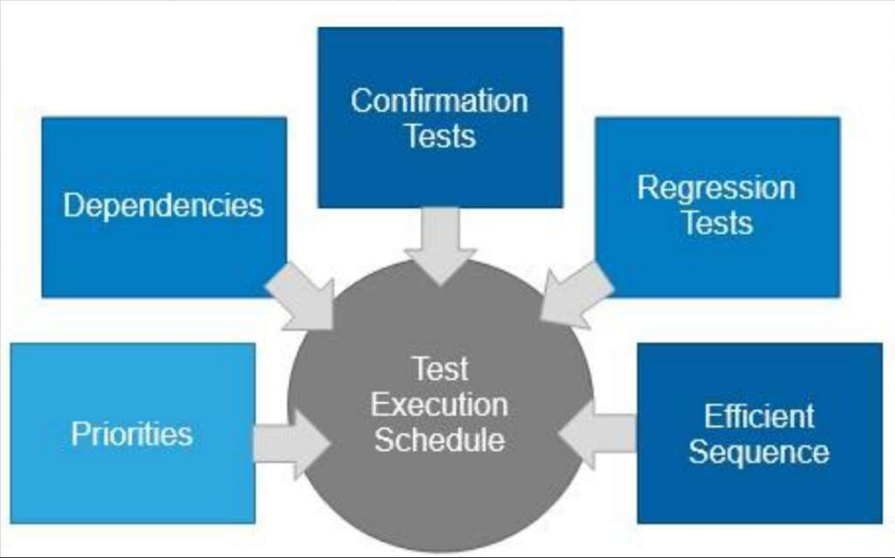
Entry Criteria (Definition of Ready)

- Availability of testable requirements, user stories, and/or models
- 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

Exit Criteria (Definition of Done)

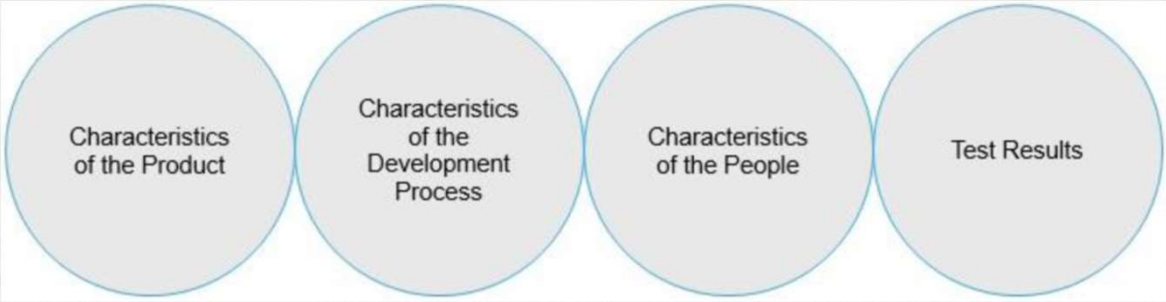
- Planned tests have been executed
- A defined level of coverage has been achieved
- The number of unresolved defects is within an agreed limit
- The number of estimated remaining defects is sufficiently low
- The evaluated levels of reliability, performance efficiency, usability, security, and other relevant quality characteristics are sufficient

Test Execution Schedule



Test Effort

- Test effort estimation is to predict the amount of test-related work to meet the objectives of the testing for a particular project, release, or iteration



Product Characteristics

- Risks associated with the product
- 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

Development Process Characteristics

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

People Characteristics

- Skills and experience of the people involved
- Team cohesion
- Leadership
- In agile:
  - All in the team are responsible for quality
  - Open communication
  - Well organised teams
  - Appropriate skills

Test Results

- Depending on the results produced by the testing teams, there may be changes required in the testing effort such as:
  - The number and severity of defects found
  - The amount of rework required

Test Estimation

- Two of the common techniques used to determine the effort required for testing:
  - Metrics-based: previous projects
  - Expert-based: experts to give an educated guess

