5.1 Test Organization (K2) — *30 minutes*

Terms
Tester, test leader, test manager

5.1.1 Test Organization and Independence (K2)

The effectiveness of finding defects by testing and reviews can be improved by using independent testers. Options for independence include the following:
- o No independent testers; developers test their own code
- o Independent testers within the development teams
- o Independent test team or group within the organization, reporting to project management or executive management
- o Independent testers from the business organization or user community
- o Independent test specialists for specific test types such as usability testers, security testers or certification testers (who certify a software product against standards and regulations)
- o Independent testers outsourced or external to the organization

For large, complex or safety critical projects, it is usually best to have multiple levels of testing, with some or all of the levels done by independent testers. Development staff may participate in testing, especially at the lower levels, but their lack of objectivity often limits their effectiveness. The independent testers may have the authority to require and define test processes and rules, but testers should take on such process-related roles only in the presence of a clear management mandate to do so.

The benefits of independence include:
- o Independent testers see other and different defects, and are unbiased
- o An independent tester can verify assumptions people made during specification and implementation of the system

Drawbacks include:
- o Isolation from the development team (if treated as totally independent)
- o Developers may lose a sense of responsibility for quality
- o Independent testers may be seen as a bottleneck or blamed for delays in release

Testing tasks may be done by people in a specific testing role, or may be done by someone in another role, such as a project manager, quality manager, developer, business and domain expert, infrastructure or IT operations.

5.1.2 Tasks of the Test Leader and Tester (K1)

In this syllabus two test positions are covered, test leader and tester. The activities and tasks performed by people in these two roles depend on the project and product context, the people in the roles, and the organization.

Sometimes the test leader is called a test manager or test coordinator. The role of the test leader may be performed by a project manager, a development manager, a quality assurance manager or the manager of a test group. In larger projects two positions may exist: test leader and test manager. Typically the test leader plans, monitors and controls the testing activities and tasks as defined in Section 1.4.

Typical test leader tasks may include:
- o Coordinate the test strategy and plan with project managers and others
- o Write or review a test strategy for the project, and test policy for the organization

© International Software Testing Qualifications Board

- Contribute the testing perspective to other project activities, such as integration planning
- Plan the tests – considering the context and understanding the test objectives and risks – including selecting test approaches, estimating the time, effort and cost of testing, acquiring resources, defining test levels, cycles, and planning incident management
- Initiate the specification, preparation, implementation and execution of tests, monitor the test results and check the exit criteria
- Adapt planning based on test results and progress (sometimes documented in status reports) and take any action necessary to compensate for problems
- Set up adequate configuration management of testware for traceability
- Introduce suitable metrics for measuring test progress and evaluating the quality of the testing and the product
- Decide what should be automated, to what degree, and how
- Select tools to support testing and organize any training in tool use for testers
- Decide about the implementation of the test environment
- Write test summary reports based on the information gathered during testing

Typical tester tasks may include:

- Review and contribute to test plans
- Analyze, review and assess user requirements, specifications and models for testability
- Create test specifications
- Set up the test environment (often coordinating with system administration and network management)
- Prepare and acquire test data
- Implement tests on all test levels, execute and log the tests, evaluate the results and document the deviations from expected results
- Use test administration or management tools and test monitoring tools as required
- Automate tests (may be supported by a developer or a test automation expert)
- Measure performance of components and systems (if applicable)
- Review tests developed by others

People who work on test analysis, test design, specific test types or test automation may be specialists in these roles. Depending on the test level and the risks related to the product and the project, different people may take over the role of tester, keeping some degree of independence. Typically testers at the component and integration level would be developers, testers at the acceptance test level would be business experts and users, and testers for operational acceptance testing would be operators.

