Certified Tester

Expert Level Syllabus

Test Management

(Managing Testing, Testers, and Test Stakeholders)

Beta 2011

International Software Testing Qualifications Board

Graham Bath, Rex Black, Debra Friedenberg, Kenji Gtonishi, Marcel Kwakernaak, Martin Klonk, Judy McKay, Gary Mogyorodi, Erik van Veenendaal (2010 – 2011)

Revision History

Version	Date	Remarks
2011 Beta	14.4.2011	Beta version

Table of Contents

Acknowledgements

This document was produced by a core team of authors from the International Software Testing Qualifications Board Expert Level Working Group for the module “Test Management”:

Graham Bath (ISTQB Chair, Expert Level)
Rex Black (Chair for the Expert Level Test Management syllabus)
Debra Friedenberg
Kenji Gtonishi
Marcel Kwakernaak
Martin Klonk
Judy McKay (Vice-Chair for the Expert Level Test Management syllabus)
Gary Mogyorodi
Erik van Veenendaal (ISTQB Vice-Chair, Expert Level)

The core team thanks the review team and all National Boards for their suggestions and input.

At the time the Expert Level Syllabus for this module was completed, the Expert Level Working Group (Test Management) had the following membership (alphabetical order):

Graham Bath	Rex Black	Monika Boegge
Debra Friedenberg	Simon Frankish	Leanne Howard
Kari Kakkonen	Marcel Kwakernaak	Martin Klonk
Vipul Kocher	Steen Lerche-Jensen	Judy McKay
Gary Mogyorodi	Joanna Nowakowska	Thomas Rossner
Andrea Rott	Dakar Shalom	Erik van Veenendaal

The following persons participated in the reviewing, commenting and balloting of this syllabus (alphabetical order):

Graham Bath	Gualtiero Bazzana	Rex Black
Monika Boegge	Debra Friedenberg	Bernard Homés
Leanne Howard	Kari Kakkonen	Beata Karpinska
Marcel Kwakernaak	Martin Klonk	Alon Linetzki
Judy McKay	Gary Mogyorodi	Ingvar Nordstrom
Thomas Rossner	Andrea Rott	Erik van Veenendaal

This document was formally released by the General Assembly of ISTQB® on <date>.

1. Introduction to this Syllabus 60 mins.

1.1 The International Software Testing Qualifications Board

The International Software Testing Qualifications Board (hereinafter called ISTQB®) is made up of Member Boards representing countries or regions around the world. More details on the structure and membership of the ISTQB may be found at [ISTQB-Web].

1.2 Purpose of this Document

This syllabus forms the basis for the International Software Testing Qualification at the Expert Level for Test Management. The ISTQB® provides this syllabus as follows:

- To Member Boards, to translate into their local language and to accredit training providers. National boards may adapt the syllabus to their particular language needs and modify the references to adapt to their local publications
- To Exam Boards, to derive examination questions in their local language adapted to the learning objectives for each module
- To training providers, to produce courseware and determine appropriate teaching methods
- To certification candidates, to prepare for the exam (as part of a training course or independently)
- To the international software and system engineering community, to advance the profession of software and system testing, and as a basis for books and articles

The ISTQB® may allow other entities to use this syllabus for other purposes, provided they seek and obtain prior written permission.

1.3 The Certified Tester Expert Level in Software Testing

1.3.1 Expectations

The Expert Level qualification is aimed at people who have already achieved an advanced point in their careers in software testing and wish to develop further their expertise in a specific area. The modules offered at the Expert Level cover a wide range of testing topics.

A testing expert in the subject of Test Management is one who has a broad knowledge of testing in general, and an in depth understanding in a special test area. An in-depth understanding is defined as having sufficient knowledge of testing theory and practice to be able to influence the direction that an organization and/or project takes when creating, implementing and executing testing processes.

The Expert Level Modules Overview [ISTQB-EL-Modules] document describes the business outcomes for this module.

1.3.2 Entry and Renewal Requirements

General entry criteria for Expert Level are described on the ISTQB web site [ISTQB-Web], Expert Level section.

In addition to these general entry criteria, candidates must hold the Advanced Level certificate for Test Manager to participate in the “Test Management” certification at Expert Level.

Holders of an Expert Certificate in a particular module should regularly renew their certification by achieving a minimum number of credits within the ISTQB Continuous Learning Concept [ISTQB-EL-CEP].

1.3.3 Level of Knowledge

Learning objectives for each chapter of this syllabus are captured at the beginning of each chapter for clear identification. Each topic in the syllabus will be examined according to the learning objective assigned to it.

The cognitive levels assigned to learning objectives ("K-levels") are described on the ISTQB web site [ISTQB-Web].

1.3.4 Examination

All Expert Level Certificate examinations shall be based on this syllabus, plus the "Test Manager" module in the Advanced Level syllabus, plus the Foundation Level syllabus. Answers to examination questions may require the use of material based on more than one section of these syllabi.

The format of the examination is described on the ISTQB web site [ISTQB-Web], Expert Level section. Some helpful information for those taking exams is also included on the ISTQB web site.

1.3.5 Accreditation

An ISTQB Member Board may accredit training providers whose course material follows this syllabus.

The ISTQB web site [ISTQB-Web], Expert Level section describes the specific rules which apply to Training Providers for the accreditation of courses.

1.4 Normative versus Informative Parts

Normative parts of the syllabus are examinable. These are:

- Learning objectives
- Keywords
- Required exercises in the workplace

The rest of the syllabus is informative and elaborates on the learning objectives.

1.5 Level of Detail

The level of detail in this syllabus allows internationally consistent teaching and examination. In order to achieve this goal, the syllabus consists of:

- Learning objectives for each knowledge area, describing the cognitive learning outcome and mindset to be achieved (these are normative)
- A list of information to teach, including a description of the key concepts to teach, sources such as accepted literature or standards, and references to additional sources if required (these are informative)

The syllabus content is not a description of the entire knowledge area of Test Management; it reflects the level of detail to be covered in an accredited Expert Level training course.

1.6 How this Syllabus is Organized

There are eight major chapters. The top level heading shows the time for the chapter. For example:

2. Test Missions, Policies, Strategies and Goals	555 mins.

shows that Chapter 2 is intended to have a time of 555 minutes for teaching the material in the chapter. Specific learning objectives are listed at the start of each chapter.

1.7 Terms and Definitions

Many terms used in the software literature are used interchangeably. The definitions in this Expert Level Syllabus are available in the Standard Glossary of Terms Used in Software Testing, published by the ISTQB [ISTQB-Glossary].

Each of the keywords listed at the start of each chapter in this Expert Level syllabus is defined in [ISTQB-Glossary].

2. Test Missions, Policies, Strategies and Goals 555 mins.

Keywords:

analytical test strategy, consultative test strategy, methodical test strategy, model-based test strategy, operation profile, operational profiling, process compliant test strategy, reactive test strategy, regression-averse test strategy, standard compliant test strategy, test mission

Learning Objectives for Test Missions, Policies, Strategies and Goals

2.1 Introduction

No learning objectives for this section

2.2 Mission, Policy, and Metrics of Success

LO 2.2.1 (K4) Discuss the quality goals of the organization and define a test team's mission consistent with those goals

LO 2.2.2 (K6) Define, describe and evaluate test objectives, priorities and test goals that might have long term effects on the software, quality, and dependencies (organizational, economical and technical)

LO 2.2.3 (K3) For a given situation, define effectiveness, efficiency, and satisfaction metrics for the test process, identifying opportunities for improvement

2.3 Test Strategies

LO 2.3.1 (K4) Analyze a given situation and determine which individual strategies, or blended strategies, are most likely to succeed in that situation

LO 2.3.2 (K4) Analyze a situation in which a given test strategy failed, identifying the likely causes of failure

LO 2.3.3 (K6) For a given situation, create a test strategy document that contains the important elements, enables success factors, and mitigates causes of failure

2.4 Alignment of Test Policy and Test Strategy with the Organization

LO 2.4.1 (K6) Define, describe, evaluate and improve the test policy and strategy(ies) both long and short term for a company, organization and/or a test team, including process and organization of test, people, tools, system and techniques

2.1 Introduction

Except for idle amusements, every journey has a purpose, a destination, and objectives. Any complex journey needs a map to arrive at the destination. The purpose of testing is its mission (the “why”), the test policy defines the destination and objectives (the “what”), and the test strategy provides the map to the destination (the “how”).

2.2 Mission, Policy, and Metrics of Success

Testing, as an activity by itself, has no business value. Testing becomes valued when it connects to something valued by one or more key stakeholders in the quality and/or testing of the system under test. The most essential activities of test management relate to the test manager’s ability to establish this connection. Without such connections, no matter how well the test team performs as a matter of technical capability, no matter how excellent the management of the test team by the test manager, the test team and the test manager will fail. Therefore, the test manager must carefully evaluate the connection between stakeholder objectives and testing, and define goals and a mission of testing that relate to those objectives.

While the specific mission and objectives for a test organization can vary, typical objectives include the following:

- Find defects that could affect customer or user satisfaction, and produce enough information about those defects so developers (or other work product authors) can fix them prior to release
- Manage risks by running important tests that relate to key quality risks, in risk order, reducing the risk to the quality of the system to a known and acceptable level prior to release
- Ensure alignment with other project stakeholders regarding common procedures, quality gates, service level agreements (SLAs) and hand off points
- Provide the project team with important information about quality, testing, and readiness to release
- Uncover non-critical defects, and help establish workarounds for them – to assist customer support or the help desk to resolve user problems should the defects not be corrected prior to release

The test organization’s mission and its specific objectives vary depending on the wider context. For example:

- Market-based or regulatory deadlines for product delivery can reduce the defect detection percentage achievable by the test organization.
- Chosen software development lifecycles can place limitations on the extent of risk coverage that the test organization can achieve.
- Business constraints or priorities can limit the amount of testing that the test organization can carry out.

Within these parameters, the test manager must seek to maximize fulfilment of the test organization’s objectives. This requires careful evaluation, and re-evaluation, of these parameters during the establishment of the test organization’s objectives and the target goals for those objectives.

Clarity about the test organization’s mission and objectives is important. A best practice for establishing this clarity is the creation of a test policy document. This document should reflect the appropriate mission and goals of the test organization, including the relationship to the organization’s quality policy. The test manager should ensure that this policy reflects what the test organization can actually achieve, given constraints around budget, schedule, tools, and the like. In addition, for a given organization and situation, the test manager should regularly evaluate the test policy to identify possible improvements.

The test policy should establish the mission and objectives in a measurable fashion. Every objective defined for the test organization should include clear metrics for effectiveness, efficiency, and/or stakeholder satisfaction, along with targets for those metrics (goals). To the extent that goals are not measurable, the test manager will often find the organization in a situation where success is difficult, stakeholder expectations are unachievable, and the potential exists for the test organization to fail.

Examples of metrics of effectiveness, efficiency, and stakeholder satisfaction throughout the fundamental test process include the following:

- Test analysis and design: The test organization develops a set of test cases that cover the most important quality risks (effectiveness).
- Test implementation and execution: The test organization increases the coverage of regression risk (i.e., reduces the risk) by executing automated test cases (efficiency).
- Test results reporting and exit criteria evaluation: The project team finds that the test results, as reported, help them guide the project to success (satisfaction).

The test manager should establish targets for these metrics (goals), in consultation with key stakeholders, and should set goals for improvement over time (see Expert Syllabus: Improving the Test Process).

2.3 Test Strategies

The test strategy is the means by which the test organization implements the test policy, i.e., the test strategy explains *how we test* while the test policy explains *why we test*. Because each project has unique properties, the test approach is the implementation of the test strategy for a particular project.

The expert test manager can do the following:

- Analyze a given situation and determine which strategies are most likely to succeed
- Evaluate and explain the benefits of a test strategy to different stakeholders
- Evaluate a situation where a test strategy failed to identify the likely causes of a failure
- Ensure that the selected strategy is appropriate for the skills, resources, and level of formality of the organization

The following subsections examine the most common test strategies, their benefits, success factors, and risks, and how each strategy relates to key test stakeholders. A test manager may determine that mixing strategies is the best approach to a project in order to maximize the benefits of the various strategies while minimizing the risks.

2.3.1 Analytical Strategies

Analytical test strategies include two of the most common test strategies: requirements-based testing and risk-based testing. The test team follows an analytical test strategy when they analyze the test basis to identify the test conditions to cover.

The following are the benefits, the success factors, and the risks for analytical test strategies:

- Benefits: Alignment of testing with a clearly-defined test basis, measurability of testing against that basis, and defect prevention through early analysis of the test basis. Stakeholders who participate in analysis sessions are given a level of transparency and insight into the testing process and its results.
- Factors for success: In the case of document-focused analytical strategies (e.g., requirements-based, design-based, or contractually-based), the test team must receive the underlying document(s), though these can undergo revisions or iterations. In the case of

stakeholder-focused analytical strategies (e.g., risk-based), the test team must receive input from the stakeholders. In either case, the analysis must occur early in the test process.
- Risks: In the case of any document-focused analytical strategy, unmanaged changes in the documents can undermine the strategy. Absent or low-quality documents may make this strategy infeasible. In the case of risk-based testing, the inability to engage the stakeholders in the risk identification and assessment process renders the list of risks incomplete and the risk ratings incorrect, reducing the effectiveness and efficiency of the strategy.

For more information on requirements-based testing, see [Craig02]. For more information on risk-based testing, see [Black09].

2.3.2 Model-based Strategies

Model-based test strategies include operational profiling, variants of which are commonly used for reliability and performance testing. Model-based test design can also leverage formal models such as UML. These models often are used to cover functional aspects rather than reliability aspects. The test team follows a model-based test strategy when they develop a model of the environment in which the system exists, the inputs and conditions to which the system is subjected, and how the system should behave. This model typically proceeds from an analysis of actual or anticipated situations.

The following are the benefits, the success factors, and the risks for model-based test strategies:
- Benefits: Testing of the system in a manner consistent with the real-world usage of the system.
- Factors for success: An accurate model of real-world usage, and, in most cases, available tools that can automate test execution according to that model.
- Risks: Insufficient data to build an accurate model, statistical inaccuracies in the model (perhaps due to improper skills of the modelers) leading to severe defects escaping to production, improper selection of automated test execution tools to implement the tests, and a tendency to focus on positive paths.

Model-based strategies should involve the key test stakeholders, at least in the validation of the model, but usually also in its actual construction. For more information on model-based testing, see [Utting06].

2.3.3 Methodical Strategies

Methodical test strategies include the use of a standard set of test conditions as a test basis. The test team follows a methodical test strategy when they use a predetermined set of test conditions, such as a quality standard (e.g., ISO 9126), a checklist or a collection of generalized, logical test conditions which may relate to a particular domain, application or type of testing (e.g., security testing). The set of test conditions does not vary from one iteration to the next or from one release to the next.

The following are the benefits, success factors, and risks for methodical test strategies:
- Benefits: Ensures the consistent testing of the defined attributes of the product or system.
- Factors for success: An adequate and current set of test conditions, checklists, or test cases, and a stable test object that does not vary significantly over time.
- Risks: An insufficient or outdated test basis that can lead to gaps in coverage and important defects escaping from testing.

Methodical strategies often involve the key test stakeholders only in the initial definition of the set of test conditions.

2.3.4 Process- or Standard-compliant Strategies

The test team follows a process- or standard-compliant test strategy when they decide to follow a set of processes defined by others, including a standards committee. These processes typically address documentation, the proper identification and use of the test basis and test oracle(s), and the organization of the test team.

The following are the benefits, success factors, and risks for process- or standard-compliant test strategies:

- Benefits: Leveraging the skills and experience of people who developed the standard, e.g., an internal standards committee, the people who built the IEEE Standards or those who conceived the Agile Manifesto [Agile-Manifesto-Web] which defines the agile software development methodology.
- Factors for success: The selected standard or process is aligned with the test problems to be resolved.
- Risks: Improper understanding of the process or standard, improper implementation of the process or standard, and application of the process or standard to a situation where the test problems are different.

Depending on the process or standard, a greater or lesser degree of stakeholder engagement might be involved with these strategies. Agile, for example, requires daily engagement between key stakeholders and the entire project team. For more information on process-compliant strategies, see [Crispin09] and [Drabick03].

2.3.5 Reactive Strategies

Reactive test strategies are dynamic (i.e., the test approach evolves rapidly) and heuristic (i.e., the test cases derive from practical experience rather than formal models such as standard-compliant). When using a reactive test strategy, the test team waits to design and implement tests until the test item is received. Whether in sequential or iterative lifecycles, reactive testing is about reacting to the actual system under test. Some amount of pre-existing structure, such as Whittaker’s software attacks method, might be involved.

The following are the benefits, the success factors, and the risks for reactive test strategies:

- Benefits: Finds defects that other strategies might miss, locates defects efficiently in terms of cost per defect found, allows for continuous re-focusing of the testing on problematic areas, and is robust under conditions of incomplete test basis documents.
- Factors for success: Highly skilled and experienced testers with a deep understanding of the business domain and the technologies used.
- Risks: Insufficient skill among the testers, testers who are not knowledgeable with the system and the subject matter, difficulty in tracking coverage and defining test oracles.

Reactive strategies are frequently used in combination with more formalized strategies to ensure coverage while still leveraging the knowledge of the test team. For more on reactive strategies, see [Whittaker02].

2.3.6 Consultative Strategies

The test team follows a consultative test strategy when they rely on the input of one or more key stakeholders to determine the test conditions to cover.

The following are the benefits, the success factors, and the risks for consultative test strategies:

- Benefits: When the test team properly designs, implements, and executes the necessary tests, the consulted stakeholder(s) receive exactly the coverage they request.

- Factors for success: The consulted stakeholder(s) must have an accurate concept of what should be tested, how much, and in what order.
- Risks: Conflicting priorities between multiple consulted stakeholders and incorrect definition of the test conditions. Stakeholders may have contradictory expectations. To the extent that consulted stakeholder(s) are wrong about the important test conditions, the test effort is also wrong, as this strategy includes no means for self-checking the direction after it is set. If not all the right stakeholders are consulted, the test effort will not enjoy sufficient support.

Because of the risk of the consultative input being inadequate or incorrect, a consultative strategy is usually used in concert with other strategies to reduce the risk of testing gaps.

2.3.7 Regression-Averse and Test Automation Strategies

Regression-averse and test automation test strategies are also quite common. The test team follows a regression-averse test strategy when they use various techniques to manage the risk of regression, especially extensive automation of testing at one or more levels. In iterative lifecycles, regression risk tends to be higher than in sequential models and the need for ongoing regression risk management more acute.

The following are the benefits, the success factors, and the risks for regression averse and test automation test strategies:
- Benefits: Minimization of the risk of regression in one or more key areas while still releasing new versions of the system or product.
- Factors for success: The ability to successfully and efficiently automate maintainable tests at the unit, integration, system, and/or system integration levels.
- Risks: Insufficient or unavailable tools, tests that resist automation, insufficient skills in the test team, poorly tested new features, and the changing scope of the product.