Expert-Based Approach for Test Estimation

- If you have no previous metrics
- Have to rely on the task owner or another expert
- Planning poker
- Wideband Delphi

33

Metrics-Based Approach for Test Estimation

- Use previous projects as a guide to estimate the current one
- Need metrics that have been recorded for other projects
- Typical metrics recorded:
  - Duration of test preparation/execution
  - Number of test cycles/defects found
  - How long defects took to be resolved
- Burndown charts
  - Feeds into the team's velocity

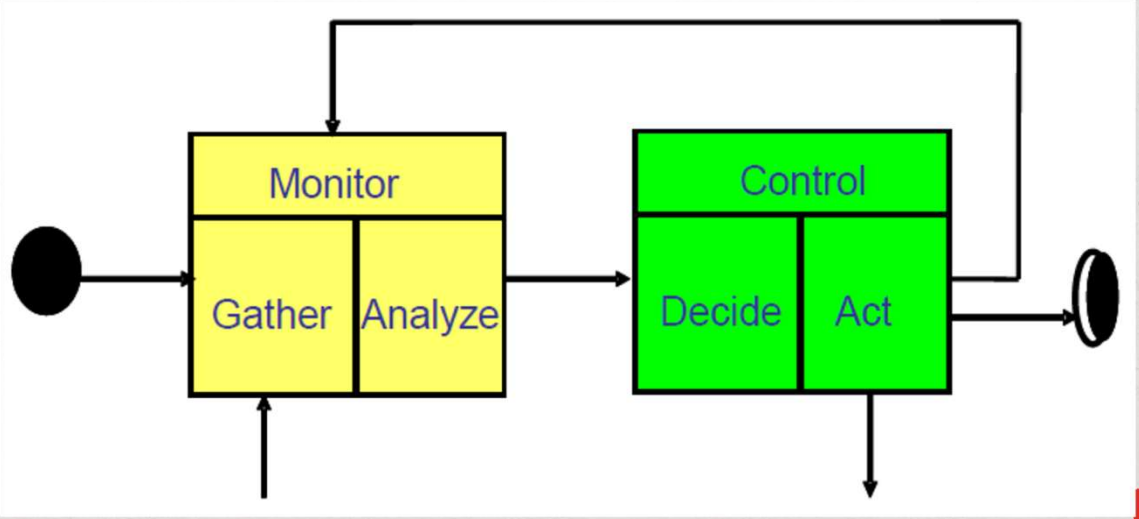
34

Estimation Challenges

- Estimates may need to be revised after the schedule is agreed. Factors that affect estimates:
  - Down time
  - Late delivery of code
  - Incorrect recording of previous metrics
  - Are project similar in size and complexity?
  - Same number of resources?

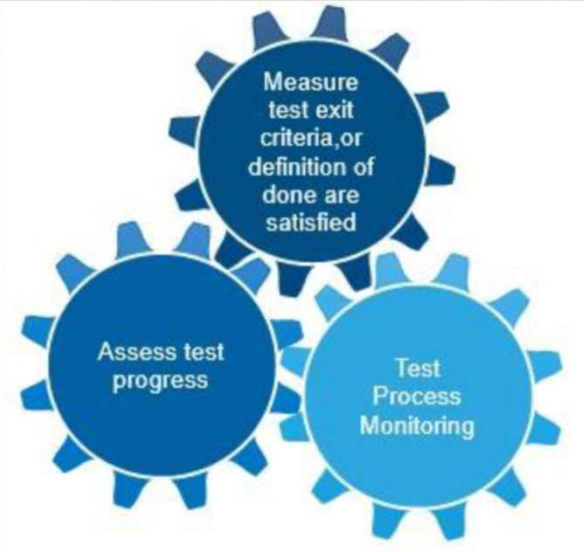
Test Monitoring and Control

Test Monitoring and Control



Test Monitoring

- To gather information and provide feedback and visibility about test activities