5.2 Test Planning and Estimation (K3) — *40 minutes*

Terms
Test approach, test strategy

5.2.1 Test Planning (K2)

This section covers the purpose of test planning within development and implementation projects, and for maintenance activities. Planning may be documented in a master test plan and in separate test plans for test levels such as system testing and acceptance testing. The outline of a test-planning document is covered by the 'Standard for Software Test Documentation' (IEEE Std 829-1998).

Planning is influenced by the test policy of the organization, the scope of testing, objectives, risks, constraints, criticality, testability and the availability of resources. As the project and test planning progress, more information becomes available and more detail can be included in the plan.

Test planning is a continuous activity and is performed in all life cycle processes and activities. Feedback from test activities is used to recognize changing risks so that planning can be adjusted.

5.2.2 Test Planning Activities (K3)

Test planning activities for an entire system or part of a system may include:

- Determining the scope and risks and identifying the objectives of testing
- Defining the overall approach of testing, including the definition of the test levels and entry and exit criteria
- Integrating and coordinating the testing activities into the software life cycle activities (acquisition, supply, development, operation and maintenance)
- Making decisions about what to test, what roles will perform the test activities, how the test activities should be done, and how the test results will be evaluated
- Scheduling test analysis and design activities
- Scheduling test implementation, execution and evaluation
- Assigning resources for the different activities defined
- Defining the amount, level of detail, structure and templates for the test documentation
- Selecting metrics for monitoring and controlling test preparation and execution, defect resolution and risk issues
- Setting the level of detail for test procedures in order to provide enough information to support reproducible test preparation and execution

5.2.3 Entry Criteria (K2)

Entry criteria define when to start testing such as at the beginning of a test level or when a set of tests is ready for execution.

Typically entry criteria may cover the following:

- Test environment availability and readiness
- Test tool readiness in the test environment
- Testable code availability
- Test data availability

5.2.4 Exit Criteria (K2)

Exit criteria define when to stop testing such as at the end of a test level or when a set of tests has achieved specific goal.

© International Software Testing Qualifications Board

Typically exit criteria may cover the following:
- Thoroughness measures, such as coverage of code, functionality or risk
- Estimates of defect density or reliability measures
- Cost
- Residual risks, such as defects not fixed or lack of test coverage in certain areas
- Schedules such as those based on time to market

5.2.5 Test Estimation (K2)

Two approaches for the estimation of test effort are:
- The metrics-based approach: estimating the testing effort based on metrics of former or similar projects or based on typical values
- The expert-based approach: estimating the tasks based on estimates made by the owner of the tasks or by experts

Once the test effort is estimated, resources can be identified and a schedule can be drawn up.

The testing effort may depend on a number of factors, including:
- Characteristics of the product: the quality of the specification and other information used for test models (i.e., the test basis), the size of the product, the complexity of the problem domain, the requirements for reliability and security, and the requirements for documentation
- Characteristics of the development process: the stability of the organization, tools used, test process, skills of the people involved, and time pressure
- The outcome of testing: the number of defects and the amount of rework required

5.2.6 Test Strategy, Test Approach (K2)

The test approach is the implementation of the test strategy for a specific project. The test approach is defined and refined in the test plans and test designs. It typically includes the decisions made based on the (test) project's goal and risk assessment. It is the starting point for planning the test process, for selecting the test design techniques and test types to be applied, and for defining the entry and exit criteria.

The selected approach depends on the context and may consider risks, hazards and safety, available resources and skills, the technology, the nature of the system (e.g., custom built vs. COTS), test objectives, and regulations.

Typical approaches include:
- Analytical approaches, such as risk-based testing where testing is directed to areas of greatest risk
- Model-based approaches, such as stochastic testing using statistical information about failure rates (such as reliability growth models) or usage (such as operational profiles)
- Methodical approaches, such as failure-based (including error guessing and fault attacks), experience-based, checklist-based, and quality characteristic-based
- Process- or standard-compliant approaches, such as those specified by industry-specific standards or the various agile methodologies
- Dynamic and heuristic approaches, such as exploratory testing where testing is more reactive to events than pre-planned, and where execution and evaluation are concurrent tasks
- Consultative approaches, such as those in which test coverage is driven primarily by the advice and guidance of technology and/or business domain experts outside the test team
- Regression-averse approaches, such as those that include reuse of existing test material, extensive automation of functional regression tests, and standard test suites

Different approaches may be combined, for example, a risk-based dynamic approach.