Regression-averse test strategies satisfy a key test stakeholder objective, which is that those features that currently work continue to work in subsequent releases. Test automation strategies are used for a variety of projects but are crucial for the success of agile projects due to the requirement for thorough testing within short iterations. As developers are always a part of the agile team, they can help by designing testable software and often participate in the development of the automation architecture and scripts. For more information on test automation strategies, see [Graham99].

2.4 Alignment of Test Policy and Test Strategy with the Organization

In order to be implemented, both the test policy and test strategy must be practical, complete, consistent, and successful. Practicality means that the policy and strategy actually can be implemented at all test levels where they apply. This often means that different test strategies apply to different test levels (e.g., heavy test automation at the unit test level and risk-based testing at the system test level). Completeness means that the documents address the important elements for such work products. Consistency means both internal consistency between policy and strategy, but also external consistency between the test policy and strategy and the wider project, process, product, and organizational imperatives.

In terms of success of the policy and strategy, the ultimate measures—and perhaps the only ones that count—are those that derive from key stakeholder's objectives for the test team in terms of effectiveness, efficiency, and satisfaction. The expert test manager should be able to evaluate metrics in these areas, and then improve the test strategy as needed. These evaluations and improvements should take into account both short-term and long-term perspectives, process and organization, and, people, tools, systems, and techniques.

Since the key stakeholders will have different objectives for testing, policies and strategies are not uniform across organizations, and sometimes vary across departments or projects within an organization. The policy or strategy that is suitable to one organization may prove to be ineffectual or even counterproductive in other organizations. Therefore, test managers should be able to evaluate policies and strategies for suitability, determine the proper mix of strategies, and identify possible improvements where needed.

The process of blending and adjusting test strategies should continue until alignment is achieved. This means alignment with the test policy, the scope of testing, the goals of the company, and the test team. An expert test manager can evaluate a test strategy to identify alignment discrepancies and suggest resolution. More than that, given a set of facts and constraints, an expert test manager can create a test strategy that addresses the proper elements, is aligned with the test policy and the company, minimizes risks, and maximizes benefits. For more information, see [Riley09], Chapter 2.

The test manager must be adaptive to new projects, programs and organization changes, but must also provide consistency across projects and project teams. A well-defined test strategy will allow adaptation to accommodate these types of changes.

In the future, ISO IEC 29119 (part 3) will provide support for the Organizational Test Policy Process, the Organizational Test Strategy Process and the Project Test Management Process. At the time of writing this syllabus, the standard was available only as a working draft.

3. Managing the Test Team 855 mins.

Keywords:

Myers-Briggs Type Indicator (MBTI) , RACI matrix, S.M.A.R.T. goal methodology, test architect

Learning Objectives for Managing the Test Team

3.1 Introduction

No learning objectives for this section

3.2 Building the Test Team

LO 3.2.1	(K5) For a given job opening, evaluate a set of resumes and determine the best candidate
LO 3.2.2	(K6) For a given job opening, devise an appropriate and effective approach to interviewing candidates, and conduct the interview
LO 3.2.3	(K3) Outline the major points of a training program for new people
LO 3.2.4	(K2) List considerations that must be made when terminating an employee or contractor

3.3 Developing the Test Team

LO 3.3.1	(K2) Discuss the purpose for creating development plans and stating goals and objectives for individual testers
LO 3.3.2	(K4) Review a set of S.M.A.R.T goals for an individual tester to determine if the goals are appropriate based on the tester's resume and the organization's test strategy
LO 3.3.3	(K2) Explain the importance of an individual understanding their role and responsibilities within a test team
LO 3.3.4	(K2) Explain how a particular Myers-Briggs Type Indicator® profile might determine the role assignment for an individual
LO 3.3.5	(K3) Identify areas where a tester should monitor and develop their skills, and select appropriate learning options including training and mentoring
LO 3.3.6	(K5) Conduct a performance review that is targeted to encourage growth in a high performing individual as well as to acknowledge and reward their accomplishments
LO 3.3.7	(K5) Conduct a performance review that is targeted to redirect the efforts of a poor performer to improve performance and to set achievable short term and long term goals

3.4 Leading the Test Team

LO 3.4.1	(K5) For a given situation, evaluate the information and guidance needed by the test team and determine the most effective method of communication
LO 3.4.2	(K3) Brainstorm on various team-building activities that will help to foster loyalty and trust within the team
LO 3.4.3	(K2) Explain the challenges of motivating a test team, both short term and long term
LO 3.4.4	(K4) Create a plan for team building activities that includes off-shore teams and helps improve communication

3.1 Introduction

People management is a critical part of the test manager's role. All test managers should be familiar with the skills necessary to build, develop and lead their organizations. Depending on the size of the test organization, a test manager may be responsible for managing individuals, multiple test teams, managing off-site or off-shore resources, and/or managing managers. This chapter is devoted to the internal management of a test team or test teams.

Larger organizations, and even some small ones, usually will have a department that is responsible for handling the formal employment agreements, administering benefits and providing advice and support for both employees and management. This department is commonly called the Human Resources department or Personnel Office. This section refers to that department with the assumption that the department exists in the organization. For organizations without the actual department, there may be an individual(s) who is assigned similar responsibilities. If the department or individuals exist, they should be consulted as noted in the following sections. If there is no such function in the organization, then the test manager will rely on the rest of the management team, the legal office (if it exists) and precedence when dealing with some of the issues discussed here.

3.2 Building the Test Team

Making good hiring decisions for the test team is a crucial skill of a test manager. Hiring the wrong person can adversely affect productivity and morale, and possibly the bottom line. Although the hiring process generally involves multiple people, it is ultimately the test manager's decision on whether to extend an offer.

It is necessary for the test manager to have a detailed hiring plan, have defined the position to be filled, and understand the skills necessary for that position. The test manager must also be a skilled interviewer in order to make the best hiring decision possible. Many corporate Human Resource departments offer courses in hiring and interviewing, and there are many books available on the subject including books specifically oriented toward test teams, such as [McKay07] and [Rothman04].

3.2.1 Job Descriptions

Depending on the size of the organization, a number of specialized test positions may be defined, based on the role and experience level of the tester and the needs of the test organization.

Each specific position within a test organization should have a detailed job description, which outlines the specific responsibilities and authorities of the position, the amount of experience required for the position, salary range, the required skills and competencies, and so forth. The specifics on what are to be included in a job description may be set by the company's Human Resources department.

Once a job description has been written, it can then be submitted to job bulletin boards, corporate employment sites and whatever other media is used by the organization, so that qualified job seekers may respond.

3.2.2 Resumes

The first step in identifying qualified candidates for an open position is the review of submitted resumes or CVs (curriculum vitae). The Human Resources department may act as a filter, and reject obviously unqualified candidates. If this is not possible, then the test manager or a delegate will need to review the individual resumes and identify those of interest.

Many companies have strict guidelines on how to manage the candidate selection process. The test manager should check with Human Resources to determine what level of documentation is required during both the resume selection and interviewing processes.

When reviewing resumes, the test manager or delegate should look for the following indicators:

- Does the candidate have the required skills and experience?
- Does the candidate have the required or preferred certifications, such as the ISTQB Certified Tester Foundation Level?

- Are the candidate's salary requirements aligned with the open position?
- Is the resume organized and easy to read?
- Are there any spelling or grammatical mistakes on the resume? (Many test managers will not consider candidates with mistakes on their resumes.)

3.2.3 Interviewing

Interviewing people for technical positions generally follows the same structure, regardless of whether the person hired is a developer, a business analyst or a software tester. Good interview practices such as multiple rounds of interviews, a mixed interview team and an understood interview process are all applicable to interviewing testers.

When interviewing potential testers, there are some particular strengths to explore in the candidate. In particular, testers should be able to demonstrate good problem solving and critical thinking skills, good written and verbal communication skills, a strong ability to work within a team, a curious nature, technical knowledge, domain knowledge and the experience/background needed for the position. Interview questions should be oriented to determine the following (at a minimum):

- Does this person have the right attitude for the job and for the organization? Will the person be a good fit for the group? Will the person be effective in the job? Do they take the role of testing seriously? Will they maintain a positive attitude even when schedules get tough?
- Does this person present themselves as a confident and knowledgeable individual without overstating their capabilities? Will the person know when to stand up for their work? Will the person be able to admit when they are wrong and handle the situation in a mature manner? If needed, can this person be sent to a project meeting to represent the interests of the test team?
- Does this person have the organizational skills required for the job? Can the person work in an environment that has frequent interruptions? If interrupted, will the person be able to return to their original tasks with minimal downtime? If they find a defect, will they remember to return to the test they were running and complete that test?
- Does the person exhibit the maturity needed in the role? Do they have the leadership potential the position requires? Can they deal effectively with interpersonal issues that may arise?
- Does the person have the capacity to deal with others empathetically? Can they present a defect to a developer without causing offense? Can they see both sides of an argument? When prioritizing a defect, can they weigh multiple factors that affect the priority rating?

This is not intended to be an exhaustive list, but rather to supply the test manager with a place to start when preparing interview questions. For more examples of interviewing questions and techniques, see [McKay07]. There are many types of interviews and interview questions. As part of the interview process, the interview team and test manager may want to include some of the following:

- Telephone screens – often a good, quick, first interview, and allows the interviewer to check verbal skills and may be the only option with candidates who are not able to be on-site for the interview.
- Group and panel interviews – multiple people interviewing the candidate at one time. This is a more stressful type of interview for the candidate as the "think time" between questions is reduced.
- Behavioral questions – what would the candidate do if confronted by a certain problem?
- Situational questions – have the candidate give a real example of a project situation and what transpired.
- Technical questions – have the candidate display coding or scripting skills, database skills, etc.
- Puzzle solving – present the candidate with a brainteaser puzzle in order to evaluate their problem solving skills and ability to think quickly.

- Specialized questions for specific domains – dependent on the position (software security, highly regulated industries, etc.).
- Questions related to team dynamics – how will the candidate fit in with the rest of the test team? Will the candidate work well with the rest of the test team members?
- Exams – some companies may allow a standardized exam to be given to all candidates for a specific position. These exams check technical knowledge, self-organization capabilities, ability to structure a problem, as well as evaluate written communication skills if essay-style questions are included. In general, if an exam is to be given, it needs to be given consistently, and may need to be approved by the organization’s Human Resources department. Check with the Human Resources department for information on the legal issues or policy regarding the use of exams during the interview process.

In addition to interviewing, assessments are also used to evaluate candidates. Assessments are usually conducted by a group and assess the candidate’s performance of such tasks as demonstrating practical skills (writing a test case for a given test condition), conducting a presentation on a topic, discussing topics with a group, demonstrating analytical skills (drawing a control flow diagram for a given set of requirements), and other pertinent and demonstrable skills.

Prior to formally offering the position to the candidate, the test manager should check past references and follow all processes required by the organization. This may include a background check, credit check, verifying security clearances, residency status, etc.

When interviewing and selecting team members, it is important to remember that the goal is to build a cohesive and effective team. Individual skills are rarely a perfect fit, but the goal is to find people with the potential and capability to develop the skills needed to create that perfect team. Different experience levels and different knowledge levels help to bring varied viewpoints to the team and allow the team to tackle a variety of tasks.

3.2.4 Assimilating New People

The previous sections discussed hiring new people. A test manager doesn’t always get to hire the team members. A test manager may inherit an existing team, may be required to integrate existing teams together or may be given people who have been hired by someone else. Regardless of how the person(s) is acquired, the test manager’s job is to create an effective working whole from the individuals who compose the team.

When a new test team member is added to the group, he or she will need be trained on the test organization’s processes and procedures, as well as become acquainted with the tools being used by the department and project. The test manager should have a solid new hire training program for the organization. This program, which may consist of documentation on the organization’s testing practices and processes, hands-on demonstrations and training by fellow test team members, and training on company policies by the Human Resources department, should help the new hire to quickly assimilate into the team. If groups of employees are hired together, they can attend the training together which helps to build unity among the new hires.

In addition to the training materials themselves, a new hire training schedule should be considered, so that whenever a new team member is hired, the team members responsible for the training know what needs to be done and when. For example, during the first week of employment, the training schedule may include such tasks as learning how to access the project materials. The second week may include such tasks as learning how to use the defect tracking and test management systems. The test manager should establish and verify the achievement of milestones in this training schedule.

The new hire training program for the testing group can be put together by a group of existing team members, allowing them to share their knowledge, and helping to motivate them by allowing them to be involved in the training of new team members. In addition to a new hire training program, the test manager may wish to assign a mentor to the new test team member, so that the new hire has a primary contact for questions while becoming familiar with the new organization.

3.2.5 Termination of Employment

Part of building the team is knowing when and where to trim the team. Unfortunately, there are times when a test manager must make the difficult decision to let a team member go – either due to poor performance, a change in business focus or a reduction in force (layoff).

The test manager should work with the Human Resources department to understand the organization’s policies and procedures, any legal ramifications, any applicable union regulations, and any national labor legislation or agreements regarding employment termination and layoffs. Levels of documentation may be required that must be gathered prior to initiating the termination procedures. In the event that a person is performing unacceptably, the manager should immediately seek the counsel of the Human Resources department to understand what will be required in the performance improvement plan or possible termination plan.

If termination is unavoidable, the test manager should ensure that any required skills or knowledge transfer tasks occur and that all company intellectual property is retained. Testers often have unique knowledge of the system, including undocumented configuration steps, passwords, data locations and other information that will be important for the person who will be assuming the tasks of the person who is leaving. Cross-training, throughout the employment period, will help to minimize the risk of someone walking out the door with critical and unique knowledge.

3.2.6 Ending Contractual Relationships

Many test teams take advantage of increasing their workforce on a temporary basis by using contract help – either individuals or outsource test vendors – to fill skills gaps, and/or to quickly increase team headcount with qualified testing resources.

Most test teams and companies use contractors and outsource test firms for a predetermined period of time, after which, the contract will end and the resources will no longer be available. The test manager should prepare well in advance for the loss of headcount or specialized skills by creating a skills and resource transition plan. This plan should include information on how skills gaps will be filled and how existing resources will be reallocated.

In the case where a contractor or outsource test firm is not performing adequately, early termination may be warranted. The test manager’s ability to terminate the relationship may be predicated by contracts, agreements or other legal documents. As with terminating an employee, contractor or outsource group terminations should be handled by working with the Human Resources department and/or the legal and procurement departments. For more information, see [Black09], Chapter 10.

3.3 Developing the Test Team

Part of a test manager’s role is to provide the necessary tools and opportunities to their team members so that they are able to successfully perform their current duties and improve their skills and capabilities over time. For more information on software team development, see [DeMarco99] and [McConnell99].

3.3.1 Developing Individuals

Test managers are responsible, in part, for the development of their people. This, of course, benefits not only the individual, but also the test team at large as developing existing employees may help to fill skills gaps within the group.

Many organizations require or encourage their managers and employees to create a development plan which outlines areas of improvement and includes a roadmap on how to achieve the changes. The roadmap may include such tasks as attending formal training courses, working toward a degree, self-study on a specific topic, taking on a new role or task, or working toward a testing certification. Successful completion of these developmental tasks may be part of the employee's periodic objectives or goals.

3.3.2 Setting Goals and Objectives

Many organizations require that managers set periodic (usually annual) goals and objectives for their team members, which are used to measure the performance of the employee, and which may be tied to the individual's performance and salary review. It is a test manager's responsibility to work with the individual test team members to mutually agree on a set of individual goals and deliverables, and regularly monitor progress toward the goals and objectives.

If team members are to be evaluated based on meeting their goals and objectives (Management by Objectives) the goals need to be measurable. The test manager should consider using the S.M.A.R.T. goal methodology (Specific, Measurable, Attainable, Realistic, Timely).

When setting goals, the test manager should outline specific deliverables (what is to be done and how it will be achieved), and a timeframe for when the goal or objective is to be completed (when it will be completed). The test manager should not set goals that are unachievable, and should not use goals and objectives that are so simple or vague that they are meaningless. This can be particularly challenging for a test manager when tasks can be difficult to measure. For example, setting a goal to complete a set of test cases may not be realistic if the timeframe is too short or if the incoming quality of the software is not sufficient. It is very important that test goals are fair and within the control of the person being evaluated. Metrics such as number of defects found, percentage of test cases passed and number of defects found in production are not within the control of the individual tester. The test manager must be very careful not to encourage the wrong behavior by measuring the wrong data. If a tester is evaluated based on the number of defects documented, the test manager may see an increase in the number of defects rejected by the developers if the quality of the defect reports is not considered.

In addition to setting individual goals for test team members, group goals may also be established for the test team at large or for specific roles within the test team. For example, a test team goal may be to work towards the introduction of a new tool, complete the training of the test team staff on the tool, and successfully complete a pilot project with that tool. Goals should always roll up to achieve the goals set at higher levels in the organization. An individual's goals should help support the manager's goals and so on.

3.3.3 Defining Clear Roles and Responsibilities

Part of a test manager's job is to make certain that the members of the test team understand their group's role within the organization, as well as their individual group and project-level responsibilities. A test manager cannot properly evaluate an individual's performance unless the person's role and responsibilities have been defined, and performance benchmarks (or goals and objectives) have been set.

Depending on the size of the organization, a number of specialized test positions may be defined, based on the role and experience level of the tester. Some examples of specialized positions that may be included in a test organization are:

- Black box tester (test analyst)
- White box tester (technical test analyst)
- Performance test engineer (technical test analyst)

- Test environment and data administrator
- Tools developer
- Test lead (test manager)
- Test architect
- Test automation specialist (technical test analyst)
- Security test engineer (technical test analyst)

A roles and responsibilities matrix for each of these specialized roles can be created which outlines, in general, the types of activities and tasks each of the positions include. The overall matrix essentially defines the roles and responsibilities of the test organization itself. This type of matrix is valuable internally as well, so that the individual testers know what is expected of them. It will also help identify gaps in skills, which can then be closed through training or recruitment. It is important to ensure the skills matrix matches the expectations on each individual's performance plan. This matrix is also valuable externally, so that the organization has a better understanding of the role of the test team and the types of activities that the test team has the skills and abilities to perform. A more formalized matrix, such as a RACI matrix that defines who is Responsible, Accountable, Consulted or Informed, for defined areas, may be useful.

3.3.4 Individual Personalities and Roles within Teams

It is important for the test team manager to understand the contribution that the individual testers make to their test team and to understand the type of role each individual is likely to play within the team itself. Within the team, the individuals need to understand their roles and capabilities as well as those of their co-workers. Respecting the skills of other team members and understanding the complementary nature of the skill sets enables the team to work effectively as a unit.

Interactions between test manager and tester are often complex and, without the necessary skills, can result in misunderstandings, the withholding of information or even the capture of incorrect or false information. Test managers do not need to be psychologists, but they do need good interpersonal skills which come from an appreciation of co-dependent behavior [Copeland Paper 01].

The test team manager may want to invest in personality profiling of the individual testers, so that both the manager and the test team members have a better understanding of their individual behaviors. A professionally administered personality test, such as Myers-Briggs Type Indicator® (MBTI®), may be used to this end [Myers&Briggs95].

In addition to individual personality profiling, the test team manager may also want to invest in a team role-type evaluation of the test team to better understand the dynamics of the team, and to help the individuals understand more about their own behavior and their team mates' behaviors in team settings. Models can also be used to help individuals understand their preferred learning methods, ability to deal with poorly defined problems and ability to cope with stressful situations.