Test Monitoring

- "A test management task that deals with developing and applying a set of corrective actions to get a test project on track when monitoring shows a deviation from what was planned"
- Monitoring tasks:
  - On-going activity comparing actual progress against plan (monitoring)
  - Will respond to information gathered by testing as well as changing conditions by the project (control) activities
  - Includes updating the plan as needed

Test Control

- Describe any guiding or corrective actions taken as a result of information and **metrics** gathered. For example:
  - Re-prioritising 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

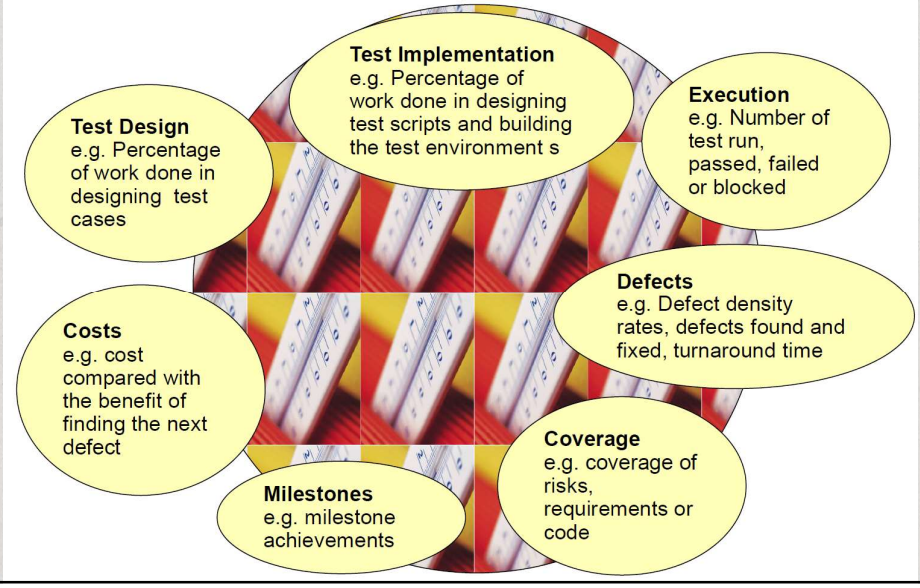
Use of Test Metrics

- 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 their objectives

Common Test Metrics

- Percentage of planned work done in test case preparation
- Percentage of planned work done in test environment preparation
- Test case execution, e.g., number of test cases passed/failed
- Defect information, e.g., defects found
- Test coverage of requirements, user stories, acceptance criteria, risks, or code
- Test completion, resource allocation and usage, and effort
- Cost of testing

Test Control Activities



Test Reporting

- To summarise and communicate test activity information, both during and at the end of a test activity, e.g., a test level
- Typical test progress reports may also include:
  - 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

Typical Contents of Test Progress Report

- 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

Test Summary Report

- When exit criteria are reached, a test summary report is issued
- Summarise information of what happened during a period of testing
- Contents vary depending on the project, the organisational requirements, and SDLC
- Should be tailored based on the context of the project and the report's audience

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 compliance with specified requirements"

Configuration Management and Testing

- Test items: uniquely identified, version controlled, tracked for changes and related to each other
- Testware items: uniquely identified, version controlled, tracked for changes, related to each other and related to versions of the test items for traceability
- All identified documents and software items are referenced unambiguously in test documentation
- Procedures and tools should be selected, documented and implemented during the test planning phase

Risk & Risk-Based Testing

Risk

- Risk: the chance of an event, hazard, threat or situation occurring and resulting in undesirable consequence or a potential problem
  - Product risk: A risk impacting the quality of a product
  - Project risk: A risk that impacts project success
- Level of risk is calculated by assessing likelihood and impact