5.3 Test Progress Monitoring and Control (K2)	*20 minutes*

Terms

Defect density, failure rate, test control, test monitoring, test summary report

5.3.1 Test Progress Monitoring (K1)

The purpose of test monitoring is to provide feedback and visibility about test activities. Information to be monitored may be collected manually or automatically and may be used to measure exit criteria, such as coverage. Metrics may also be used to assess progress against the planned schedule and budget. Common test metrics include:

- Percentage of work done in test case preparation (or percentage of planned test cases prepared)
- Percentage of work done in test environment preparation
- Test case execution (e.g., number of test cases run/not run, and test cases passed/failed)
- Defect information (e.g., defect density, defects found and fixed, failure rate, and re-test results)
- Test coverage of requirements, risks or code
- Subjective confidence of testers in the product
- Dates of test milestones
- Testing costs, including the cost compared to the benefit of finding the next defect or to run the next test

5.3.2 Test Reporting (K2)

Test reporting is concerned with summarizing information about the testing endeavor, including:

- What happened during a period of testing, such as dates when exit criteria were met
- Analyzed information and metrics to support recommendations and decisions about future actions, such as an assessment of defects remaining, the economic benefit of continued testing, outstanding risks, and the level of confidence in the tested software

The outline of a test summary report is given in 'Standard for Software Test Documentation' (IEEE Std 829-1998).

Metrics should be collected during and at the end of a test level in order to assess:

- The adequacy of the test objectives for that test level
- The adequacy of the test approaches taken
- The effectiveness of the testing with respect to the objectives

5.3.3 Test Control (K2)

Test control describes any guiding or corrective actions taken as a result of information and metrics gathered and reported. Actions may cover any test activity and may affect any other software life cycle activity or task.

Examples of test control actions include:

- Making decisions based on information from test monitoring
- 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
- Setting an entry criterion requiring fixes to have been re-tested (confirmation tested) by a developer before accepting them into a build

5.4 Configuration Management (K2) — *10 minutes*

Terms

Configuration management, version control

Background

The purpose of configuration management is to establish and maintain the integrity of the products (components, data and documentation) of the software or system through the project and product life cycle.

For testing, configuration management may involve ensuring the following:

- All items of testware are identified, version controlled, tracked for changes, related to each other and related to development items (test objects) so that traceability can be maintained throughout the test process
- All identified documents and software items are referenced unambiguously in test documentation

For the tester, configuration management helps to uniquely identify (and to reproduce) the tested item, test documents, the tests and the test harness(es).

During test planning, the configuration management procedures and infrastructure (tools) should be chosen, documented and implemented.

© International Software Testing Qualifications Board

5.5 Risk and Testing (K2) | *30 minutes*

Terms

Product risk, project risk, risk, risk-based testing

Background

Risk can be defined as the chance of an event, hazard, threat or situation occurring and resulting in undesirable consequences or a potential problem. The level of risk will be determined by the likelihood of an adverse event happening and the impact (the harm resulting from that event).

5.5.1 Project Risks (K2)

Project risks are the risks that surround the project's capability to deliver its objectives, such as:

- Organizational factors:
  - Skill, training and staff shortages
  - Personnel issues
  - Political issues, such as:
    - Problems with testers communicating their needs and test results
    - Failure by the team 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)
- Technical issues:
  - Problems in defining the right requirements
  - The extent to which requirements cannot be met given existing constraints
  - Test environment not ready on time
  - Late data conversion, migration planning and development and testing data conversion/migration tools
  - Low quality of the design, code, configuration data, test data and tests
- Supplier issues:
  - Failure of a third party
  - Contractual issues

When analyzing, managing and mitigating these risks, the test manager is following well-established project management principles. The 'Standard for Software Test Documentation' (IEEE Std 829-1998) outline for test plans requires risks and contingencies to be stated.