3.3.5 Skills Development

It is important for a test team manager to encourage test team members to improve their skills over time. This benefits not only the individual team member, but the team as a whole, and aids in filling skills gaps within the team. Areas to be considered for skills development include:

- Tools skills
- Technology skills
- Testing methodology skills (e.g., Scrum)
- Technical testing skills
- Domain knowledge
- Soft skills (e.g., communication)

3.3.6 Training Opportunities

One of the best ways a test manager can develop a team is to advocate the use of training within the test organization. Training can be informal – materials and training done by other test team members, on the job training, or formal – training held by professional training organizations, on-site or by using an e-learning system.

Regardless of the size of the organization and the available budget, internal resources can always be identified and should be used to help increase the skill level of test team members. A test manager should consider the resources at hand, and take advantage of the knowledge within the group as well as leveraging the knowledge of the developers, DBAs, business analysts and other technical professionals. Cross-training test team members both in technical and domain areas so that they can be assigned to different projects within the group is a key training goal. Embedding a test analyst in another group or bringing a developer into the test group for a period of time is also an excellent way to build relationships with other groups as well as to provide learning opportunities. A more formal version of this, job rotation, may be a possible way to implement this cross-training.

3.3.7 Mentoring

In order to develop test team members, the test manager may wish to encourage the use of mentors. If there are specific skills that a person wishes to tune, working with a mentor to receive feedback and ideas may be ideal. Senior people within the test organization can act as mentors for more inexperienced testers, or for testers who are interested in learning a new aspect of the discipline such as performance testing.

In addition to using mentors from within the team, using people from other teams can also be valuable. For example an emerging leader in the test team or the test manager may be mentored by a manager from a different group in order to gain a new perspective and learn new management skills. Similarly, a test engineer may enhance their white-box testing skills by working with a developer.

3.3.8 Performance Reviews and Feedback

Many companies require that each employee receive a yearly performance review, where the agreed-on goals and objectives are reviewed and discussed. An employee's salary, bonus or employment status may be tied to this review. The test manager should contact the corporate Human Resources department for specific requirements for performance reviews.

When preparing for the performance review, the test manager should gather information on the test team member's work over the past review period. This can be done by using work products created by the tester, including test cases, status reports, etc. In addition, the test manager may want to consider asking for the opinion of other employees who have worked closely with the individual under review. This can be done by creating a questionnaire, which asks specific and open-ended question about the employee. Performance reviews incorporate both objective evaluation (work products, test performance) as well as subjective evaluation (the manager's opinion on the performance). It is important that all reviews be supported by facts and examples.

One difficult problem that a manager may encounter occurs when an employee has unrealistic expectations about the job or about their ability to advance through the organization. In this case, the test manager should clearly define the job requirements, the employee's demonstrated abilities and the delta between the two. An employee's aspirations should be dealt with factually and clearly.

3.4 Leading the Test Team

In order to be a strong manager and effective leader, a test manager should strive to use appropriate management and leadership principles and techniques. Many of these are not specific to test management, but rather are general principles and techniques that are widely used for the management of technical teams or teams in general. The focus of this section is on those aspects which affect test managers. Further information on the subject of managing and leading test teams is provided in [Craig02], in particular Chapter 10, "The Test Manager."

3.4.1 Information Sharing and Communication

The test manager has the responsibility of ensuring that the test team understands their group's role within the overall organization, and their individual role within the test team itself.

It is the responsibility of the test manager to share necessary information pertaining to project-related activities with his or her staff, and make sure that the test team members have correct and current information so that they can successfully carry out their duties.

A good test manager must always remember that information sharing and communication must be two-way. A test manager cannot manage in a vacuum, so it is imperative that information from the test team members be collected and considered regularly so that informed decisions can be made. Communication of this type may be formal or informal, and could include:

- One-on-one meetings
- Weekly status reports
- Informal communications

In order to be an effective leader, a test manager must display strong active listening skills, which includes not yielding to distractions such as text messages, instant messages, etc. Test managers with good listening skills are more open, present and responsive. Information should also be available via the organization's test management tools that will provide current project status to the stakeholders.

3.4.2 Fostering Loyalty and Trust

In order to promote loyalty and trust within the test team, the test manager must be open and honest with the team and should encourage a safe environment where test team members are not punished for sharing bad news. In addition, the test team manager should treat test team members with respect, and ensure that team members feel that both they and their opinions are valued.

3.4.3 Team Building

Part of the test manager's role is to create and foster a cohesive test team and help create positive team dynamics. Team building can be done in a variety of formal and informal ways. Open communication, respectful behavior between team members and a shared understanding of the roles and responsibility of the group are all factors in creating a cohesive team.

In addition to the daily activities that help build a strong team, there are various exercises and activities that support team development. Below are some examples of team building activities which a test manager can use:

- Hold a group lunch, where each team member brings in a special dish
- Have a celebratory party after a project is completed
- Celebrate test team member special events, such as birthdays, weddings and births
- Create a test team newsletter or intranet
- Organize an educational session during lunch time, where test team members present something of interest to the team – a new test tool, an interesting testing article, etc.
- Engage an external firm which specializes in team-building activities

When scheduling and holding team-building activities, the test manager should strive to include all team members, including off-site and off-shore team members. If this is not possible, then separate activities can be scheduled.

On a broader, more project-focused basis, the test manager can also coordinate a set of information sharing events where the different disciplines (analysis, development, testing, technical documentation, etc.) give presentations regarding what they do, what they need and what products they produce. This helps to build pride within the various organizations and builds understanding within the project team bringing it together as a whole.

3.4.4 Motivating and Challenging the Test Team

Although many people are self-motivated by a job well done, their salary, an upcoming special event, or the thought of their next assignment, it is important for the test manager to keep motivational factors in mind for the team. In addition to the team building exercises previously discussed, below are several ideas which can help in the motivation of test team members:

- Keep team members informed on important project and corporate information
- Give test team members honest and helpful feedback on their work
- Give people the opportunity to do something different by moving test team members between different projects and tasks
- Give test team members the chance to develop their skills by attending training sessions
- Ask test team members what they are interested in doing and set goals and objectives that work towards these interests
- Assign tasks according to ability and growth plans – something that is challenging for a junior person may be boring for a more experienced person

3.4.5 Managing Distributed Teams

Many test managers are responsible for managing groups of testers that are located in different geographic locations. Team members may be split among different floors or buildings within the same city, or may be spread out across the globe.

Managing a distributed team has unique challenges some of which include:

- Managing across time zones
- Managing with cultural differences
- Managing teams with multiple languages
- Managing teams with different skill levels and expectations

The test manager must be aware of these factors and work with the team members and team leads to ensure that there is clear communication and clear expectations set for each of the teams.

Communication with off-site teams is critical, and the test manager may want to create a communication plan and to schedule regular conference calls or video meetings to ensure that information is shared between the home office and off-site locations.

4. Managing External Relationships 330 mins.

Keywords:

none

Learning Objectives for Managing External Relationships

4.1 Introduction

No learning objectives for the section

4.2 Types of External Relationships

LO 4.2.1 (K3) For a given contract, define the third-party relationship regarding testing responsibilities

4.3 Contractual Issues

LO 4.3.1 (K4) For a given project with an external vendor, define suitable SLAs

4.4 Communication Strategies

LO 4.4.1 (K2) Discuss potential communication strategies for a given third party engagement

4.5 Integrating from External Sources

LO 4.5.1 (K4) Analyze a given project scenario to determine the appropriate level of testing required by the internal test team for a product that has employed a third party for some portion of the development/testing

4.6 Merging Test Strategies

LO 4.6.1 (K2) Summarize the items that must be considered when merging test strategies with a third party organization

4.7 Verifying Quality

LO 4.7.1 (K6) Create a set of entrance and exit criteria for a specified third party project scenario

4.1 Introduction

It is not unusual for a test manager to encounter various project situations where there is a third party vendor involved in creating, testing and/or delivering parts of the product. The test manager must be able to identify where these situations occur, understand the role and understand the level of quality expected from the various deliverables. In addition, considerations must be made regarding the schedule, test approach, documentation and communication required to facilitate the smooth delivery of a quality product. This external relationship may be with a vendor (someone who is paid to provide the designated services) or an external group within the same organization (e.g., tested by a group in a different part of the organization). These will be referred to jointly as “third party”. See [Black09], Chapter 10.

4.2 Types of External Relationships

There are several basic types of external relationships:

- Third party delivers a completed product (turn-key)
- Third party develops the product but the testing is the responsibility of the recipient organization
- Third party develops a portion of the code that must be integrated with the recipient’s code
- Third party is responsible for all of the testing of an internally-developed product
- Third party shares responsibility for the testing of an internally-developed product with the internal test team

When dealing with multiple third parties working on a single project, communication between the third parties may also have to be managed. Each of these relationships may have different requirements for communication, documentation and coordination. The test manager must be aware of these differences as well as any contractual constraints that may be in place.

4.3 Contractual Issues

Contracts are usually used when working with external resources. Although the contract administration is usually handled through the legal department, the test manager must be aware of the contractual issues that may affect the testing project and provide technical/methodological input as the expert of testing services.

Specific contract items that may influence the testing project include:

- Service Level Agreements (SLAs) - turnaround time on defect fixes (may be related to severity/priority scheme), turnaround time on questions/information requests, availability of support staff and method of contact
- Deliverables - documentation (user guides, help files, specifications), build schedules, information regarding known defects, release notes (including notification of areas not ready for test), test data (e.g., large data files from which sensitive information has been removed)
- Quality level - expectations for code reviews, unit test coverage (code coverage), integration testing, system testing results, static analysis testing, acceptable number of residual defects (severity/priority)

Depending on the contract, the industry, the product being developed and the relationship with the third party, additional items may be required. For example, a third party that is contracting to produce a medical device probably will be required to obtain levels of government (e.g., the FDA in the U.S.) approvals as well. Similarly, a third party testing organization that is within the contracting company may require less definition of the SLAs and deliverables if the practices are already established and followed.

If the test manager is involved in the vendor selection process, it is important to ensure coverage in the contract for all the items discussed in this section. If the test manager does not have input into the third party selection, it is wise to track metrics that will help the selection team evaluate potential third parties for future engagements. These evaluation criteria will help improve the contractual standards and may also help determine which third parties should be used again and which should not.

Ideally, the test manager should have significant input into the content of the contract, particularly any items related to schedule, testing aspects and quality measurements. The solution that is to be delivered under the contract must be accurately described to allow for validation of the finished product.

4.4 Communication Strategies

Communication with a third party may vary widely in the amount, detail, frequency, style, format and formality depending on the product, the third party, the locations involved, the criticality and complexity of the project and any previous working relationships. The test manager must be aware of each of these factors before defining the communication strategy for the project. For example, if the project is employing development and unit testing in an off-shore facility, it might make sense to require weekly detailed status reports, daily defect reports, semi-weekly conference calls and perhaps even an agreement on the language to be used in written communication.

Using standard tools for defect tracking and test management can help facilitate effective and consistent communication between multiple parties. This means that appropriate access and responsiveness must be available with these tools and 24/7 support of the tools may be required as well. It may also be necessary to publish usage guidelines (such as severity/priority schemes) to all tool users to avoid misunderstandings. This is particularly important when the third party is doing the development and the local team is doing the testing.

External communication is not limited to other test and development resources. End users, regulation authorities and other stakeholders must also be considered in the communication strategy. Risk assessment workshops are a good way to include these people into the testing process and to create open communication paths for continued and effective exchange of information.

Meeting schedules should be established at the beginning of the project to accommodate all pertinent time zones. It’s always nice to avoid making someone attend a meeting at 3:00 a.m.! Holiday and vacation schedules must also be considered (particularly with different cultures and distributed sites). With agile projects, daily “stand up” meetings, sprint hand off meetings and other methods of frequent and regular communication are used to keep the team informed, even when the team is distributed.

A summary of specific communication styles is provided in the appendix of [Evans04].

4.5 Integrating from External Sources

The test manager must define their team’s level of involvement in the testing of the final product. The expectation for involvement will vary depending on the expected deliverables as follows:

- Third party (or several third parties) delivers a developed and tested product (turnkey) - All testing should be complete with this product. The test team should plan to conduct acceptance testing per pre-defined criteria (verification of contractual requirements, specifications, etc.). Systems integration testing may be needed if this product is being integrated with others prior to release as a product package.
- Third party delivers a developed and partially tested product (unit tested) - At this point, code coverage metrics should be reviewed before the product is accepted into integration/system

testing. The test team should expect to conduct full integration/system testing and any acceptance testing that may be required. This is the same scenario as occurs when the external team develops part of the code that is subsequently integrated with internally developed code.

- Third party delivers a developed but untested product - Ideally, the test team would have the technical capabilities to conduct unit testing, but unit testing is usually better done by the developing team. It may be more expedient to assume no unit testing was done and proceed with integration/system testing. Tracking cost of quality metrics is particularly important in this case to determine the cost to the project of bypassing unit testing.
- Third party conducts all the testing for an internally developed product - This is a test management issue and the test manager must either define or agree to the defined test strategy that will be used to conduct this testing. This includes understanding the test approach, the test documentation, the test tools and any automation that should be developed and delivered along with the tested product. Communication needs in this type of project may differ from one where only the development is outsourced due to the need to closely manage the testing effort as it progresses.

Some projects may require a more extensive level of integration and handoff documentation. It is important for the test manager to understand any requirements that are particular to their domain and to ensure there are handoff checkpoints (milestones) that will validate the completion of all testing requirements.

A contract may be the only enforceable criteria for the project. As noted above, if the contractual specifications are insufficient to ensure an acceptable level of quality, the test manager should track metrics that can be used for future agreements to raise quality levels to meet the internal needs of the organization. Ideally the test manager has a significant influence in determining and enforcing the quality standards for a product. If this authority is not built into the contract, it is imperative that the test manager gather the metrics necessary to prove that future contracts must include this authority. Metrics such as test escapes, defects trends, unit test coverage, etc., will be helpful in setting this requirement.

4.6 Merging Test Strategies

One of the most difficult projects for a test manager to manage is one in which some parts of the testing will be conducted by a third party. This may be across multiple locations, multiple organizations and/or multiple internal/external groups. There are many levels of complication in these scenarios. These include:

- Merging disparate defect tracking tools
- Merging disparate test management tools
- Defining and unifying automation strategy/tools
- Defining test levels and responsibilities for those levels
- Defining shared testing and quality goals
- Understanding the various test approaches that may be employed by the various members of the project
- Defining acceptable entry and exit criteria
- Defining a common glossary of terminology, including technical and testing terms
- Defining common metrics and reporting frequency that will be used to control the project
- Identifying areas of responsibility and touch points between teams (lack of defined overlap may lead to gaps whereas excessive overlap leads to inefficiency)

Since these issues will become more critical as the project progresses, it is important to resolve the approach and tools questions before the project starts. This requires planning and coordination which may not have been included in the schedule or project plan. Regardless, it is work that has to be done and is a time investment that will pay back over time. Another complication that sometimes occurs when trying to reach agreement on these items involves the availability of the testing counterparts in other organizations. Since test resources may be assigned as the project progresses, planning and coordination resources may not be available when the planning should be done. It's important to be sure the planning time is allocated in the project schedules and perhaps even in the contracts to ensure efficient use of the resources when they are available.

When multiple organizations are involved in the testing of a product, the test approaches have to be in alignment. This does not mean everyone has to use the same test approach, test techniques or test automation, but it does mean that the overall result of the testing meets the needs as defined in the test strategy for the project. There is usually one owner of the test strategy and that is usually the test manager for the organization that is procuring the services of the other organization(s). One way to help unify the strategy is to ensure there is risk assessment process that is used to evaluate the various risk factors, and that the prioritized risk factors are addressed in the overall test strategy. Unifying the understanding of the risks and agreeing on the mitigation plan helps to bring about a common implementation of the strategy.

4.7 Verifying Quality

How do we make sure that what we get is what we expected/required? The first step to getting what we want is to define what we want up front, at the beginning of the project, preferably in the project documents or contracts. Objective quality measurements of all the incoming software must be defined and measured before the software is accepted. This can be accomplished by defining measurable entry and exit criteria for each phase of the project. These criteria are sometimes aligned with project milestones although milestones may be more schedule-oriented (schedule milestones) than quality-oriented (quality gates).

The following are examples of entry and exit criteria that might be used to determine the acceptance of the software into various levels of the testing cycles:

- Exit from unit test - statement coverage meets or exceeds 85%
- Entry into integration testing - code static analysis complete, no outstanding errors
- Exit from integration testing - all components of functional areas integrated (interfaces verified to be working correctly)
- Entry into system testing - no outstanding blocking defects
- Entry into system testing - all known defects documented
- Exit from system testing - all performance requirements met
- Entry into system integration testing - no outstanding high priority/severity defects open
- Entry into acceptance testing - all planned testing by the test group(s) has been completed and the testing results meet the specified criteria
- Exit from acceptance testing - sign off by the accepting parties

The list for an actual project would be more extensive and the stringency of the criteria would vary with the product domain, experience with the third party, and the requirements stated in the contract/agreement. It's important to remember that the criteria be measurable in an objective way and that they be agreed to at the beginning of the project. In order to be useful, the criteria must be enforceable. The test manager must be sure of having the authority to accept or reject a project before publishing criteria that might not be enforceable.

Milestones are important for tracking a project and ensuring it is on schedule. As with criteria, milestones must be defined in such a way that completion of the milestone can be demonstrated. This often results in milestones being tied to the criteria defined above. Milestone tracking is often the responsibility of the project manager but should also be tracked by the test manager when the milestones could pertain to the testing schedule or quality-related issues.

5. Managing Across the Organization 780 mins.

Keywords:

None

Learning Objectives for Managing Across the Organization

5.1 Introduction

No learning objectives for this section

5.2 Advocating the Test Team

LO 5.2.1	(K4) Define appropriate steps to take to promote or advocate the test team
LO 5.2.2	(K6) For a given situation, define the quantitative and qualitative benefits of testing, and communicate those benefits effectively to project stakeholders
LO 5.2.3	(K3) Give examples of situations in which the test manager would need to defend the test team

5.3 Placement of the Test Team

LO 5.3.1	(K5) Evaluate an organization's structure, its missions, its products, customers, and users, and its priorities, in order to determine the options for proper placement of the test team within the organization, assess the implications of those options, and create an analysis for upper management of those options

5.4 Stakeholder Communication

LO 5.4.1	(K3) Communicate effectively with testing stakeholders, including non-test staff, about critical issues related to testing

5.5 Creating and Building Relationships

LO 5.5.1	(K3) For a given situation, demonstrate how to create and build relationships with other managers and teams
LO 5.5.2	(K3) Explain and give examples of important relationships a test manager needs to create in a complex project environment

5.6 Advocating Quality Activities Across the Organization

LO 5.6.1	(K2) Identify other groups within the organization who are also involved in quality-related activities

5.7 Integrating Tools Across the Organization

LO 5.7.1	(K2) Define the issues that should be considered when dealing with multi-use tools
LO 5.7.2	(K4) Analyze a proposed change to a multi-use tool and assess the impact on the test organization
LO 5.7.3	(K6) Create a plan for rolling out a new multi-use tool considering all phases of the tool lifecycle

5.8 Handling Ethical Issues

LO 5.8.1	(K5) Evaluate a given set of behaviors under a given set of circumstances to identify possible ethical issues

5.1 Introduction

It's not enough to be an effective manager only within the testing organization. The successful test manager must also be effective with outward management. This includes managing up to one's supervisors, across to the other development and interfacing teams as well as effectively communicating and sharing data. This is true regardless of the general organizational and reporting structure.