Risk

- Product risks (quality risks)
  - Software might not perform its intended functions
  - Software might not perform its intended functions according to user needs
  - A system architecture may not adequately support some non-functional requirement(s)
  - A particular computation may be performed incorrectly
  - A loop control structure may be coded incorrectly
  - Response times may be inadequate
  - User experience might not meed product expectations

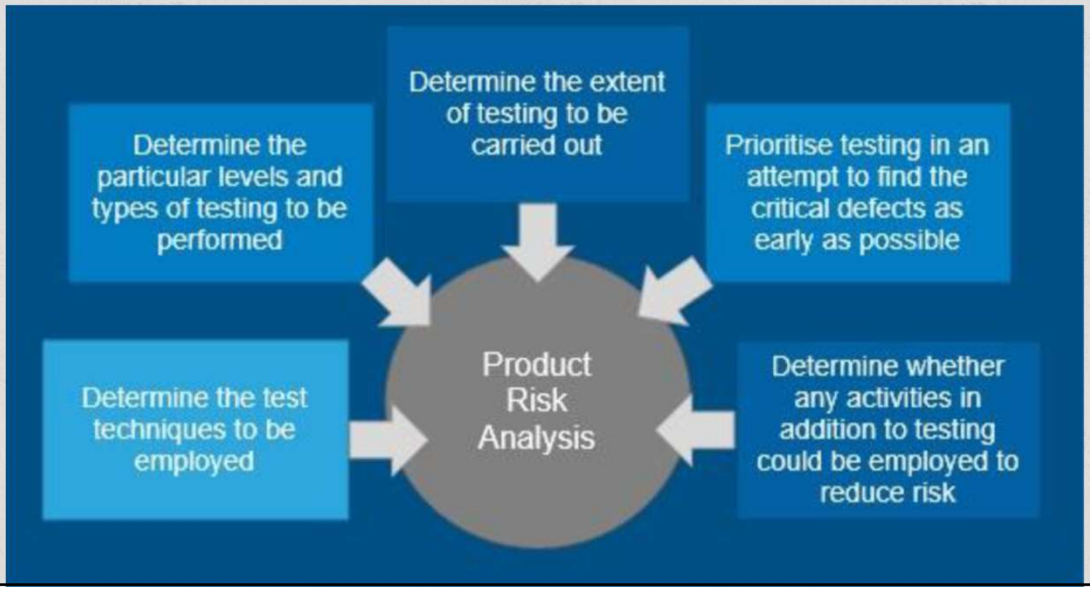
Risk

- Project risks: may affect both development and test activities
- Project managers are responsible for handling project risks
- Test managers may have responsibility for test-related project risks
- Examples: project issues; organisational issues, political issues; technical issues; supplier issues

Risk-Based Testing

- Risks are used to
  - Focus effort required during testing
  - Decide where to test and areas to concentrate on
  - Reduce the probability of adverse events occurring
  - Reduce impact of adverse events
- Risk-based testing approach
  - Involves product risk analysis
  - Provides guidance to other testing activities or tasks like test planning, the specification, preparation and execution of test cases, and test monitoring and control

Product Risk Analysis



Risk Management

- Risk management activities provide a disciplined approach to
  - Analyse and re-evaluate on a regular basis what can go wrong
  - 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
- Testing may also identify new risks

Defect Management

Defect

- Defects: discrepancies between actual and expected results during execution can be caused by faults in different areas such as:
  - Design
  - Documentation (requirements, specification)
  - Programming
  - Testware
  - Environment
  - Hardware

Defect Management

- Defect logging may vary, depending on the context of the component or system being tested, the test level, and the SDLC model
- Any defects identified should be investigated and should be tracked from discovery to their resolution
- Defects may be reported during coding, static analysis, reviews, dynamic testing, or use of a software product

Defect Management

- Defects may be reported for issues in code or working system, or in any type of documentation
- In order to have an effective and efficient defect management process, organisations may define standards for the attributes, classification, and workflow of defects
- Attempt to minimise the number of false positives reported as defects