5.5.2 Product Risks (K2)

Potential failure areas (adverse future events or hazards) in the software or system are known as product risks, as they are a risk to the quality of the product. These include:

- Failure-prone software delivered
- The potential that the software/hardware could cause harm to an individual or company
- Poor software characteristics (e.g., functionality, reliability, usability and performance)
- Poor data integrity and quality (e.g., data migration issues, data conversion problems, data transport problems, violation of data standards)
- Software that does not perform its intended functions

Risks are used to decide where to start testing and where to test more; testing is used to reduce the risk of an adverse effect occurring, or to reduce the impact of an adverse effect.

© International Software Testing Qualifications Board

Product risks are a special type of risk to the success of a project. Testing as a risk-control activity provides feedback about the residual risk by measuring the effectiveness of critical defect removal and of contingency plans.

A risk-based approach to testing provides proactive opportunities to reduce the levels of product risk, starting in the initial stages of a project. It involves the identification of product risks and their use in guiding test planning and control, specification, preparation and execution of tests. In a risk-based approach the risks identified may be used to:

- Determine the test techniques to be employed
- Determine the extent of testing to be carried out
- Prioritize testing in an attempt to find the critical defects as early as possible
- Determine whether any non-testing activities could be employed to reduce risk (e.g., providing training to inexperienced designers)

Risk-based testing draws on the collective knowledge and insight of the project stakeholders to determine the risks and the levels of testing required to address those risks.

To ensure that the chance of a product failure is minimized, risk management activities provide a disciplined approach to:

- Assess (and reassess on a regular basis) what can go wrong (risks)
- Determine what risks are important to deal with
- Implement actions to deal with those risks

In addition, testing may support the identification of new risks, may help to determine what risks should be reduced, and may lower uncertainty about risks.

5.6 Incident Management (K3) — *40 minutes*

Terms
Incident logging, incident management, incident report

Background
Since one of the objectives of testing is to find defects, the discrepancies between actual and expected outcomes need to be logged as incidents. An incident must be investigated and may turn out to be a defect. Appropriate actions to dispose incidents and defects should be defined. Incidents and defects should be tracked from discovery and classification to correction and confirmation of the solution. In order to manage all incidents to completion, an organization should establish an incident management process and rules for classification.

Incidents may be raised during development, review, testing or use of a software product. They may be raised for issues in code or the working system, or in any type of documentation including requirements, development documents, test documents, and user information such as “Help” or installation guides.

Incident reports have the following objectives:
- Provide developers and other parties with feedback about the problem to enable identification, isolation and correction as necessary
- Provide test leaders a means of tracking the quality of the system under test and the progress of the testing
- Provide ideas for test process improvement

Details of the incident report may include:
- Date of issue, issuing organization, and author
- Expected and actual results
- Identification of the test item (configuration item) and environment
- Software or system life cycle process in which the incident was observed
- Description of the incident to enable reproduction and resolution, including logs, database dumps or screenshots
- Scope or degree of impact on stakeholder(s) interests
- Severity of the impact on the system
- Urgency/priority to fix
- Status of the incident (e.g., open, deferred, duplicate, waiting to be fixed, fixed awaiting re-test, closed)
- Conclusions, recommendations and approvals
- Global issues, such as other areas that may be affected by a change resulting from the incident
- Change history, such as the sequence of actions taken by project team members with respect to the incident to isolate, repair, and confirm it as fixed
- References, including the identity of the test case specification that revealed the problem

The structure of an incident report is also covered in the ‘Standard for Software Test Documentation’ (IEEE Std 829-1998).