5.2 Advocating the Test Team

A critical part of a test manager's job is to advocate, defend and promote the test team and its contribution to the organization.

5.2.1 Promoting and Advocating the Test Organization

At all times, the test manager must promote the test team and the benefits of testing. This includes pointing out successes accomplished by the team, benefits produced by the team, and a realistic appraisal of future improvements that can be made. The test team's accomplishments should be presented honestly and should be accompanied by areas for intended improvement. If mistakes were made, then forthright admission of such mistakes, along with the corrective actions taken or planned, will typically better serve the interests of the test team than obfuscation or blame-shifting.

The test manager should advocate the testing team and should continuously encourage upper management to also advocate the testing team. This includes keeping the team well-informed, adequately staffed, well-trained, properly resourced and respected throughout the organization.

The test manager accomplishes this advocate role by communicating respectfully, effectively and consistently, procuring opportunities for the team, and ensuring that the team receives the respect it deserves. A test team, as with most technical teams, must earn respect. By dealing respectfully with other groups, the test manager helps to pave the way for the team to be treated with respect. The test manager must constantly be vigilant for opportunities for training and advancement for the team. In some cases, this may mean that the test manager detects any areas for improvement or new opportunities and quickly positions the team to take advantage of these. For example, if some members of the test team are falling behind in their skill levels for the technology being tested, the test manager should procure training for those individuals and ensure that the new skills can be exercised in the work environment.

Objective metrics should be used to help reinforce the importance of the testing team to the overall organization, both in terms of cost of quality reduction, but also in terms of efficiency and productivity improvements facilitated by the testing team and the testing activities. In addition, the test manager can use metrics to forecast the results of certain options available to the organization, which is an additional value available from the test team.

5.2.2 Selling the Value of Testing

An expert test manager must be able to effectively communicate and sometimes also sell the value of testing. Testing provides value to the organization in both quantitative and qualitative ways. Quantitatively, cost of quality analysis can be used to determine the benefits of testing either in terms of money or time saved. Qualitatively, the benefits of testing can be assessed in terms of improved customer satisfaction, reduced risk, increased goodwill and similar beneficial returns. The benefits of testing vary based on the industry and the organization. The test manager must be able to accurately present the benefits and costs of testing in terms that the stakeholders can understand and appreciate.

The information about the benefits of testing should be used by the test manager to procure the necessary resources to conduct the necessary testing practices as well as to build the testing organization for future requirements. For more on the value of testing, see [Black03].

5.2.3 Creating a Defensible Team

At some point during a test manager's career, a project will not go as planned (see, for extreme examples, [Yourdon03]). It is possible that the test team will be singled out as one of the causes of the difficulty, and it is important for the test manager to know how to respond and defend the test team in these situations.

The keys to creating a defensible test team are primarily proactive:

- Open communication
- Good documentation
- Strong processes

The company at large and the project stakeholders need to understand the role of the test organization. The test manager should establish expectations for the group's overall responsibilities, as well as their role within a particular project. This can be done by providing a roles and responsibilities matrix for the test team, as well as providing test approach and test plan documentation for a given project.

The test organization should follow the ISTQB Fundamental Test Process (introduced in the Foundation syllabus), which includes appropriate documentation and feedback mechanisms, so that problems and issues can be identified and reported early, and can be resolved successfully. Having strong processes allows the project stakeholders to understand how the test organization works, enables them to gain confidence in the team, and helps understand what to expect during a project life cycle.

The test manager must continually gather information from the team, and communicate status out to project stakeholders. There should be an appropriate level of visibility into the test organization through test documentation, status reports and defect information. There should be no surprises.

If there are project issues, the test manager should deal with them promptly and efficiently. The test manager should not make excuses, but rather demonstrate through objective means what has occurred, and what can be done to mitigate the risks related to project issues.

With agile teams, the testers are a part of the integrated development team (often including analysts and users/customers). Because of this, there is no need to "defend" the test team itself but rather to ensure that the team members understand and embrace the testing activities. This is done using open communication, an appropriate process and ensuring there is an awareness of the testing efforts.

5.2.4 Protecting and Supporting the Team

A testing team is frequently vulnerable to external interference, well-meaning or otherwise. This usually occurs for one of the following reasons:

- The testing tasks are not understood and are under-estimated
- Testing is undervalued
- Testing skills are undervalued, especially when non-test staff believe that anyone can test
- Testers are not effectively utilized within the testing team
- Development organizations are frustrated with the testing effort and want to own it
- Test management is seen as obstructive, uncooperative and inefficient for the overall project
- The testing responsibilities and boundaries are not clearly and measurably defined

It is the job of the test manager to be sure these conditions do not exist or worse, persist. Any of these conditions, or a combination thereof can lead to the following types of resource and respect issues.

5.2.4.1 External Interference

Test teams sometimes receive interference from other project stakeholders and other organizations. For example, if the test team is considered to be too slow in accomplishing the testing, the development organization may attempt to dictate what should and should not be tested and to what depth. It's not uncommon for the test team to receive suggestions for the scope of testing, but it becomes a problem when the test team is no longer determining the testing strategy and approach and is not keeping a quality focus.

External interference often takes the form of schedule control but can sometimes extend to test case formats, automation strategies, test documentation and other areas that should always maintain a quality focus. While input from project stakeholders and interfacing organizations should be welcomed, a strong test manager should not allow other organizations or individuals to interfere with the test team's mission or ability to accomplish that mission.

5.2.4.2 Micromanagement

As with external influence, micromanagement of the testing tasks and testing resources is an indication of lack of trust. Micromanagement sometimes occurs because there is a lack of visibility into the testing tasks and overall mission, strategy and approach. When an outside organization or individual attempts to micromanage a test team or a test team member, the test manager must rectify the problem by determining the cause of the micromanagement. For example, if a project manager is requiring the test team members (and only the test team members) to account for their time in hourly increments, there is a mistrust situation and it is likely the project manager feels there is inadequate access to testing information.

In order to avoid micromanagement scenarios, the test manager must be part of overall schedule planning, must clearly communicate test plans and progress to the project team and must respond to requests for additional information. Clear communication during the planning activities about the test approach, test plan, and test estimates, including reviews and sign off by the stakeholders, can help. The more visible the test team's activities and plans, the less likely the team will be subjected to micromanagement.

5.2.4.3 Disrespect

The test manager should never tolerate disrespectful behavior, either direct or indirect, toward the testing team. This lack of respect is usually rooted in a lack of understanding of the value of the contribution of the testing team. This may take the form of snide remarks regarding the skill of the testers, outright insults, allocation of tasks that are inappropriate for the test team, inadequate allocation of time and resources, insufficient access to training or knowledge sharing, and other similar actions.

5.2.4.4 Re-organization and Resource Re-allocation

Test teams can be disproportionately affected by re-organizations or re-allocations that result in insufficient testing resource availability for the committed work. As with the above problems, this usually occurs when the rest of the organization doesn't understand the testing tasks and under-estimates the work to be done.

In some extreme cases during workforce reductions, the test team may be more severely affected than the development organizations. This can occur when people don't see (and the test manager does not measure) the value of testing whereas development produces more easily quantified results. The test manager must ensure that the value and contribution of the test organization is well-understood and valued by the rest of the organization.

5.3 Placement of the Test Team

All organizations are different and organizational structures vary widely. In addition, many organizations adopt different structures on a project by project basis. There is no one perfect structure that will work for all organizations and all projects. There are, however, common characteristics of successful testing organizations as discussed below.

While working within the required schedules and budgets, the test team’s primary focus should be on quality. Quality does not mean releasing only perfect products, but it does mean being able to accurately and honestly assess and report the quality of the software under test without fear of retribution, particularly within the management chain. Schedules and budgets are created with the understanding that adequate time and money must be devoted to the quality effort in order to produce a quality product.

Ideally, the quality-focused test team reports into a quality-focused management chain all the way to the top of the organization. It is the job of the test manager to ensure the most optimal positioning of the team to facilitate this focus while accomplishing the organization’s mission and objectives.

In addition to quality focus, there are other considerations in the placement of the test team. Access to information is critical to the success of the test team. Information usually comes from the development team, analyst team, business stakeholders, project managers and many other project stakeholders. The test team must have adequate access to the information that is available from these sources, both formally (in the form of specifications and requirements) and informally (email, conversation, white board discussions, etc.).

The overall skill level and type of knowledge the test team possesses is also important. In some organizations, domain knowledge may be a critical requirement for each tester. In other organizations, knowledge of testing may be the most valued skill. Technical skills such as system configuration, networking knowledge, etc. may be highly valued. In an organization, the mix of skills needed from the testing team will contribute to the respect they receive and their access to information. At times the skill mix that is required is not organization-specific but is actually project-specific. For example, in an agile team, the test team may be a part of the development team and may be required to have programming expertise. Because agile teams tend to be more self-organizing, individuals tend to evolve into the roles for which they are best suited. As a test manager, it is important to be aware of the changing requirements and expectations of the external and internal stakeholders in order to be sure the test team is staffed and trained as needed.

5.4 Stakeholder Communication

A successful test team has access to the information it needs and supplies information to the various stakeholders in an accurate, timely and understandable way, and in a way that matches stakeholders’ information requirements. This communication includes information regarding the testing project in general, specific test status, specific defect status, trending information, test process effectiveness and efficiency indicators, and other metrics that are regularly and periodically communicated. The successful test manager understands the two-way nature of the communication with the other stakeholders and is effective at both supplying and gathering information.

When considering the “how” and “when” to communicate, the test manager should assess the urgency of the information and the clarity of the information. For example, an urgent schedule issue should be communicated actively to the concerned stakeholders whereas it might be acceptable just to post the weekly status report to an intranet site and publish the link to those who might be interested. If the information will require some level of interpretation in order for the message to be clearly understood (for example, quality problems during system test are indicating a lack of unit testing), then that information would be better presented in a meeting where the test manager can also answer any questions that may occur.

The test manager should focus on gathering and interpreting information that is actually required and will add value for one or more stakeholders. Too much communication or communication of information that is not pertinent to the stakeholder frustrates the recipient of the information and wastes the time of the supplier. The motivation of the communication must be understood. If the communication is intended to gain management support or commitment, the communication must be oriented in such a way that the receiving party understands what is being requested, understands the expected costs and benefits and understands the required actions.

Effective communication is defined as having occurred when the message that was intended to be sent by the sender is the same message that is received and understood by the recipient. If the sender intended that the receiver take action as a result of the communication, then the communication is effective if that action is taken.

5.5 Creating and Building Relationships

The ability to create, build and maintain relationships with other managers and other teams is a determining factor in the success of a test manager. The successful test manager must be able to cultivate these relationships, determine mutual benefits of the association and maintain that association. For example, a test manager should be able to speak honestly with a development manager regarding the status of the development phases of the project, the quality expectations, expected metrics and identified deliverables. Within contractual limitations, a test manager must also be able to communicate effectively with vendors and other third party groups who may have input into a project or who may be consumers of the output from the project. While every situation is different, the test manager should consider informal as well as formal communication opportunities to help build relationships. Taking a development manager to lunch, sharing coffee break times, including stakeholders in discussions of new techniques, and acknowledging and discussing successes on other projects are all ways to help build relationships.

Creating relationships is not sufficient, however. The test manager must also work to maintain those relationships even though the organization environment may change, personnel may change and even responsibilities and expectations may change. A successful test manager is able to adapt to these organizational fluctuations while still leading a respected and productive test team.

Necessary relationships are not just limited to others in the same peer group, but should also include individual contributors, upper management, project management and any other stakeholders who may have valuable contributions to make to the project and the testing organization. A test manager should remember that the hierarchical relationships shown in an organizational chart are only one indicator of the power and influence wielded by certain individuals; true power and influence derives also from respect, especially in organizational and ethnographical cultures that are egalitarian. A test manager should be a visible contributor and a strong and respected representative of the test team.

While building the external relationships, the test manager must apply the same diligence and concern to the internal relationships with the leaders and individual contributors within the testing team. The same level of respect, communication and professionalism that is afforded to the external relationships must also be applied to the internal relationships.

While working only within a project team, the test manager is likely to know all the participants and is able to build individual relationships with those people. As the scope increases, the relationships become more remote and the test manager may not have met members of the various project teams. In this case, the test manager must select the representative individuals with whom to cultivate relationships. At a minimum, the test manager must have a positive relationship with peers within the various project teams. The test manager must also form relationships with the people who actually get the work done (this may not be the managers) and must also have relationships with the people who have access to the information and resources needed by the test team.

As with many management tasks, relationship building can sometimes be successfully delegated. The manager doesn't have to have personal relationships with each member of the project teams, but the test manager does need to know someone who can access each individual if needed. While this type of "remote" management may be difficult to master, it is imperative in a large or distributed environment.

Political and stakeholder management skills become more important as the test manager climbs the management chain and increases scope of responsibility. These soft skills are essential to building and maintaining the relationships that make the work possible for the test group. Key relationships include those with project managers, development managers, key stakeholder representatives, marketing, sales, technical support, technical documentation, vendors and any others who contribute to or influence the testing work or the assessment of quality.

5.6 Advocating Quality Activities Across the Organization

Testing and quality activities should not be limited to the testing group. Every group that contributes to the definition, creation and support of a product should be aware that they are also contributing to the quality of that product and have testing responsibilities. For example:

- Project stakeholders should participate in work product reviews of such items as the requirements, design documents, architecture documents, database design documents, test documentation and user documentation.
- Business analysts are frequently expected to do some validation testing to ensure that the product being built is the product that will meet the needs of the customer.
- Developers unit test, perform some level of integration testing, and may also create automated tests used by the test team for system and system integration testing.
- Customer representatives conduct user acceptance tests.
- Operations people conduct operational acceptance tests.

The results of this testing should be gathered by the test manager and used as input for the testing done by the test team as well as an assessment of the quality of testing that has been performed by the test team. For example, problems found in user acceptance tests may be an indication that requirements reviews were insufficient, system testing was not completed or sample test data was not reflective of production data.

Another challenge related to other testing work is that other groups often do not track and manage defects according to the same rigorous processes typically used by testing groups. The test manager should work with these other groups to ensure that sufficient information is gathered about such defects to ensure proper resolution. It can help when defects and other test-related information are managed using the same test management tools as those used by the test group.

Unless the test manager is aware of the other testing work that occurs and has a way of verifying if it has been done and to what extent, the test team risks inefficiency due to re-testing areas that have already been well-tested and not adequately leveraging the available skills of the other project stakeholders.

Of course, just because other groups are expected to perform their testing tasks doesn't mean that they actually do. The test team also risks failing to test important conditions if it assumes, without proper verification, that other groups covered these conditions. These other groups, just as the test group, deal with tight schedules, scope creep and resource issues. A successful test manager understands the relationship between quality, schedule, budget and features for a given project within a given organization. Trade-offs should be expected and the test team should be ready to deal with changing priorities between the different aspects of the project. In an agile team, the test manager may assume the role of the quality coach by advocating quality activities throughout the project.

5.7 Integrating Tools Across the Organization

Tools are frequently shared across multiple functions in an organization but the usage of the tools may vary. For example, a defect tracking tool may be used by the test organization for tracking defects, by the development organization for tracking their implementation tasks and by the business analysts for tracking requirements. The data stored in this tool, while useful to each individual group, cannot be used in an aggregated form without careful planning for identifiers in the data which will allow each data grouping (e.g., defects, tasks) to be processed appropriately. For example, a trend chart would make sense for defects, but not for developer tasks.

Ideally, tools that are used across groups for various functions, multi-use tools, should be acquired, implemented and used according to a pre-defined plan that will maximize the potential of the tool while ensuring usability for all the parties. More often, the usage of a tool grows over time and is largely undocumented. This results in issues for maintainability, updates, data conversions and tool migrations.

The test manager is sometimes the owner of the multi-use tool and sometimes a stakeholder with vital interest in the accessibility and accuracy of the data maintained by the tool. As such, the test manager is usually involved, sometimes absorbed, in issues regarding purchasing, updating, converting and retiring these types of tools. This is considerably more complicated than dealing with a tool over which the test manager controls all usage.

Multi-use tools have different issues than single use tools. For each phase in the lifecycle of the tool there are specific considerations:

- Purchasing/Selecting/Acquiring
  - A selection committee is required, including all stakeholders in the purchase, use and maintenance of the tool.
  - Integration, while important at the single tool level, is vital for the multi-use tool. All integration points must be defined, data types identified, data transfer mechanisms understood, and requirements for customized code identified (as well as who will write it and maintain it).
  - Capability of the tool to support various defined user and administrator roles must be clearly understood as well as the management of those users.
  - Ownership of the tool must be clearly stated, including who pays for it, who will own the implementation effort, who will maintain it and who has the authority to make implementation decisions.
- Updating/Maintaining/Supporting
  - Initial integration of tools into a single multi-use tool may require programming, testing, and support resources. Selection of “best of breed” tools and subsequent integration of the tools can require considerable planning and technical expertise.
  - A defined process must be agreed to by all parties regarding who will be able to change what and the approvals that will be required for the changes.
  - Responsibility for support of the user base (help desk) must be assigned and funded.
  - Rules regarding tool customization by the individual usage groups must be defined and clearly communicated and monitored.
  - Data migration, if needed, must be documented and facilitated with tool support to allow data to flow into the new tool without compromise to the incoming data quality or the existing data integrity.

  - Impact analysis must be conducted for any modification considered to the tool set to determine if there is any impact to other consumers of the tool.
  - Backup and restore requirements and responsibilities as well as SLAs for support-related functions must be defined.
- Converting
  - A tool is never permanent and changing business needs, vendor issues, forced upgrades and other external factors may force conversion from the current tool set.
  - Conversion both onto and off of the tool must be carefully planned to ensure data integrity, continuation of service, consistent data access and acceptable levels of usability.
- Retiring
  - A tool usually can't be retired unless a replacement is available.
  - As with conversion, retirement requires converting to another tool or method that will allow the continuation of the business function.
  - Archived data must be preserved, secured, and, in some cases, remain accessible.

Tool integration is rarely easy and requires considerable time, effort, diplomacy and forethought. Compromise, negotiation and creativity are usually required to reach an acceptable solution for all stakeholders. Rather than having a tool owned within one stakeholder group, a separate tools group is often formed to provide independent and objective usage guidelines for all stakeholders while supporting the business needs of the overall organization.

5.8 Handling Ethical Issues

Ethical issues can be encountered in many phases of a project and in different interactions. The test manager's role in dealing with these situations is discussed in the following sections.