Defect Management

- Defect report objectives
  - Provide developers and other parties with information about any adverse event that occurred, to enable them to identify specific effects, to isolate the problem with a minimal reproducing test, and to correct the potential defect(s), as needed or to otherwise resolve the problem
  - Provide test managers a means of tracking the quality of the work product and the impact on the testing
  - Provide ideas for development and test process improvement

Defect Management

- Defect report attributes
  - Identifier, title, short summary
  - Date, issuing organisation, author
  - Identification of test item and environment
  - Development lifecycle phase(s) in which defect was observed
  - Description to enable reproduction and resolution
  - Expected and actual results
  - Scope or degree of impact (severity)
  - Urgency/priority to fix
  - State of the defect
  - Conclusions, recommendations and approvals
  - Global issues, change history, references

Guidelines for writing a bug report

- Open a new bug report for each issue
- Summary: Write a clear summary so triager/developer understand it
  - "Software crashes" – Bad
  - "Cancelling a File Copy dialog crashes File Manager" – Good
- Component: which part of the software does the bug exist in?
- Version: which version of the software can the bug be reproduced with
  - Helps narrow down the root cause of bug
  - Helps identify the commit that introduced the bug
- OS: On which operating system did you find the bug? All or a specific OS

https://developer.mozilla.org/en-US/docs/Mozilla/QA/Bug_writing_guidelines

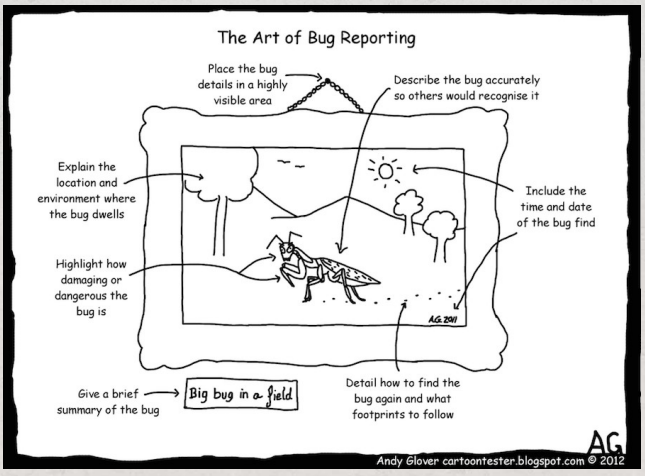
Guidelines for writing a bug report

- Description: details of the problem
  - Overview of the bug (detailed restatement of summary)
  - Build id – which build has the bug
  - Additional builds that are affected by the same bug
- Steps to reproduce: easy-to-follow steps that will trigger the bug
- Actual Results: what the application does after performing the above steps
- Expected Results: what the application should have done, were the bug not present

Art of Bug Reporting

- **Summary: "Big Bug in the field"**

- Describe bug accurately so others would recognize it
- Include date/time bug was found
- Detail how to reproduce the bug
- Highlight the impact of the bug
- Explain the location/environment of bug
- Make these details visible to entire team