5.8.1 Managing the Team's Ethics

The test manager's conduct must always comply with the ISTQB code of ethics as those ethics apply to interactions within and external to the organization. The test manager's ethics are an example for the individual team member ethics. The test manager should hold the team to these ethics and should periodically reinforce the code of ethics through training, reviews and situational analysis. Adherence to the code of ethics should be rewarded during performance reviews and breaches in ethics should not be tolerated.

While it may seem that ethics are clearly understood and obvious, the test manager should never assume this is true. It's important that the ethics be clearly stated and that the test team understands that the ethics must be adhered to regardless of the behavior of other individuals in the organization. Because a test team is frequently faced with situations ripe with conflict, professional conduct is critical. A brief lapse can result in serious long term consequences for the individual and the test group as a whole.

5.8.2 Interacting with Test Stakeholders

Professional behavior, factual and objective reporting and adherence to the ISTQB code of ethics are required for all interactions the test manager will have with the various test stakeholders. By being the presenter of objective and accurate information, the test manager will help to build respect and value for the test organization. The test manager must be trusted by the stakeholders.

It can be difficult to remove the emotion from the presentation of the test status, for example, when the test team is pressed for time and other organizations are trying to tailor the testing to fit the schedule rather than to address the quality goals. The test manager must understand the perspectives and motivations of the various stakeholders, but must also present the objective test status information clearly and honestly, regardless of how unpopular the information may be.

5.8.3 Report Results

A large amount of the test manager’s communication with the external test stakeholders is in the form of written reports, charts, metrics and presentations. All reports must be reviewed to ensure the data is accurate, presented correctly and clearly and does not target any individual. For example, a report of defects found should never be reported by developer but rather should be reported by area of the software, feature, quality characteristic or using some other impersonal partition.

In addition to reporting the results of testing, internal communication such as defect reports, emails, status reports and other written and verbal communication should always be objective, fair and contain the proper background information so that the context is understood. The test manager must ensure that every member of the test team conforms to these requirements in all communications. Test teams have lost credibility when just one unprofessionally written defect report found its way into the hands of upper management.

5.8.4 Test Management Ethics

In addition to managing the ethics of the team and their own personal ethics, test managers must also consider the ethics required of managers in general. Managers are bound by an ethical code that prohibits unprofessional behavior regarding their employees. This includes discussion of performance issues, resolving interpersonal issues between team members, handling potential legal issues such as sexual harassment and other personnel-related situations. The test manager, in particular, is working in a high pressure environment, often juggling schedule, budget, quality and technical issues. The test manager must also deal with the individual personnel issues that arise in this pressure environment and must handle those issues promptly and professionally.

As a manager, the test manager must have an open and honest relationship with other managers. It is not unusual for a test manager to be the first to hear about performance issues with employees in other areas. For example, the test team often is the first to identify a developer who is consistently producing bad code. This puts the test manager in the position of providing the development manager with the information they need to take appropriate action. As with all communication, conveying this sensitive and unpopular news must be done professionally and factually. Cultivating these relationships with other managers becomes particularly important when problems such as personnel issues become visible across interfacing groups. Relationships are difficult to build and easily damaged. Each situation requires an assessment to determine the most appropriate behavior.

6. Project Management Essentials 615 mins.

Keywords:

confidence intervals, planning poker

Learning Objectives for Project Management Essentials

6.1 Introduction

No learning objectives for this section

6.2 Project Management Tasks

LO 6.2.1	(K6) For a given project, estimate the test effort using at least two of the prescribed estimation methods
LO 6.2.2	(K6) Use historical data from similar projects to create a model for estimating the number of defects that will be discovered, resolved, and delivered on the current project
LO 6.2.3	(K5) During the project, evaluate current conditions as part of test control to manage, track, and adjust the test effort over time, including identifying any deviations from the plan and proposing effective measures to resolve those deviations
LO 6.2.4	(K5) Evaluate the impact of project-wide changes (e.g., in scope, budget, goals, or schedule), identify the effect of those changes on the test estimate
LO 6.2.5	(K6) Using historical information from past projects and priorities communicated by project stakeholders, determine the appropriate trade-offs between quality, schedule, budget, and features available on a project
LO 6.2.6	(K2) Define the role of the test manager in the change management process

6.3 Project Risk Management

LO 6.3.1	(K4) Conduct a risk assessment workshop to identify project risks that could affect the testing effort and implement appropriate controls and reporting mechanisms for these test-related project risks

6.4 Quality Management and Testing

LO 6.4.1	(K4) Define how testing fits into an organization’s overall quality management program

6.1 Introduction

The test manager's scope of involvement in a project often spans many areas. While there may not be direct responsibility in all these areas, such as project management, the successful test manager will certainly pay attention to and be involved in project management, risk management, reviews and assessments and in determining the proper levels of documentation that must be produced by the test team.

For an overview of project management essentials and concepts relevant for software managers, see [Drucker06], [Rothman07], [McConnell97], [Brooks95], [Glass02], and [PMI08].

6.2 Project Management Tasks

Although a test manager primarily manages the testing project, the test manager must also be aware of and participate in the overall project management tasks as well. Each of these areas is discussed in more depth below. It's important for the test manager to remember that the testing project does not occur in isolation from the rest of the project.

6.2.1 Test Estimation

Test estimation includes the time and effort required to complete all the activities in the fundamental test process. Estimations for the Execution and Reporting activities are heavily influenced by the complexity and quality of the software under test, but, in truth, all testing activities are influenced to some extent by the complexity of the software and the incoming quality of both the software and the documentation. The following is a list of techniques that can be used to determine the time and effort requirements for test implementation and execution:

- Brainstorming with the project team regarding testing estimates. Work to find commonality in the estimates and discuss any outlying estimates to ensure that common assumptions are being used. In agile projects, this is done by conducting planning poker sessions and compiling story point estimates to determine level of effort for user stories.
- Estimating the time required to execute each test case once and then multiplying that number by an estimated number of iterations for each test case.
- Using the quality risk analysis to determine an estimate of the number of test cases required for a project. For example, the number of tests needed for a high technical/business risk item would likely be several times higher than the number needed for a low risk item. From there, estimate the average time required to create/maintain each test case and the time required to execute each test case.
- Using function point analysis (FPA) or a calculation that converts number of lines of code to function points (e.g., 50 lines of code equals one function point) to determine the size of the project. Based on function points, estimate the number of test cases needed and then determine the time required to create and execute those test cases.
- Using developer hours to determine testing hours required. This is usually done using historical data for similar projects in similar organizations.
- Conducting a test point analysis (TPA) to determine the total test hours required for the project.
- Using a historical heuristic from the same or a similar organization and project.
- Applying project management techniques such as work breakdown structure (WBS) and product breakdown structure (PBS) to scope the project and determine high level schedules.

All test estimates should include the following:

- Time to create/assemble the testing documentation (test plan, test cases, etc.)
- Time to acquire test data

- Time to obtain, configure and verify test systems
- Time to execute the tests
- Time to gather and report the test execution information

Other project time considerations may include time for meetings, training, administrative overhead, planned vacations/holidays and other time constraints that may affect the schedule or availability of key resources. It may also be important to clearly state the number of test hours that are available to the team. When project reports are created, these estimates will be compared to the actual time spent.

Additional factors to be considered in the estimation process include an estimation of the incoming quality of the software (based on who else is testing), change control and configuration management processing, maturity of the testing process itself, maturity of the project team, type of software development lifecycle, reliability of the defect fixes, support resources for test environments, the cost of business resources, and the availability of accurate documentation and trained personnel. In addition, the reliability of the test systems, test data, and the availability of a test oracle all contribute to the ability of the test team to perform efficiently.

In addition to the time required to conduct the testing, the test manager must also estimate the number of defects that will be found. This is used to help determine the time that will be required for defect investigation, re-testing (both confirmation testing and regression testing) and defect documentation activities. The test manager may decide to assign different time requirements based on the severity or complexity of the problems.

Estimating the number of defects is also helpful in monitoring the progress of the testing project and assessing if the quality is better, worse or as expected. Defect estimations can be made using techniques similar to those listed above for test estimations. The size of the software being created (lines of code, FPA, TPA, developer hours) as well as complexity measures can be used to determine the number of expected defects. For example, the test manager might determine that it's reasonable to expect that a developer will produce one defect for every two hours of coding. Similarly, the test manager's research may indicate that five defects should be expected for each function point.

These numbers can be derived from the history of the project team working on similar projects, from industry averages or from experience with similar developers and similar organizations. Once the defect numbers are estimated, the test manager can then assign a unit of time that an average defect requires from the testing team for investigation, documentation and retest. This allows the test estimate to flexibly adjust to the actual number of defects found versus the number of defects estimated. If the defect estimate is buried in the testing estimate, it's difficult to adjust the estimate when the quality of the software is better or worse than expected. For industry averages, see [Jones03].

In agile projects, brainstorming methods such as planning poker are frequently used to estimate effort on an iteration/sprint. Using techniques such as this allows the team to deal with uncertainty and risk quickly because the planning is for a short and relatively well-defined set of functionality. Potential risks will either be realized or discarded within the short time interval, allowing the team to adjust the next iteration accordingly.

Regardless of how the estimates are derived, the test manager must carefully track the metrics to verify the accuracy of the estimates. If there is a wide variance, adjustments to resources, planned features, and/or schedules may be required.

6.2.2 Defining the Testing Schedule

Once the estimates are in place, the test manager can define the testing schedule. This schedule must include measurable milestones, touch points, dependencies, and delivery of work products between groups and must work within the overall project plan. Because testing schedules are highly dependent on smooth hand offs with all expectations met, a list of objectively verifiable criteria should be defined for each dependency and work product delivery. This will help the test team to enforce the requirements for the dependencies and deliveries as well as to be able to assess the impact of deliveries that do not meet the designated criteria.

For example, if the testing schedule is based on an assumption that the unit testing done by the developers will have achieved 100% statement coverage, that assumption should be clearly stated and a method of measurement defined. If the code arrives and the testing has only achieved 20% coverage, the testing schedule will need to be adjusted to increase the testing time due to the increased number of undetected defects that will be delivered with the code.

Project management tools are usually used to define and track the testing schedules. These tools provide charting capabilities as well as identification of critical path items. Each defined dependency and delivery point should be clearly identified and all subsequent tasks should be shown to be dependent on the previous dependency and/or delivery.

Deliverables between the project team members aren’t the only items that need to be defined on the schedule. There may be times when input or participation by other stakeholders is required before testing can progress. For example, a usability analysis may be required after initial functional testing and before non-functional testing is started to address usability issues discovered during functional testing. This is a dependency with another group and, although it doesn’t have a formal delivery, there are expectations of the deliverables to the usability team (feature complete, functional software) and deliverables from the usability team (defect reports, enhancement suggestions). These should all be clearly defined in the project plan and noted in the schedule.

Depending on the environment, the testing schedule may be constantly changing. For example, in an agile environment, the schedule is based on what is planned for an iteration but during the work in the iteration, the schedule may need to adjust as items are added or deleted. Schedules that are established at the beginning of a project rarely remain unchanged throughout the life of the project. A test manager must keep a realistic view of the elasticity of the schedule and must monitor to detect when changes/adjustments may be warranted.

6.2.3 Budgeting and Resource Allocation

The requirements for budgets and resources vary widely between projects and between organizations. In some cases, the test team may operate out of an equipped lab with set resources. In this case, the budget and resource allocation may not be a significant part of the test manager’s job, however budget and resource restrictions may significantly affect test schedules. Even with set resources, budgeting may be done on a project basis, meaning the test manager may have to allocate resources across multiple projects while adhering to the budgeting restrictions of the various projects.

When dealing with budgets the test manager should consider the needs for the following:

- Salaries, benefits, periodic salary increases
- Cost of external or contingent staff
- Vacation allocations
- Equipment to be tested (equipment itself being tested)
- Equipment used in testing (servers, networks, printers, etc.)
- Software used in testing (operating systems, interfacing software, databases, etc.)
- Software tools (database tools, reporting tools, analysis tools, defect tracking tools, test execution automation tools, etc.)
- Facilities
- Books, training, travel
- Investments in long term efficiency improvements (test automation, tool procurement, etc.)

There may be other considerations depending on how the organization is structured and how resources may be used across the organization. The composition and capability of the team are important factors, i.e., the specific mix of permanent employees, contractors, off-shore, and outsourced team members and also the skill levels of each member of the team. The test manager should be aware of which resources can be leveraged from other organizations and which resources should not be leveraged. For example, the test team sharing the same systems with the developers can often result in an unstable testing environment.

It is important for the test manager to remember that budgeting is an on-going process. For example, adding resources to the project, switching from on-shore to off-shore resources, finding a higher number of defects than was anticipated or re-assessing the risk on a project can all affect the budget requirements. The test manager must closely monitor the project to track the budget expenditures so that any noted discrepancies can be reported and resolved quickly.

6.2.4 Managing and Tracking a Project

The key to effectively managing and tracking a project is to make an estimate and monitor adherence to that estimate. Since budget and schedules are usually based on these estimates, tracking to the estimate is a sensible way to quickly spot variances and to improve the estimation process. For example, if a test project is taking longer than was estimated, the test manager should determine the cause. Are test cases being executed more times than expected? Are there more defects than estimated? Is testing being blocked or rendered inefficient by test system problems? When the test manager has a schedule and estimate to track to, it’s easier to spot variances and to address quickly those variances.

In some cases, the test team cannot fix the cause of the variance. For example, if the software is of poor quality and more defects were received than were expected, the test manager should review the various hand offs to see where the problem should have been detected. For example, was the unit testing insufficient? Was the planned integration testing not performed? This information helps not only to suggest process improvement areas, but also will help the project team to understand the issues and to work together to fix the problems.

In an agile team, variances are usually due to circumstances within the team and are usually solved within the team. In Scrum projects, this identification and eventual resolution of unforeseen variances caused by problems that are not immediately resolvable (called impediments) is the one of the main tasks of the Scrum Master. Scrum is an agile development methodology characterized by specific team roles and terminology (e.g., Scrum Master replaces the role of project manager). Team members may be defined as either core team members or ancillary team members. Further information can be obtained from [P&C-Web].

Managing and tracking a project requires that there is a plan in place with measurable metrics that can be assessed at various points in the project. A project with no plan can quickly derail but the derailment can be undetected because no one knew where the project should be at that point in the schedule. The test manager should work with the project manager to define the measurable quality goals and to track the progress toward those goals. There are often trade-off decisions that must be made as part of managing a project. For example, is it better to accept the late delivery of a feature that is highly desired by the business, but risk the quality of the project? Being able to quantify and explain the risk is the job of the test manager so the project team can make the best decisions.

A well-managed project has clear task ownership. The test team should have its clearly assigned tasks, and individuals within the team should each know what they are expected to do and when it should be done. For example, if a test team member has been assigned a specific quality characteristic to test such as performance, that team member should know when performance testing should start, when it should complete and what metrics should be produced as a result of that testing.

Useful metrics for project tracking include test case design completion percentage, cost of quality, defect detection percentage, defects found versus defects expected, and test case execution results. Developer metrics such as root cause, defect removal effectiveness, feature completion and unit test coverage should also be available to the test team.

For more on managing and tracking a project, see [Ensworth01].

6.2.5 Dealing with Trade-offs

Perfect projects are rare. Most projects require a trade-off between quality, schedule, budget and features. These trade-offs may be known at the beginning of the project or may become apparent during the testing phases. The test manager should be aware of the interdependencies between these project components and be prepared to participate in trade-off decisions. While the test manager should always defend quality, there are times when quality is not the most important aspect of a project. The test manager must be able to objectively analyze what changes in the schedule, budget or feature set may affect the overall quality of the project – for better or worse. For example, adding unplanned features into a project may reduce the overall quality as the testing resources are spread more thinly, but the new feature may be critical to the acceptability of the project.

While the test manager may not agree with project trade-offs, particularly those that may lower the overall quality of the delivered product, the manager must be able to provide the information to the stakeholders so that everyone can clearly understand the anticipated impact of the change. For example, if the net result of a proposed change is to reduce the testing time for all components by one third, what does that mean to the resultant quality? The test manager should be able to explain the impact of the reduction in testing in terms of the increased risk to the product quality and should clearly explain areas that will receive less or no testing as a result of the added features.

For more information on dealing with trade-offs, see [Galen04].

6.2.6 Change Management

As noted above, changes to project scope can influence the overall schedule, budget, features and quality goals of the project. Changes must also be managed in order to be tracked, scheduled and assessed regarding impact. A good change management system is a requirement for a well-managed project. The test manager, in particular, needs to track all changes that affect the original estimations and schedules. It’s important to remember that these changes are not limited to additions or deletions of features. Any change that affects the testing aspects of the project must be considered. For example, if the development team does not receive a server they need for compatibility testing, the test team will be affected and will likely be expected to pick up this additional testing.

The test manager must constantly be vigilant regarding any changes to the project that could affect the testing aspects of the project. If proper impact analysis is done for every change, the test manager must be included in the analysis. If little or no impact analysis is done by the team, the test manager should drive an impact analysis process and ensure adherence to that process.

In agile projects, change is an accepted, even welcome, reality. In order to deal with the constant changes inherent in agile projects, the test manager must have a lightweight and quickly adaptable solution that will be flexible enough to handle the changes. Established agile practices help to guide this flexibility by providing simple yet efficient solutions.

6.2.7 *Time Management*

Time is often a determining factor in the quality that is obtained for a release. The test manager must ensure that time is used efficiently by each member of the testing team. Efficient time usage includes the following:

- Limit time spent in meetings where the information could be more effectively conveyed via other communication means such as email
- Provide adequate time for test design to identify and resolve issues with the requirements
- Ensure all test systems are working correctly, prior to the start of testing
- Effectively smoke test incoming releases before propagating the code to multiple testers
- Adhere to an effective regression test strategy
- Use test execution automation for repetitive testing
- Build time for training into the schedules as an investment in the team for the future
- Ensure proper configuration management processes are followed
- Provide effective communication for the entire team, particularly across geographical boundaries and time zones
- Conform to the time box established for the iteration or sprint in an agile process (untested or insufficiently tested items are moved to a later iteration)

A test team should spend the majority of their time testing and making effective progress in their test activities. It is the test manager's job to ensure this happens.

6.3 Project Risk Management

Project risks are a reality in any project. The likelihood and the impact of the risks will vary between projects and organizations. The test manager must be able to identify these risks and must be able to develop a mitigation plan for each risk. For more information on software project risks, see [Jones93] and [DeMarco03].

6.3.1 *Managing Project Risks*

The purpose of project risk management is to track and control risks that can or will endanger the project. It is not necessary (or in many cases even possible) to eliminate all risks. The project manager must be aware from the start of all potential risks so the appropriate action can be taken if any of the risks are realized and become a threat to the project. The test manager and project manager must align testing activities with other risk-related measures within the project.

Throughout the project, it is possible that the test team and/or test manager can identify test-related project risks. Of course, ideally the most important project risks are identified during the planning activities. As with quality risks, any project risks identified should be analyzed to determine their likelihood and impact and thus their level of importance. It is also typically important to determine trigger dates (sometimes called confidence intervals), which are dates by which a contingency must be implemented in order to be effective in reducing the impact of the risk.