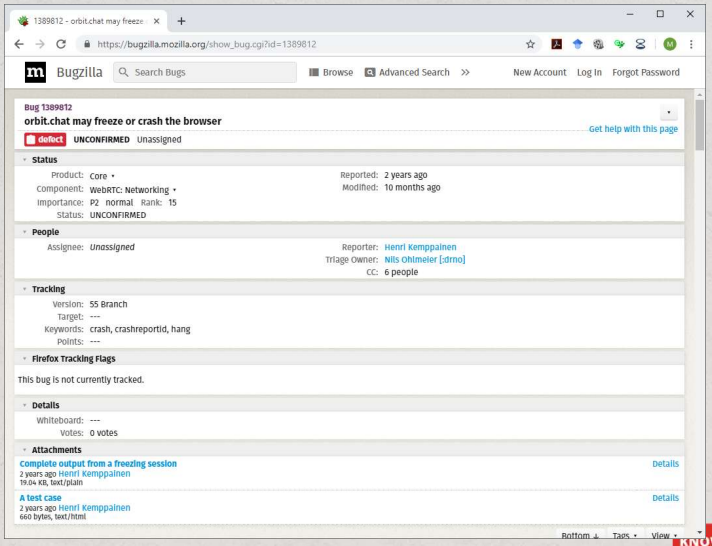


Source: https://usersnap.com/blog/easy-bug-tracking-hacks-developers/

Example Bug Report

- About a bug in the WebRTC Component
- Raised by user/developer
- Includes:
  - Steps to reproduce
  - Expected behaviour
  - Actual behaviour
  - Stack trace or error message (if any)

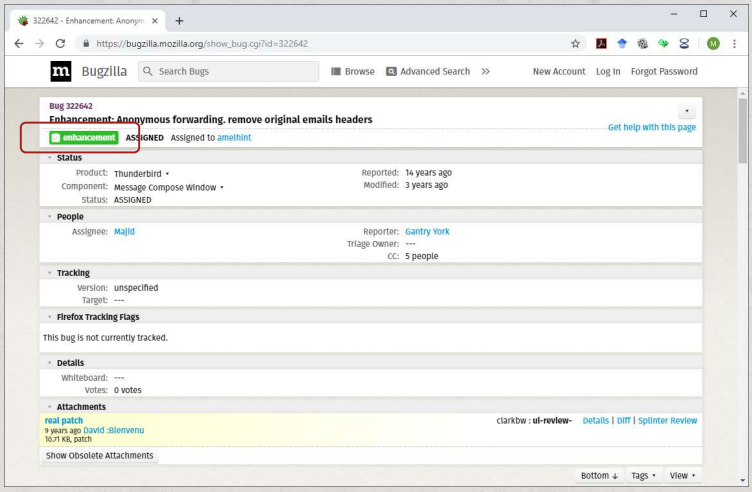
  (Refer to link for details)



https://bugzilla.mozilla.org/show_bug.cgi?id=1389812

Example Feature Request

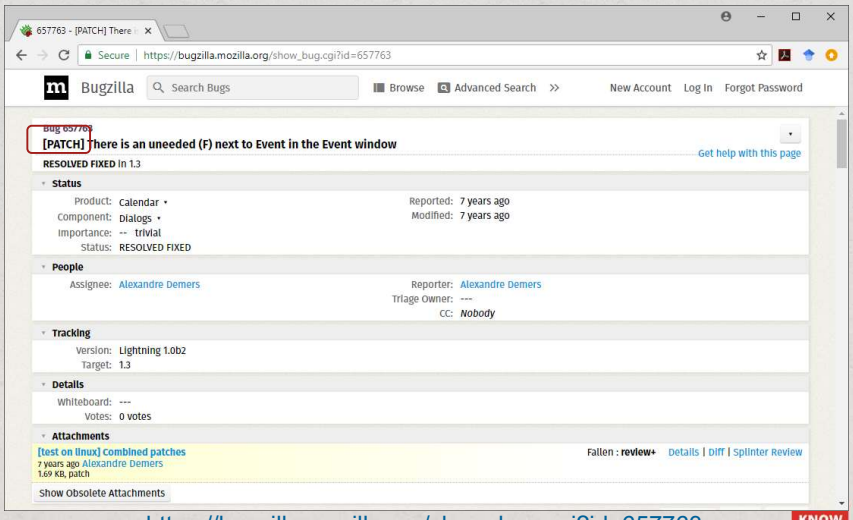
- About a new feature to be added to the software system
- E.g. add sorting to results (by modified time)
- Raised by user/developer



https://bugzilla.mozilla.org/show_bug.cgi?id=322642

Example Patch

- A fix to the software
- Raised by a professional user, or a developer on an important fix
- Should include:
  - Version to patch
  - Patch code: basically a code diff, Diff (buggy, correct)



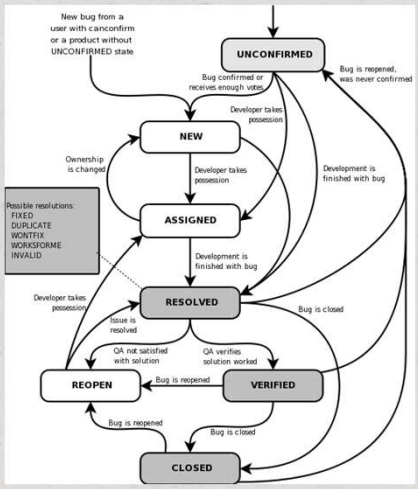
https://bugzilla.mozilla.org/show_bug.cgi?id=657763

Bug/Issue Tracking

- Once a bug has been reported, it goes through different states before it gets resolved

Issue Lifecycle

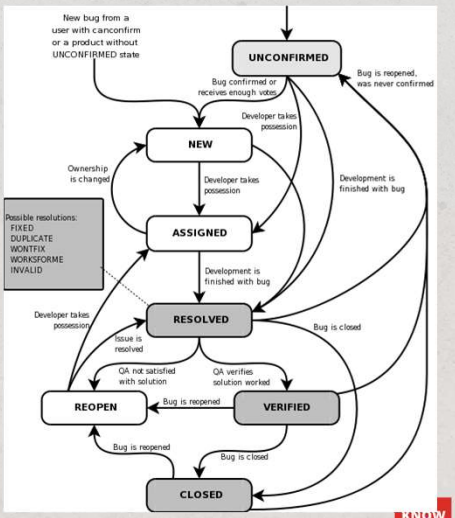
- Unconfirmed:
  - Default status for public bugs – waiting for someone to validate, reproduce, or otherwise confirm that it is a bug
- Open/New
  - When a new bug/feature request is logged and posted for the first time it is assigned NEW status
- Assigned
  - Once the bug is logged, the triager assigns issue to a developer
    - Bug report: developer tries to reproduce the bug and then tries to fix it
    - Feature request: discuss with team on whether to accommodate request and implement the feature
    - Patch: Validate the patch



Source: https://www.bugzilla.org/docs/3.6/en/html/lifecycle.html

Issue Lifecycle

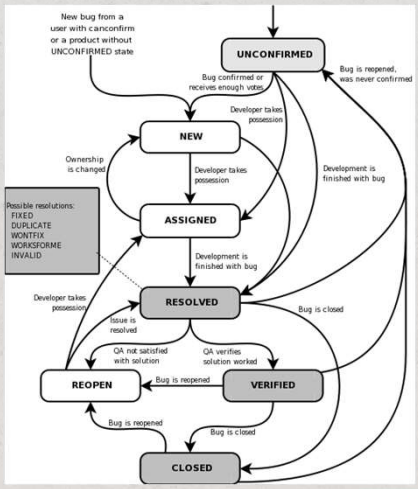
- Resolved: When the decision on the issue is made in any way
  - Fixed – usually accompanied by code commits
    - Bug is fixed, feature is added or patch is applied
  - Invalid
    - Bug cannot be reproduced, feature does not make sense, patch is not correct
  - Wont fix (Deferred)
    - Not high priority, deferred to next release
  - Duplicate
    - Some systems automatically merge duplicate issues
  - WorksForMe (Not A Bug)
    - Expected behaviour incorrectly reported as a bug



Source: https://www.bugzilla.org/docs/3.6/en/html/lifecycle.html

Issue Resolution

- Verified: The fix has been verified by testing
- Closed: The issue has been closed
- Reopened: If something happens after the issue is closed, and it becomes active again
  - Incomplete fix
  - Start to implement a postponed feature
  - Revoke any wrong decision before