6.3.2 *Participating in Project-wide Risk Management*

Risks are not isolated to the testing phases of the project, but can occur anytime in the software development lifecycle. The test manager must consider the likelihood and impact of risks that can occur in the following areas. Some examples are provided here of the types of risks that could occur.

- Risks during the requirements process – inadequate access to potential customers, inadequate requirements documentation, insufficient time for requirements gathering, coding starting before the requirements are completed, etc.

- Risks during the design process – inadequate requirements, poorly chosen designs, incorrect or inadvisable implementation choices (wrong database, wrong language), missing design considerations such as performance, maintainability, etc.
- Risks during implementation, coding, unit testing and code review processes – insufficient unit testing, bad coding practices, inadequate review processes, lack of documentation in the code, no static analysis, etc.
- Risks during integration processes – integration not done methodically, integration platform not representative of testing environment, integration only done inter-functionally rather than end-to-end, lack of realistic data or realistic scenarios during testing, etc.

While the test manager may not be able to control these risks, the test manager should be aware when risks in other areas are realized and be able to assess the impact of that risk to the test organization. Controls should be in place to deal with risks that are anticipated and early detection should be possible via metrics and other monitoring techniques. The test manager should take a pro-active approach to identified risks and ensure that adequate effort is being taken to mitigate the risk within the project team. The test manager should also position and prepare the test team to deal effectively with risks that are realized and should be able to quickly adjust estimates and schedules based on these occurrences. The test manager should work with the project manager to determine who should manage such project-wide risks.

6.4 Quality Management

The expert test manager should recognize that their area, that of software testing, is a part of a larger set of quality management activities necessary to deliver quality software products and services which are fit for use by customers and users, on time, within budget, and with the desired features. Quality management also extends to the various deliverables created and used during the development lifecycle such as analysis deliverables (e.g., requirements documents), development deliverables (e.g., unit test documentation, code review information), and test deliverables (e.g., test cases, defect reports). The study and practice of the larger set of quality management activities is often called software quality assurance (SQA) or simply quality assurance (QA).

True quality management requires that software quality assurance activities be integrated into the entire software process and lifecycle. This involves key practices such as reviews, verification and validation, process standardization and software process maturity. It requires removal of people issues and other barriers to successful implementation of quality management systems in the organization. It also searches for inconsistencies in quality management and seeks methods for resolution. These issues are beyond the scope of test management, but intersect with it.

Some organizations have both testing teams and quality assurance teams, while other organizations rely on their test teams to serve both roles.

If the test team serves both roles, then it is essential to clarify with management what quality assurance and quality management activities and outcomes are expected from the test team. The test policy and test strategy should expand to be general quality policies and quality strategies, with testing being only one set of the quality-related activities.

Whether the test team is also the SQA team, or a separate SQA team exists, an integrated, consistent set of quality management and assurance processes, activities, and metrics should exist. If not, the test manager should work with the SQA manager (if that role exists) and with the organizational management to create it. If SQA processes, activities, and metrics do exist, the test manager should evaluate them, identify inconsistencies, and plan to resolve those inconsistencies. Effective quality management requires management buy-in.

For more information on quality management and the current state of software quality in the industry, see [Jones08] and [Kan02].

7. Test Project Evaluation and Reporting 510 mins.

Keywords:

convergence metric, dashboard, Ishikawa, Shewhart chart,

Learning Objectives for Test Project Evaluation and Reporting

7.1 Introduction
No learning objectives for this section

7.2	**Tracking Information**
LO 7.2.1	(K4) For a given level of testing determine the most effective metrics to be used to control the project
7.3	**Evaluating and Using Information - Internal Reporting**
LO 7.3.1	(K6) Create an effective test results report set for a given project for internal use
7.4	**Sharing Information - External Reporting**
LO 7.4.1	(K6) Create an accurate, comprehensible and audience-appropriate test status dashboard
LO 7.4.2	(K4) Given a specific project and project team, determine the proper level and media for providing weekly test reports
7.5	**Test Results Reporting and Interpretation**
LO 7.5.1	(K6) Given a particular lifecycle, project, and product, design appropriate test results reporting processes and dashboards, including metrics, taking into account the information needs and level of sophistication of the various stakeholders.
LO 7.5.2	(K4) Analyze a given test situation during any part of the test process, including appropriate qualitative interpretation, based on a given test dashboard and set of metrics.
7.6	**Statistical Quality Control Techniques**
LO 7.6.1	(K2) Explain why a test manager would need to understand basic quality control techniques

7.1 Introduction

The test manager must be able to objectively track a project that is in progress, evaluate the data, and take any corrective actions necessary. The manager must also be able to report the appropriate information both to internal and external stakeholders. For more information on test evaluation and reporting, see [Black03].

7.2 Tracking Information

Part of controlling a project is knowing what data to track that will be indicative of the progress and quality of the project. Tracked information must be objective, informative and understandable in order to be useful. Tools should be leveraged to gather and report the necessary information. Tools that can help automate the reporting process are valuable in this endeavor.

The data that should be tracked for a project include:

- Test cases designed (planned vs. actual)
- Defect metrics (by status, by severity/priority, by area, trending, convergence, resolution turnaround, re-open rate, rejection rate)
- Test execution metrics (by status, automated vs. manual, execution rates, convergence, planned vs. actual, number of test passes)
- Coverage metrics (code, requirements, risks)
- Management metrics (resource hours, setup time, downtime incurred, planned vs. actual)

Trending analysis is an important factor in tracking information. “Point in time” information gives a snapshot view, but may overlook overall trends that indicate schedule, resource or quality issues. The detailed information must be tracked so the trending information can be derived.

Part of tracking the data may also include “cleaning” the data to ensure that the data being reported is correct. This flows through to the usage guidelines of the tools that are used to gather the information. If people are reporting incorrect data on test case completion, your summary reports will also be inaccurate and may result in presenting an invalid or incomplete picture of the project status.

7.3 Evaluating and Using Information - Internal Reporting

The information gathered must be processed to be useful. This includes conducting a careful evaluation of the information both to validate the data and to compare the actual to the expected values. If the information does not match the project expectations, investigation will be required to understand the variance and perhaps take corrective actions. For example, if the expected defect discovery rate was 100 defects per week but the actual rate is 50 defects per week, this could mean that the software quality is considerably better than expected, or that the testers are out sick, or that the test environment is down so people have not been able to test.

Evaluating the information requires expertise. Not only must the evaluator understand the information that has been gathered, they must be able to effectively extract it, interpret it and research any variances. Reporting raw data is dangerous and may result in loss of credibility if the data later is discovered to be flawed.

Once the data has been evaluated and verified for accuracy, it can be used for internal reporting and action plans. Internal reports are circulated within the project team. For example, management metrics might be kept internal to the testing team while corrective actions are put in place to increase tester efficiency. Detailed defect metrics are usually shared with the development team, but may not be circulated to the project managers or external team members until they have been summarized.

Internal reporting is used by the test manager to manage and control the testing project. Some samples of internal reports that test managers may find useful include:

- Daily test design/execution status
- Daily defect status
- Weekly defect trending
- Daily test environment reports and status
- Resource availability

While internal reports may be less formal than external reports, the quality of the data is just as critical, particularly when management decisions will be based on the information in these reports. The time spent to develop good reporting tools and methods will result in time savings throughout the project.

7.4 Sharing Information - External Reporting

The purpose of external reporting is to accurately portray the testing status of the project in terms that are understandable to the audience. It's not unusual for the testing status to be at variance with the status that may be presented by other aspects of the project. This doesn't mean that other reports are necessarily incorrect but rather that they are reporting from a different perspective. If the test manager realizes this is a problem area, it is a good practice to work with the other reporting managers to unify the data rather than risk confusing the audience.

One of the biggest challenges in external reporting is to tailor the information presentation to the audience. For example, while a detailed defect report might be effective for the development team, an executive will find a dashboard that presents the high level snapshot more informative. Generally, the higher the consumer is in an organization, general trending information may be more appropriate than a detailed report. Charting and visually attractive graphs become more important as the technical expertise of the consumer drops. Because executives may have little time to devote to reviewing reports, quick visuals, easily interpreted charts and trending diagrams are particularly useful in getting the point across quickly and accurately.

Another consideration when preparing external reports is the amount of information to supply. For example, if the reports are to be presented in a meeting, the presenter will be there to supply additional detail information if needed. If, on the other hand, the reports are published as a dashboard on the intranet, it may be useful to supply a drill down capability that will access the details if the reader is interested. If only the high level information is supplied and there is no detailed information to support it, credibility may be lost.

External reporting varies with the criticality of the project, the product domain and the expertise of the audience. Some examples of commonly used external reports include:

- Dashboard showing the current testing and defect status
- Resource and budget reports
- Risk mitigation chart
- Test coverage chart (requirements or risk-based)
- Defect trends
- Milestone achievement
- Project health indicators (red, yellow, green)
- Predictive metrics
- "Go/No go" recommendations

External reporting should be done on a regular basis, usually weekly. This trains the audience to expect to receive the information and to use that information in their own planning and decision making. There are various ways of effectively publishing this information. Some of these include:

- Real time dashboard
- Published to a known site on the intranet
- Emailed out to a distribution list
- Included in regular project status meetings
- Physically posted in an office window

External reporting is only effective when it is received and understood by the target audience. The reports must be specifically targeted for the audience and must contain the proper level of detail. Overwhelming the audience with details that they don't understand or don't care about may result in their disregarding the report in its entirety.

Success as a test manager is tightly related to the ability to accurately communicate testing status to those outside the testing organization. The test manager must consider the audience and the message and tailor the information appropriately.

7.5 Test Results Reporting and Interpretation

The Advanced syllabus discusses a number of different types of metrics to track during various phases of the software development lifecycle and the testing phases in particular. This section concentrates on how to effectively report the information gathered and how to interpret those results, both for internal use as well as for the other consumers of that information. Automated and manual testing are grouped together in this discussion.

This section looks at each phase of the fundamental test process, quickly reviews the metrics that should be tracked and then discusses how these metrics should be interpreted.

7.5.1 Planning and Control

From the Advanced syllabus, test planning involves the identification and implementation of all of the activities and resources required to meet the mission and objectives identified in the test strategy. Test control is an ongoing activity. It involves comparing actual progress against the plan, reporting the status and responding to the information to change and guide the testing as needed.

- Metrics for the test planning and control processes
  - Risk coverage
  - Test case development and execution status
  - Defect discovery trends
  - Planned versus actual hours for testing activities
- Interpreting the data
  - Adherence to the estimated schedule and budget is critical to all test projects. These metrics can be used to quickly determine the planned versus actual status of the testing processes.
  - Quality of the software can be determined based on the expected versus actual defect discovery trends. Finding more defects, or a larger proportion of critical defects, at a higher rate than was anticipated can indicate significant incoming quality issues.
  - Properly sequenced risk mitigation can be determined by checking that the team has created and executed tests for the highest risk items prior to the lower risk items. If lower risk items are being addressed first, the testing approach needs to be reviewed and corrected.
  - Insufficient test progress is sometimes due to insufficient productive test hours. If the planned versus actual test hours is not meeting expectations, test coverage will be insufficient and the project will quickly start to slip the schedule. The test manager must carefully investigate the causes of the lack of progress to determine which factors are occurring (test systems not ready, insufficient incoming quality, test documentation not prepared, resources not available, etc.) and create a plan to rectify these problems.

7.5.2 Analysis and Design

Through the test planning phase, a set of test objectives is defined. During the test analysis and design phase, the test team takes these objectives and identifies the test conditions and creates the test cases that exercise the identified test conditions.

- Metrics for analysis and design
  - Coverage metrics
  - Defects found during test design

  - Defects found during review cycles
- Interpreting the data
  - Coverage metrics include coverage of the requirements by the test cases, coverage of the risk items by the test cases, coverage of the quality characteristics and any other measurable coverage type. If coverage is insufficient in any area, an unaddressed risk exists. That risk must be analyzed to determine the best method to deal with it based on the impact and likelihood.
  - Defects found during the review cycles and during test design, if resolved, prevent those defects from escaping to the next phase. If the defect is identified but not resolved, there is still additional work incurred by the existence of that defect (documentation, status reporting, possible workarounds, etc.) but the time spent dealing with a known defect is usually less than that spent detecting a previously unfound defect. Cost (in money or time) is often associated with these defects and that cost is reported, usually in terms of cost savings, as part of the general test reporting.
  - Defect trending is important at this phase also. If high numbers (higher than planned) of defects are found in the review cycles, this may indicate a lower quality of the work products than was anticipated. If so, multiple review cycles may be required to reach an acceptable quality before these work products move to the next phase in the lifecycle.

7.5.3 Implementation and Execution

Test implementation and execution includes organizing the test cases, finalizing the test data and test environments and creating the execution schedule as well as actually executing the tests.

- Metrics for implementation and execution
  - Percentages of environments, systems and data configured and ready for testing
  - Percentage of test data created and used
  - Percentage of test conditions and cases covered by test execution and the results of that execution
  - Percentage of test cases that are automated and results of automation execution
  - Percentage of test cases that are marked for inclusion in the regression test suites
  - Time spent maintaining automated and manual test cases
  - Breakdown between types of testing, e.g., black-box, white-box, experience-based
- Interpreting the data
  - To start testing without 100% of the test systems and environments ready may be an acceptable, even a preferred strategy. However, if the missing environments are those needed for the highest risk items, that will force the higher risk testing toward the end of the testing cycles, and that is not acceptable. The test manager must not look only at the percentage numbers, but also understand if there is any schedule impact or changes required to the overall testing plan in order to accommodate any configuration issues.
  - Similar to the environment issues, test data availability and accuracy can also impact the order of test execution and can significantly reduce the efficiency of the test team. For example, if the requisite amount of test data is not available, forcing repeated re-use, concurrent testing of some areas may not be possible. Automated tests may have to be changed to re-use the data and reset it to a quiescent state after use.
  - Test coverage must be closely monitored for several reasons. If the coverage is not proceeding according to plan, risks may be appearing where they were not planned. For example, a testing area that is blocked may contain significant defects that will result in architectural changes to the system, thus delaying performance testing. The test manager also must monitor pass/fail information to determine if the expected pass/fail rate is being achieved. If a higher number of test cases are failing than was expected, additional test cycles may be required. In addition, investigation is needed as to why the quality level is lower than expected. If a higher number of test cases is passing than was expected, the

test manager needs to determine if testing has been sufficient, where the extra time in the schedule should be focused and if the plans should be adjusted for future testing.

- The percentage of test automation achieved and the results of the automation should be monitored by the test manager to ensure the automation effort is achieving its goals and that the test automation results are being integrated into the overall testing results. If the test automation only covers 5% of the testing, but is finding 50% of the defects, investigation is needed to determine the types of defects being found, verification that the automation is doing the correct validation and to investigate why the manual testing is finding so few problems.
- Time spent on maintenance of test documentation can be a significant factor in a project. A test manager may see the maintenance time increase beyond schedule when the requirements used to create the test cases have changed since the test cases were created. Maintenance time may also increase when the delivered code doesn't match the requirements, the environments or data have changed, the interfacing systems have changed and myriad other reasons. Since time spent maintaining test cases is time that is not being spent actually testing, this maintenance effort can have a significant impact on the overall testing project.
- Test type breakdown is interesting from several standpoints. The test manager needs to know how the testing time is being allocated. The test manager also needs to know how productive the various testing types are in terms of defect detections, risk coverage and tester skill usage. If the test manager determines that 60% of the defects are being found with exploratory testing, the test allocations may need to change to focus on more exploratory testing. The test manager may also want to understand why the scripted testing is finding fewer bugs than expected. Perhaps the software has stabilized in those areas and the scripted testing is less productive than it has been in the past. If white box testing is finding no defects, the test manager will want to understand what code coverage levels are being achieved by that testing.

7.5.4 Evaluating Exit Criteria and Reporting

The test manager evaluates the exit criteria throughout the test project to ensure steady progress toward meeting those criteria. Reporting for the exit process is a summarization of the testing activities and the metrics discussed in the sections above, but with a total project perspective. There should be no new metrics introduced at this point in the testing project – metrics used here should have been planned and tracked throughout the project as noted above.

- Metrics finalized at this stage
  - Test conditions, test cases or test specifications executed versus planned and the final status of each
  - Number of testing cycles used versus planned
  - Number and complexity of changes to the software that were introduced, implemented and tested
  - Schedule and budget adherence and variances
  - Risk mitigation
  - Planned versus actual effective testing time
- Interpreting the data
  - At this point, these metrics can no longer be used to make changes for the current project (unless testing time is to be extended or re-work is planned). Metrics at this point are used for process improvement for future projects.
  - Metrics gathered at this stage are reported in the summary report which is prepared at the conclusion of the test effort.

7.5.5 Test Closure Activities

When test execution is complete, the test manager must still conduct the test closure activities. These consist of ensuring that all the test work is actually completed, delivering the final work products, participating in retrospective meetings, and archiving the data, test cases, system configurations, etc. for this project.

- Metrics completed as part of the test closure activities
  - Percentage of test cases that can be re-used on future projects
  - Percentage of test cases that should be moved to the regression test suite
  - Percentage of the defects that have reached the end of their lifecycle (will not be fixed, no action required, enhancement request, etc.)
- Interpretation of the metrics
  - As with the previous stage, these metrics should have been tracked throughout the testing process but this is a good time to review them. Test cases that can be re-used are generally considered to be a better investment of time than those that were used once and are no longer usable. In rapidly changing environments, such as those following an agile lifecycle, test cases developed in the early iterations are often obsolete by the time the later iterations occur. Time invested in these “throw away” test cases should be reviewed to see if it was well-spent or if another test design methodology would have been more efficient (for example, using checklists rather than scripted test cases).
  - Test cases are moved to the regression test suite because the features they test are expected to remain constant are good candidates for review by the automation team for potential automation.
  - Defects that will not be resolved should be reviewed (as always) to determine if the test team is following good practices in identifying and documenting these items. If defects are marked “never fix” because they are not reproducible, the test manager should investigate to see if additional testing was required or better documentation was needed in these defect reports.

Test closure activities may identify areas for additional process improvements and may also identify areas that need to be tracked more closely on future projects.

7.6 Statistical Quality Control Techniques

The test manager, being in a quality control role, should be able to apply basic statistical quality control techniques such as control charts and cause/effect graphs. Examples include the following:

- Using an Ishikawa (fishbone) diagram to identify possible causes for automated test reliability problems
- Using a Shewhart chart to determine if the testing process is under control

Additional information regarding statistical quality control techniques can be found in other quality assurance oriented documentation. For more on statistical quality control, see [Ishikawa91] and [Juran10].

8. Testing Considerations for Domain and Project Factors 270 mins.

Keywords:

feature-driven development, test-driven development

Learning Objectives for Testing Considerations for Domain and Project Factors

8.1 Test Management Considerations for Lifecycle Models

LO 8.1.1	(K6) Evaluate and report advantages and disadvantages associated with each lifecycle model in given situations
LO 8.1.2	(K2) Describe the concepts usually found in agile projects which may influence the testing approach

8.2 Managing Partial Lifecycle Projects

LO 8.2.1	(K6) Compare the various partial project types, discussing the differences between these projects and pure development projects

8.3 Release Considerations

LO 8.3.1	(K4) Analyze the business context with respect to deployment, installation, release management, and/or product roadmap and determine the influence on testing.