Source: https://www.bugzilla.org/docs/3.6/en/html/lifecycle.html

Tool Support for Testing

Test Tool Considerations

- Activities that tools can support:
  - Directly used in testing: test execution & test data preparation tools
  - Help manage requirements, test cases, test procedures, automated test scripts, test results, data, defects process and report/monitor testing
  - Used for investigation & evaluation
  - Any tool that aids in testing, e.g., spreadsheet

Purpose of Tool Support

- Improve efficiency by automating repetitive tasks
- Improve efficiency of activities that support manual test activities
- Improve quality of test activities by more consistent testing
- Automate activities that cannot be executed manually
- Increase reliability of testing

Tool Classification

- Tools can be classified on several criteria, such as purpose, technology used and the test activities they support
- Support for one or more than one activity
- Intrusive tools can affect the actual outcome of the test (probe effect), e.g. performance tool may affect response time
- Many tools are more appropriate for developers

Tool Types

- For management of testing and testware
- For static testing
- For test design and implementation
- For test execution and logging
- For performance measurement and dynamic analysis
- For specialised testing needs

Tools for Management of Testing and Testware

- Test management tool & application lifecycle management tools (ALM)
- Requirements management tools
- Defect management tools
- Configuration management tools
- Continuous integration tools

Tool Support for Static Testing

- Tools that support reviews
  - Assist with review processes, checklists, review guidelines
  - Store and communicate review comments
  - Reports on defects and effort
- Static analysis tools
  - Help developers and testers find defects prior to dynamic testing
  - Support for enforcing coding standards
  - Support for planning or risk analysis by providing metrics for code
  - Used by developers during the review and debugging of their code

Tool Support for Test Design & Implementation

- Test design tools
- Model-based testing tools
- Test data preparation tools
- Acceptance test driven development & Behaviour driven development tools
- Test driven development tools

Tool Support for Test Execution and Logging

- Test execution tools
- Coverage tools
- Test harness
- Unit test framework tools

Tool Support for Performance Measurement & Dynamic Analysis

- Performance testing tools
- Monitoring tools
- Dynamic analysis tools

Tools Support for Specialised Testing Needs

- Data quality assessment
- Data conversion & migration
- Usability testing
- Accessibility testing
- Localisation testing
- Security testing
- Portability testing

Potential Benefits of Test Automation

- Easier access to information about testing
- More objective assessment
- Greater repeatability and consistency
- Reduce repetitive work

Potential Risks of Tools

- Unrealistic expectations for the tool
- Under estimating the initial time cost and effort
- Under estimating the effort to achieve significant and continuing benefits
- Effort to maintain test assets
- Tool over reliance
- Neglecting version control
- Neglecting relationship and interoperability issues between tools
- Vendor may go out of business, retire or sell the tool to another vendor
- Poor response from vendor for support, upgrades
- Risk of suspension of open source
- A new platform or technology may not be supported
- No clear ownership of the tool

Tool Selection Considerations

85

Proof of Concept Evaluation

- Establish whether the tool performs effectively
- Identify any changes needed
- Carry out on site
- On a small project
- Determine if potential benefits will be realised

86

Pilot Project

- Gain knowledge
- Evaluate fit with processes and practices
- Decide on standards
- Assess if benefits will be achieved
- Determine metrics and configuration

87

Success Factors for Tools

- Rolling out the tool incrementally
- Adapting and improving processes to fit
- Defining guidelines for the use of the tool
- Implementing a way to gather usage information from the actual use of the tool
- Monitoring tool use and benefits
- Providing support to the users of a given tool
- Gathering lessons from all users

88

Tools in COS80022

- Pytest
  - Week 6, 7, 8
- Jmeter
  - Week 10

Summary

- Test planning
- Test monitoring and control
- Risk-based testing
- Tool support for testing

References

- International Software Testing Qualifications Board. *ISTQB Foundation Level (Core) Syllabus*

Next

- Tutorial
  - Pytest
- Next week
  - Black-box testing