8.1 Test Management Considerations for Lifecycle Models

Lifecycle models can strongly influence a test manager's approach to a given project. The lifecycle determines the moment of involvement, the level of involvement, the amount of documentation (both available and produced) and the testing goals.

8.1.1 Comparison of Lifecycle Models

The following chart provides a comparison of the lifecycle models and the testing implications:

	Moment of involvement	**Documentation supplied**	**Documentation created**	**Level of involvement**	**Testing Goals**
Waterfall	After code is complete	Complete specifications	Complete test documentation	None until code complete	System testing, some acceptance testing
V-model	Reviews during requirements phases, planning during code development, testing when code is complete	Complete specification	Complete test documentation	Planning early, reviews of requirements documents, testing when code complete	Integration, system and acceptance testing
Iterative / Incremental	Involved for first testable iteration of code	Iterative specific documentation	Iteration test plans, iteration relative test cases, some automation	"Just in time" planning and testing for the iteration, emphasis on regression testing	Integration, system and acceptance testing
Agile	Project initiation	Light, if any	Test automation, test checklists	Embedded in project team from initiation	All levels of testing, including some unit

The test team's time and level of involvement varies considerably based on the project lifecycle. Since the lifecycle is often determined by the development team, the test manager must monitor these decisions and determine the appropriate time and level of involvement.

A test manager should be familiar with the most common lifecycle models, as already discussed in the Advanced Level syllabus. In addition to that, the expert test manager must also be familiar with agile approaches and how to participate effectively in an agile project.

In actual practice, many organizations implement lifecycles that incorporate elements of multiple lifecycle models. For a discussion of this approach, see [Boehm03].

8.1.2 Agile Methods

Agile methods are a group of approaches to the software development lifecycle that are fundamentally different from traditional models such as the V-model. The most common of the agile approaches are:

- Scrum - focuses on roles and project management issues in an iterative project delivery
- eXtreme Programming (XP) - focuses on technical practices used by a software development team
- Kanban - focuses on increasing value flow through using self-organizing teams and actively limiting the work in progress

Although these approaches were developed in parallel, they are often combined in a project to leverage the complementary aspects.

There are some specific practices and sets of values associated with agile methods. Development practices such as feature-driven development and testing practices such as test-driven development are often found in agile projects. Although the test manager will need to adapt to the agile methods, it is important to remember that many good practices in testing common to other models still apply.

To become familiar with agile, it is helpful to become familiar with the Agile Manifesto which defines the 12 principles of the agile methodology [Agile Manifesto-Web]. These are:

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for a shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within the development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity -- the art of maximizing the amount of work not done -- is essential.
11. The best architectures, requirements and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Agile methods rely on high attention to quality throughout the lifecycle. Skilled testers and test managers are critical to the success of agile projects, but the roles are somewhat altered. For example, the role of test manager may not exist but may be reformed into the role of Scrum Master, Product Owner or even an individual contributor. The test manager must maintain the role as the quality/testing coach to ensure that all team members are contributing to building a quality product. In this way, the test manager becomes an advocate for the testing activities of the project rather than for the testing organization itself.

The testing approach on agile projects tends to differ from the traditional lifecycle models. In particular, the following concepts are usually found in agile projects:

- The whole team is responsible for quality. Skilled testers and test managers learn more about other processes, such as design and development, and help other team members to understand the need for testing. Each individual must assume responsibility for quality.
- Lightweight solutions for processes and tooling are preferred. The solutions may change frequently within a project. Simple solutions help to promote efficiency.

- Test automation must be utilized effectively to minimize the cost of regression testing. Tests should be run frequently. Development should take an active role in developing and maintaining the automation.
- The definition of the term “done” or “complete” must be established. The testers need to contribute to the definition and be sure it is measurable and understandable.
- Good communication is vital because changes are frequent and documentation is light. Cross training within the group is necessary to provide the best coverage of all tasks in an iteration.
- Testing is conducted throughout an iteration and must cover all aspects to deliver a completed “product” at the end of the iteration.
- Integration of the software into a larger system must be done early and often in incremental stages.
- Continuous process improvement must be implemented and planned approaches should change as needed to react to project realities.

In general the analysis and design techniques of testing remain the same in agile projects as with other lifecycles. The role of the tester and test manager must adapt and the test effort must be tailored to the needs of the iteration. Best practices in testing such as defining test conditions and selecting appropriate testing techniques are still applicable.

In addition to the standard approaches to agile projects, test managers may also be faced with some unique challenges. One of the premises of agile projects is that the team is tightly integrated, usually co-located (no walls, one giant cube), and relies on person-to-person communication rather than formal documentation to convey ideas, plans, schedules, requirements and test results. In the global economy, co-location can be particularly challenging. If the team cannot be co-located, easy and quick communication methods must be established. For example, a daily status meeting via phone call to synchronize the entire team and the use of an Instant Messaging tool during the working period will often suffice to enable communication. Collaboration software can be used to provide up to the minute update and editing capabilities for project documentation such as user stories, design notes, implementation information, etc. Agile projects are driven by the capability of the team to achieve real time communication. Tools are being developed rapidly to meet the needs of the distributed team that needs to feel like it is sitting together in one location.

Time zone issues are still a problem for agile projects in that communication is necessarily forced to be delayed. Again, the team can work around this effectively by having overlap hours that allow all team members to be online and accessible during certain windows of the day or night. While this may not provide an optimal working environment (particularly for the late night or early morning meetings) it is necessary to facilitate team communication.

Test documentation in an agile project is also minimal due to the maintenance burden that would be incurred with more detailed documentation of evolving software. Checklists are often used to guide the testing and exploratory testing is heavily used for each new iteration of the software. Checklists are built and updated based on the features targeted for implementation and then modified based on the features that actually are implemented.

Adapting current test practices to an agile project can be challenging. In some cases, new practices must be developed to accommodate the changed project approach. It is important for the test manager to leverage the existing good practices while also adapting to the new model.

8.2 Managing Partial Lifecycle Projects

The test manager is often involved in projects for which all or part of the development is being done by an external team. This type of project brings its own test management challenges that are sometimes caused by late involvement, sometimes by lack of documentation and sometimes by multiple ownership. This section takes a closer look at some of these partial lifecycle projects.

8.2.1 Integration Projects

The role of the test manager when working with third parties was discussed in a previous chapter. This section focuses on the testing lifecycle changes that may be required when dealing with an integration project. The moment of involvement for an externally developed project (or partially externally developed project) is often later than a project which is developed in house. The code is usually fully developed and assumed to be at least unit tested at the time it is received for integration. This means the test team may not have been involved in requirements and design reviews, and that the planning, analysis and design time for the project are significantly reduced.

It is important to get a clear definition of the “expected” incoming quality level. Was the product tested by a different test group? Is it considered ready for general release or just ready for further development and integration? This information may or may not be available (and may or may not be accurate). The test manager is smart to start with smoke testing to verify the functionality of the documented features. Once that is confirmed, spot checking the known and expected detailed functionality should proceed. This may require the development or re-use of existing drivers and stubs to support the testing. This can be done with exploratory testing or even checklist testing if a checklist is available. At this point, a quality assessment can be done to determine if more testing is needed and what type of testing will be needed. Software that has already exhibited significant defects at this point of the testing will need additional system testing. Software that is yielding no unknown defects is ready for non-functional testing (performance, usability, etc.) as warranted by the intended use.

Commercial-off-the-shelf (COTS) software is generally assumed to be of higher quality and ready for general release. This, of course, may prove to be a false assumption after the preliminary testing, but that is generally the working assumption until proven otherwise. Additional testing in this area will depend on the level of integration, the responsibility for the testing, and the experience with the vendor. For example, if the team is producing printer drivers and wants to be sure images generated by the vendor’s package can be printed, the goal is not to test the vendor’s package but rather to ensure that a generated image (or a variety of those) is printable. If the team is testing a component that was developed by a third party that is to be integrated into the functionality of an internally produced application such that a single interface will be presented to the user, thorough testing will be needed for the functionality of the third party piece (or verify the testing that was done by another testing group) as well as the integration points with your application.

In some cases, the product that is received from the third party will require a formal acceptance signoff. In this case, if functional and non-functional testing is not required, the test manager will prepare the acceptance criteria, conduct the test and report the results. A successful acceptance test may conclude the test team’s involvement. An unsuccessful acceptance test may indicate the need for functional/non-functional testing and further involvement from the test team.

8.2.2 Maintenance Projects

Software that is in production periodically requires a maintenance release. This may be due to the need to migrate the system to a new environment, retire a portion of the system or to add features and/or provide fixes. Maintenance releases are sometimes scheduled in advance with defined features sets. Maintenance releases may also take the form of “emergency” fixes with little planning (if any). The most important aspect of maintenance testing is to ensure no regressions are introduced with the fixes/new features. As such, an existing regression test suite is valuable in dealing with the last minute scheduling and unexpected scope of the usual maintenance release. An automated regression suite is invaluable in this case, freeing manual testing resources to verify the fixes and new features.

Depending on the urgency of the fix, normal processes may be bypassed by the development group, including unit and integration testing. If so, the test group may need to assume the code is virtually untested. As such, more thorough testing will be required during the system testing phase thus increasing the time required for testing. Impact analysis is helpful in targeting both the feature testing as well as the regression testing.

Any maintenance project requires a degree of change management. In most organizations there is a tendency to add additional fixes/changes to the planned release because it is a convenient carrier. Additional changes incur additional risk and the test manager must inform the project team regarding the risk being incurred by any additions to the planned maintenance release.

8.2.3 Hardware/Software and Embedded Systems

Hardware/software combination projects and embedded systems projects require some specific planning steps. In particular, because the test configuration tends to be more complex, more time should be allotted in the schedule for set up and testing of the test configurations. Prototype equipment, firmware and tools are often involved in these environments, resulting in more frequent equipment failure and replacement issues that may cause unexpected downtime. The testing schedule should allow for outages of equipment and provide ways to utilize the test resources during system downtime.

Dealing with prototype equipment and prototype tools such as diagnostics can further complicate testing. Prototype equipment may exhibit behavior that the production models will not exhibit and the software might not be intended to handle that aberrant behavior. In this case, reporting a defect might not be appropriate. Anytime pre-production models are used in testing, the test team must work closely with the hardware/firmware developers as well as the software developers to understand what is changing. One word of caution though, this can work the other way. Prototype hardware may exhibit incorrect behavior and the software may accept that incorrect behavior when it should raise an error. This will mask the problem that will only appear with the properly working hardware.

Testing these types of systems sometimes requires a degree of hardware/firmware expertise in the testing team. Troubleshooting issues can be difficult and time consuming, and retesting time can be extensive if many releases/changes occur for the hardware and the firmware. Without a close working relationship and an awareness of what the other teams are doing, the continually changing hardware/firmware/software can destroy the forward progress of the testing team due to the repeated integration testing.

An integration strategy is important when testing these types of systems. Anticipating and planning for problems that may hinder the integration helps to avoid project delays. An integration test lab is often used to separate the untested equipment/firmware from the software testing environment. The equipment/firmware has to “graduate” from the integration lab before it can enter the system testing phase.

The test manager plays an important role in helping to coordinate the schedules of the components so that efficient testing can occur. This usually requires multiple schedule meetings, frequent status updates, and re-ordering of feature delivery to enable testing while also supporting a logical development schedule.

8.2.4 Safety-critical Systems

Safety-critical systems provide some special challenges, including the need for additional testing for conformance to regulations and published standards. The test team should expect to be involved in all phases of the lifecycle and should have time allocated for requirements reviews, design reviews, code reviews, unit test reviews, integration, system and acceptance testing. In addition to the continuous involvement in the lifecycle of the project, the test team will also spend significant time creating, verifying and reviewing documentation, including test documentation, as well as maintaining traceability matrices. All defects noted throughout the lifecycle must be documented and all documents must be versioned, including test documentation. Documents must be checked for internal consistency and all changes must be verified and signed off. Version description documents are commonly used to track each version of each document, noting all changes that occurred per version.

Compliance to industry specific regulations and standards may dictate levels of documentation, specific tools that can and cannot be used, automated test coverage, code and requirements coverage, exit criteria, acceptance criteria, defect classification and other specifics required by the industry and the product.

See [Leveson95] for more information on safety-critical systems.

8.3 Release Considerations

The test manager may or may not be able to have a voice in release considerations. This is often the domain of the business, marketing and product managers. The test manager does, however, need to be aware of considerations for the different release methods, products and customers.

8.3.1 Market Demand

Depending on market demands it may make sense to release a product before it is feature complete. This may be because competitive products are already in the market, because this is a replacement for a product that has been problematic or because the organization wants to get early feedback on a new concept. Whatever the reason, the test manager may have to adjust the exit criteria for the testing phases to support this type of partial release. Exit criteria adjustments should be done only with the approval of the project team as this can have quality implications and support implications.

8.3.2 Ease of Maintenance

Release decisions may also consider the ease of delivering fixes to the customers. Depending on the product, fixes may be automatically downloaded, may require media delivery or may be available for the customer to download as needed. The ease of both obtaining and installing fixes may determine the acceptability of releasing a product prior to final testing approval.

Pre-releasing with a plan for fixes can be a dangerous approach if there are risks of bad publicity, unhappy customers or product failure. Non-critical software may be released in this way with the assumption that the user base would rather deal with some defects that will be resolved quickly than wait for a more solid release. Again, this is usually a decision that is made by the business, marketing or the product manager. The test manager needs to be able to supply accurate information regarding the status of the testing, the residual risk, the known issues and an approximate schedule for the delivery of fixes.

Another consideration for delivering frequent fixes is the difficulty the customer may have in adopting those fixes. For example, if the team is working with software that exposes APIs for the customers to use, changing those APIs may require additional coding on the part of the customer, making them less likely to quickly adopt changes and fixes. Multiple fixes that are not adopted can make support difficult because of a user base that is running multiple different versions of the software.

8.3.3 Ease of Installation

When considering a maintenance release, the test team must understand any effects the fixes will have on the base software. Is a data conversion required? Will coding be required? Will the customer need to re-train their users because of a change to the user interface? Will the customer incur downtime during the installation and implementation of the fix?

Some fixes are easy to install and others are difficult. If customers have the capability to pick and choose among a set of fixes, conversion and installation requirements may become very complicated both from a testing and a support aspect. If five separate fixes have been released and the third fix required a conversion, there is an assumption in the fourth and fifth fixes that the conversion will have been done. If the customer never took the third fix, but wants the fifth fix, it may not work on their system. Fixes are often bundled together to prevent these types of situations, but this may require more testing effort.

The deployment mechanism/capability must also be tested. This is to ensure that the proper fix is delivered in the proper format to the proper systems. This may include testing the installation procedure, any automated wizards, various upgrade paths, de-installation procedures and mechanisms used to actually deliver the fix to the user. As with initial installation testing, upgrade capabilities must be thoroughly tested as a delivered part of the product.

9. Evaluating Effectiveness and Efficiency 195 mins.

Keywords:

effectiveness, efficiency

Learning Objectives for Evaluating Effectiveness and Efficiency

9.1 Introduction

None for this section.

9.2 Effectiveness, Efficiency and Satisfaction Metrics for the Test Process

LO 9.2.1 (K5) Explain the purpose of tracking effectiveness, efficiency and satisfaction metrics and give examples that would apply to a test project

9.3 Effectiveness, Efficiency and Satisfaction Metrics for the Test Policy Objectives

LO 9.3.1 (K6) For a given test policy, define, evaluate, and report effectiveness, efficiency, and satisfaction metrics

9.4 Project Retrospectives

LO 9.4.1 (K6) For a given completed project, manage a project retrospective, analyze data gathered during the project retrospective, and formulate plans for improvement

9.1 Introduction

A test manager must continually evaluate the effectiveness and efficiency of the test group both to look for improvement opportunities as well as to monitor the on-going test efforts.

9.2 Effectiveness, Efficiency and Satisfaction Metrics for the Test Process

Effectiveness metrics examine the extent to which the test process produces the proper work products and outcomes, including risk mitigation (i.e., how well did we do?). Efficiency metrics examine the effort, duration, and/or resource expenditures required to achieve those work products and outcomes (i.e., how well did we deliver value?). Satisfaction metrics examine the extent to which the test process elegantly delivers both actual and perceived value to test stakeholders (i.e., how is it perceived?).

Effectiveness metrics:

- Percentage of new tasks identified during control activities
- Percentage of new and changed test conditions identified after initial analysis
- Percentage of requirements covered by identified test conditions
- Surveys of test analysts' understanding of the tests to be designed to cover test conditions
- Percentage of test conditions covered by tests
- Appropriate use of testing techniques to verify test coverage
- Post-project evaluation of exit criteria escapes, e.g., unnoticed set of planned tests that were neither run nor deliberately skipped prior to exit
- Defect escape rate
- Increase in test objective achievement on current project based on lessons learned from previous projects

Efficiency metrics:

- Effort, duration, and resource impact of unmanaged changes to the testing plan
- Percentage of time lost due to unforeseen blockages
- Consistency of detail in test documentation
- Average effort required per test designed
- Percentage of tests automated
- Average effort required per test implemented
- Effort, duration, and/or resource savings achieved on current project based on lessons learned from previous projects
- Project overhead associated with results reporting
- Effort, duration, and/or resource savings achieved on current project based on testing work product re-use from previous projects

Satisfaction metrics:

- Project stakeholder survey results
- Extent to which the planning and scheduling is aligned with the business objectives
- Surveys of the recipients of test reports
- Surveys of testers on the usefulness of test closure activities

Test managers should remember that testers and other project team members will change their behaviors to attain satisfactory outcomes in terms of process metrics. In the absence of clearly defined goals from management, metrics can result in undesirable distortions in the actions taken and work products delivered by individual contributors and managers. Therefore, the test manager should carefully plan the introduction of process metrics, and constantly monitor for unintended side effects.

Test managers should also remember that these process metrics are strongly affected by overall project and organizational behaviors, activities, and attributes. Successful execution of upstream processes such as system requirements specification, design, coding, unit testing, and release engineering can have significant positive impact on test process effectiveness, efficiency, and satisfaction; unsuccessful upstream processes can have the opposite effect on the test process. The test manager tends to have a good overall view of all the processes and should leverage that information to help influence the overall process flow.

9.3 Effectiveness, Efficiency and Satisfaction Metrics for the Test Policy Objectives

After the test objectives are identified and documented in a test policy, metrics should be defined that measure effectiveness, efficiency, and satisfaction. This section includes examples of such metrics for typical test objectives. Each test project is different, so the metrics that are appropriate for a specific project may vary.

Consider the following typical test objective: Find important defects that could affect customer or user satisfaction, and produce enough information about these defects so developers (or other work product authors) can fix them prior to release. Effectiveness, efficiency, and satisfaction metrics for this objective can include the following:

- Percentage of defects detected (effectiveness)
- Percentage of critical defects detected (effectiveness)
- Percentage of defects rejected by development (efficiency)
- Percentage of critical defects that escaped to production (effectiveness)
- Defect report recipient satisfaction with information delivered (satisfaction)
- Cost per defect found during testing, possibly as a percentage of costs per defect found in production (efficiency)

Consider the following typical test objective: Manage risks by running important tests that relate to key quality risks, in risk order, reducing the risk to the quality of the system to a known and acceptable level prior to release. Effectiveness, efficiency, and satisfaction metrics for this objective can include the following:

- Percentage of critical bugs found early in test execution (effectiveness)
- Percentage of critical tests run early in test execution (effectiveness)
- Stakeholder perception of accepted risk prior to release, possibly in comparison to perception of that risk after some period in production (satisfaction)
- Percentage of identified risks covered by (executed) tests (effectiveness)
- Average cost per risk item covered during testing (efficiency)

Consider the following typical test objective: Provide the project team with important information about quality, testing, and readiness to release. Effectiveness, efficiency, and satisfaction metrics for this objective can include the following:

- Survey of project team about sufficiency, importance, and timeliness of information provided by testing (effectiveness and satisfaction)
- Overhead costs associated with producing test reports (efficiency)

Consider the following typical test objective: Find less-important defects, including workarounds for these defects, to enable customer support or help desk to resolve user problems related to these defects if they are not fixed prior to release. Effectiveness, efficiency, and satisfaction metrics for this objective can include the following:

- Survey of customer support or help desk about sufficiency, importance, and timeliness of information provided by testing (effectiveness and satisfaction)
- Stakeholder perception that defect information included sufficient information for appropriate defer/fix decisions (effectiveness)
- Costs associated with delivering information to customer support or help desk staff (efficiency)

For a detailed discussion on derivation of metrics for test objectives, see [Riley & Goucher 09].

The same cautions mentioned in the previous section, regarding unintended side effects and the impact of upstream processes, applies to the metrics discussed in this section.

9.4 Project Retrospectives

A project retrospective meeting is a team meeting that assesses the degree to which the team effectively and efficiently achieved its objectives for the project or activities just concluded, and team satisfaction with the way in which those objectives were achieved. In agile projects, retrospectives are crucial to the overall success of the team. By having a retrospective at the end of each iteration, the agile team has an opportunity to review processes, adapt to changing conditions and interact in a way that helps to identify and resolve any interpersonal issues.

A test manager should be able to organize and moderate an effective retrospective meeting. Because these meetings are sometimes discussing controversial points, the following rules should be applied:

- Encourage open exchanges of ideas, but discourage personal criticism
- Have participants come ready to present data to illustrate their points, and avoid reliance on opinions or feelings
- Have people bring ideas for improvement on future projects, rather than simply complaints about how badly something went
- Have people bring examples of what went well and should be continued

According to [Derby&Larsen 06] a retrospective should follow a structure as outlined here:

- Set the stage
- Gather the data
- Generate insights
- Detect what to do
- Close the retrospective

Of course, the objective of the retrospective is to deliver outcomes that result in actual change. Successful retrospectives involve the following:

- A library of data, reports, and minutes from retrospective meetings, with this information readily available as a resource for future projects.
- A root cause analysis (RCA) session that concentrates on identifying the source of the problems encountered and suggesting solutions that will eliminate the causes of the problems in the future.
- Management commitment to implementing good ideas from retrospective meetings, so that improvements are not just discussed, they are made.

A test manager may participate in project-wide retrospectives. There’s also value in having a test retrospective with the test team alone focusing on those things the test team can control and improve.

10. References

10.1 ISTQB Documents

Identifier	**Reference**
[ISTQB-ALTM]	ISTQB Certified Tester, Advanced Level Syllabus, Test Manager, Version 2007, available from [ISTQB-Web]
[ISTQB-EL-Modules]	ISTQB Expert Level Modules Overview, Version 1.0, March 11th 2011
[ISTQB-EL-CEP]	ISTQB Certified Tester Expert Level, Certification Extension Process, Version 1.0, June 17th 2008. Available from [ISTQB-Web]
[ISTQB-Glossary]	ISTQB Glossary of Terms Used in Software Testing, Version 2.1, available from [ISTQB-Web]

10.2 Trademarks

The following registered trademarks and service marks are used in this document:

ISTQB®, MBTI®)

ISTQB® is a registered trademark of the International Software Testing Qualifications Board

Myers-Briggs Type Indicator® (MBTI®) is a registered trademark of the Myers & Briggs Foundation

10.3 Books

Identifier	**Book Reference**
[Black03]	Black, Rex, Critical Testing Processes: Plan, Prepare, Perform, Perfect. Addison-Wesley, 2003.
[Black09]	Rex Black, Managing the Testing Process, Wiley, 2009, ISBN: 0-470-40415-9.
[Boehm03]	Boehm, Barry, and Richard Turner, Balancing Agility and Discipline: A Guide for the Perplexed, Addison-Wesley, 2003.
[Brooks95]	Brooks, Fred, The Mythical Man-Month: Essays on Software Engineering, 2e. Addison-Wesley, 1995.
[Craig02]	Craig, Rick David and Stefan P. Jaskiel, Systematic Software Testing, Artech House, 2002, ISBN: 1-580-53508-9.
[DeMarco99]	DeMarco, Tom, and Timothy Lister, Peopleware: Productive Projects and Teams, 2e. Dorset House, 1999.
[DeMarco03]	DeMarco, Tom, and Timothy Lister, Waltzing With Bears: Managing Risk on Software Projects. Dorset House, 2003.
[Derby&Larsen 06]	Derby, Esther and Diany Larsen, Agile Retrospectives - Making Good Teams Great, The Pragmatic Bookshelf, 2006, ISBN 0-9776166-4-9.
[Drucker06]	Drucker, Peter, The Practice of Management, Harper, 2006.
[Ensworth01]	Ensworth, Patricia, The Accidental Project Manager: Surviving the Transition from Techie to Manager, Wiley, 2001.
[Evans04]	Evans, Isabel, Achieving Software Quality through Teamwork, Artech House, 2004, ISBN: 978-1580536622.
[Galen04]	Galen, Robert, Software Endgames: Eliminating Defects, Controlling Change, And The Countdown To On-time Delivery, Dorset House, 2004.
[Glass02]	Glass, Robert, Facts and Fallacies of Software Engineering, Addison-Wesley, 2002.
[Graham99]	Graham, Dorothy and Mark Fewster, Software Test Automation, Addison-Wesley, 1999.
[Ishikawa91]	What Is Total Quality Control? The Japanese Way, Prentice Hall Business Classics, ISBN:013952441X
[Jones03]	Jones, Capers, Assessment and Control of Software Risks, Prentice Hall, 1993.
[Jones07]	Jones, Capers, Estimating Software Costs: Bringing Realism to Estimating, McGraw-Hill, 2007.
[Jones08]	Jones, Capers, Applied Software Measurement: Global Analysis of Productivity and Quality, 3e, McGraw-Hill Osborne Media, 2008.
[Kan02]	Kan, Stephen, Metrics and Models in Software Quality Engineering, 2e, Addison-Wesley, 2002.
[Leveson95]	Leveson, Nancy, Safeware: System Safety and Computers, Addison-Wesley, 1995.
[McConnell97]	McConnell, Steve, Software Project Survival Guide, Microsoft Press, 1997.
[McConnell99]	McConnell, Steve, After the Gold Rush: Creating a True Profession of Software Engineering, Microsoft Press, 1999.

[McKay 07]	McKay, Judy, Managing the Test People, Rocky Nook, 2007, ISBN: 1-93395-212-1.
[Myers&Briggs 95]	Myers, Isabel Briggs (1980). Understanding Personality Type. Davies-Black Publishing, reprint edition, 1995, ISBN: 0-89106-074-X.
[PMI08]	Project Management Institute, A Guide to the Project Management Body of Knowledge (PMBOK® Guide), 4e, Project Management Institute, 2008.
[Riley 09]	Riley, Tim and Adam Goucher, Beautiful Testing, O'Reilly, 2009, ISBN: 0-596-15981-1.
[Rothman04]	Rothman, Johanna, Hiring The Best Knowledge Workers, Techies and Nerds: The Secrets & Science Of Hiring Technical People, Dorset House, 2004.
[Rothman07]	Rothman, Johanna, Manage It!: Your Guide to Modern, Pragmatic Project Management, Pragmatic Bookshelf, 2007.
[Utting06]	Utting, Mark and Bruno Legeard, Practical Model-Based Testing: A Tools Approach, Morgan Kaufmann, 2006.
[Yourdon03]	Yourdon, Edward, Death March, 2e. Prentice Hall, 2003.
[Whittaker02]	Whittaker, James, How to Break Software: A Practical Guide to Testing, Addison Wesley, 2002.
[Wiegers96]	Wiegers, Karl, Creating a Software Engineering Culture, Dorset House, 1996.

10.4 Papers and Articles

Identifier	**Paper / article reference**
[Copeland Paper 01]	Lee Copeland, "When helping doesn't help", SQE Magazine, January 2001

10.5 Web References

Even though these references were checked at the time of publication of this Expert Level syllabus, the ISTQB cannot be held responsible if references are no longer available.

Identifier	**Web Reference**	**Link**
[Agile Manifesto-Web]	Website of the Agile Manifesto	www.agilemanifesto.org
[ISTQB-Web]	Web site of the International Software Testing Qualifications Board. Refer to this website for the latest ISTQB Glossary and syllabi.	www.istqb.org.
[P&C-Web]	Roles in agile projects	http://en.wikipedia.org/wiki/The_Chicken_and_the_Pig.

11. Appendix A – Notice to Training Providers

11.1 Training Times

Each chapter in the syllabus is assigned an allocated time in minutes. The purpose of this is both to give guidance on the relative proportion of time to be allocated to each section of an accredited course, and to give an approximate minimum time for the teaching of each section.

Training providers may spend more time than is indicated and candidates may spend more time again in reading and research. A course curriculum does not have to follow the same order as the syllabus. It is not required to conduct the course in one continuous block of time.

The table below provides a guideline for teaching and exercise times for each chapter and shows separately the timing for exercises which may be conducted in the workplace (all times are shown in minutes). Note that exercises in the workplace may also be conducted as part of the course given by the training provider (see below).

Nr.	**Chapter**	**Course teaching and exercises**	**Exercises in the workplace**	**Total (minutes)**
1	Introduction	60	0	60
2	Test Missions, Policies, Strategies, Goals	285	270	555
3	Managing the Test Team	690	165	855
4	Managing External Relationships	330	0	330
5	Managing Across the Organization	525	255	780
6	Project Management Essentials	255	360	615
7	Test Project Evaluation and Reporting	510	0	510
8	Testing Considerations for Domain and Project Factors	270	0	270
9	Evaluating Effectiveness and Efficiency	195	0	195
	Total	**3120**	**1050**	**4170**

The following table shows the total course times in days, based on an average of seven hours per working day.

Course element	**Days**	**Hours**	**Minutes**
Course teaching and exercises	7	3	0
Exercises in the workplace	2	3	30
Total:	9	6	30

11.2 Guidelines for Practical Exercises in the Workplace

Practical work should be included for all aspects where candidates are expected to apply their knowledge (learning objective of K3 or higher). The lectures and exercises should be based on the learning objectives and the description of the topics in the content of the syllabus.

Certain learning objectives may be achieved by conducting practical exercises in the workplace. The ISTQB web site [ISTQB-Web], Expert Level section, describes the rules which apply to Training Providers for performing exercises in the workplace.

The following table shows the learning objectives which may be covered by practical exercises in the workplace:

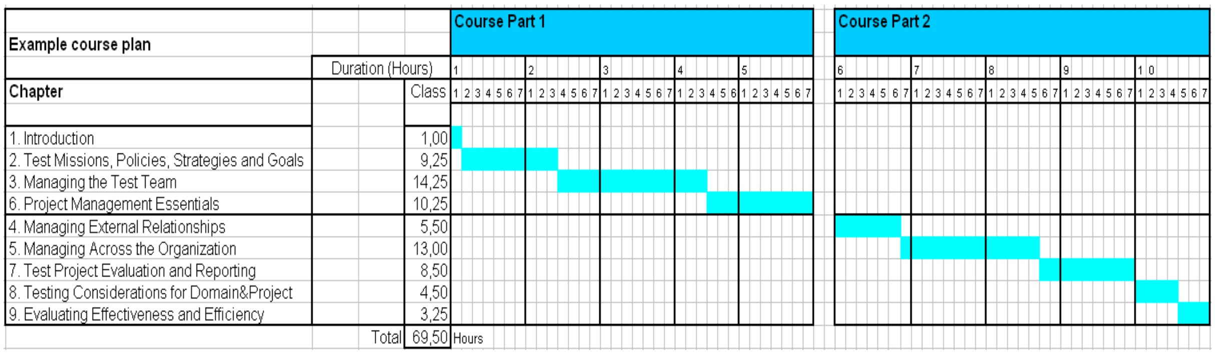
Subject Area	**Relevant Learning Objectives (text in italics indicates adjustments made to the LO for workplace)**
2.2 Mission, Policy, and Metrics of Success	LO 2.2.2 (K6) Define, describe and evaluate test objectives, priorities and test goals that might have long term effects on the software, quality, and dependencies (organizational, economical and technical) *in your organization*.
2.3 Test Strategies	LO 2.3.3 (K6) Create a test strategy document *for your organization* that contains the important elements, enables success factors, and mitigates causes of failure
2.4 Alignment of Test Policy and Test Strategy with the Organization	LO 2.4.1 (K6) Define, describe, evaluate and improve the test policy and strategy(ies) both long and short term for *your* organization and/or a test team *in your organization*, including process and organization of test, people, tools, system and techniques
3.3 Developing the Test Team	LO 3.3.2 (K4) Create a set of S.M.A.R.T goals for an individual tester to determine if the goals are appropriate based on their resume and *your* organization's test strategy
3.4 Leading the Test Team	LO 3.4.1 (K5) Evaluate the information and guidance needed by a test team *in your organization* and determine the most effective method of communication
5.2 Advocating the Test Team	LO 5.2.1 (K4) Define appropriate steps to take to promote or advocate the test team *in your organization*
5.2 Advocating the Test Team	LO 5.2.2 (K6) Define the quantitative and qualitative benefits of testing *in your organization*, and communicate those benefits effectively to project stakeholders
5.3 Placement of the Test Team	LO 5.3.1 (K5) Evaluate *your* organization's structure, its missions, its products, customers, and users, and its priorities, in order to determine the options for proper placement of the test team within the organization, assess the implications of those options, and create an analysis for upper management of those options
6.2 Project Management Tasks	LO 6.2.1(K6) For a given project *in your organization*, estimate the test effort using at least two of the prescribed estimation methods
6.2 Project Management Tasks	LO 6.2.2 (K6) Use historical data from similar projects *in your organization* to create a model for estimating the number of defects that will be discovered, resolved, and delivered on the current project
6.2 Project Management Tasks	LO 6.2.3 (K5) During a project *in your organization*, evaluate current conditions as part of test control to manage, track, and adjust the test effort

Subject Area	**Relevant Learning Objectives (text in italics indicates adjustments made to the LO for workplace)**
	over time, including identifying any deviations from the plan and proposing effective measures to resolve those deviations
6.2 Project Management Tasks	LO 6.2.5 (K6) Using historical information from past projects *in your organization* and priorities communicated by project stakeholders, determine the appropriate trade-offs between quality, schedule, budget, and features available on a project

11.3 Example Course Plans

The following diagrams illustrate possible course plans which may be adopted by Training Providers. These are intended for information only.

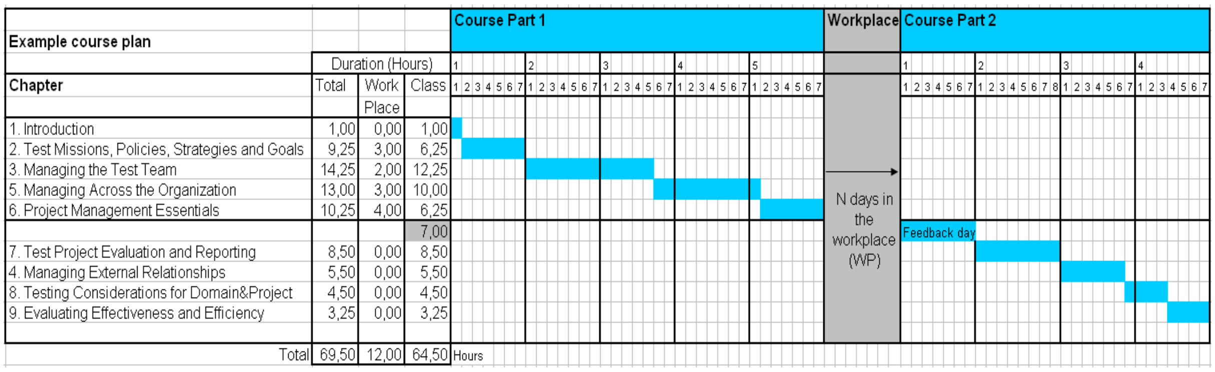
11.3.1 Example Course Plan – without workplace learning objectives



Example course plan	Duration (Hours) Class	Course Part 1	Course Part 2
Chapter		Days 1–5	Days 6–10
1. Introduction	1,00		
2. Test Missions, Policies, Strategies and Goals	9,25		
3. Managing the Test Team	14,25		
6. Project Management Essentials	10,25		
4. Managing External Relationships	5,50		
5. Managing Across the Organization	13,00		
7. Test Project Evaluation and Reporting	8,50		
8. Testing Considerations for Domain&Project	4,50		
9. Evaluating Effectiveness and Efficiency	3,25		
Total	69,50	Hours	

In this example course plan, all learning objectives are taught in class. The plan suggests that the course be split into two parts, each of five days duration. The time spent in class is 10 days (69.5 hours).

11.3.2 Example Course Plan – with workplace learning objectives



Example course plan	Duration (Hours) Total	Work Place	Class
1. Introduction	1,00	0,00	1,00
2. Test Missions, Policies, Strategies and Goals	9,25	3,00	6,25
3. Managing the Test Team	14,25	2,00	12,25
5. Managing Across the Organization	13,00	3,00	10,00
6. Project Management Essentials	10,25	4,00	6,25
			7,00
7. Test Project Evaluation and Reporting	8,50	0,00	8,50
4. Managing External Relationships	5,50	0,00	5,50
8. Testing Considerations for Domain&Project	4,50	0,00	4,50
9. Evaluating Effectiveness and Efficiency	3,25	0,00	3,25
Total	69,50	12,00	64,50

In this example course plan, all learning objectives which may be conducted in the workplace are allocated to a single block of time following course part 1. At the start of course part 2 a whole day (7 hours) is allocated to feedback from the workplace. The total time spent in class is nine days (64.5 hours).

11.4 Rules for e-Learning

The following parts of this syllabus are **not** considered appropriate for implementation as e-Learning.

Section	**Reason**
3.2 Building the test team	Interviewing skills cannot be effectively practiced remotely
3.3 Developing the test team	Conducting performance reviews cannot be effectively practiced remotely

11.5 Standards Used

The syllabus contains references to established standards, which must be used in the preparation of training material. Each standard used must be the version quoted in the current version of this syllabus. Other publications, templates or standards not referenced in this syllabus may also be used and referenced, but will not be examined.

Index