



**INFORMATION SYSTEMS  
EXAMINATIONS BOARD**

**Practitioner Certificate  
in  
Software Test Analysis**

**Syllabus**

**Version 2.0 – January 2008**

# Software Test Analysis

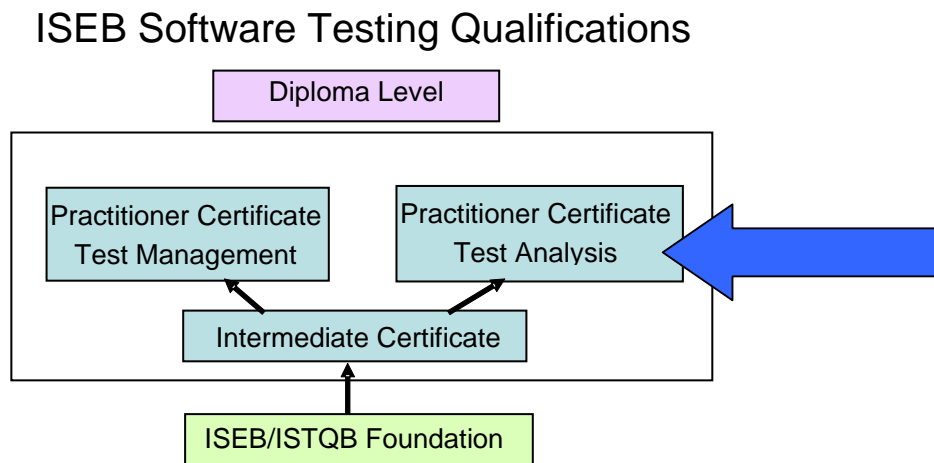
## Background

This document is the syllabus for the ISEB Practitioner Certificate in Software Test Analysis, as administered by the British Computer Society's Information Systems Examination Board (ISEB).

Background information on the first level qualification, the Foundation Certificate, and the ISEB Intermediate Certificate, may be found at: [www.iseb-exams.com/st](http://www.iseb-exams.com/st).

A Testing Practitioner is anyone involved in Software Testing. This includes testers, test analysts, test engineers, test consultants, test managers, user acceptance testers, and software developers.

This syllabus extends the content of the ISEB/ISTQB Foundation Certificate syllabus in Software Testing, and builds directly on the topics in the areas of Test Analysis covered in the ISEB Intermediate Certificate in Software Testing syllabus.



## Entry Criteria

The entry criteria for candidates wishing to take the ISEB Practitioner Certificate in Software Test Analysis examination are:

- hold the ISEB Intermediate Certificate in Software Testing

AND EITHER

- have at least 18 months experience in software testing

OR

- have completed an ISEB accredited training course for the Practitioner Certificate in Software Test Analysis

BUT preferably have all three of the above.

## The Test Analysis Examination

The examination will consist of 6 optional questions. Candidates will be required to choose 4, with each question being worth 25% of the overall mark.

Candidates will be allowed 2 hours for the examination (with extra time allowed for those for whom English is not their first language).

To obtain a pass the candidate must achieve 60% or more, to achieve a distinction 80% or more.

The ISEB Practitioner Certificate in Test Analysis examination will be based on the syllabus in this document. Examination questions will be drawn from all topics in the syllabus, and coverage of any given topic can be expected to be in proportion to the amount of time allocated to that topic in the syllabus.

Answers to examination questions may require the use of material based on more than one section of this syllabus and where necessary supported by information from the required supporting syllabuses. All sections of the syllabus are examinable.

No section is intended to be independent, or will be examined independently - questions will typically ask candidates to demonstrate their understanding of the interactions between these subjects. A good standard of knowledge from the ISEB Intermediate Certificate in Software Testing syllabus will be required to succeed in this module.

## Notice to Training Providers

Each major subject heading in the syllabus is assigned an allocated time. The purpose of this is to give both guidance on the relative proportion of time to be allocated to each section of an accredited course and an approximate minimum time for the teaching of each section. Course providers may spend more time than is indicated and candidates may spend more time again in reading and research.

The total time specified in this syllabus is 18 hours of lecture and practical work.

The course may be delivered as a series of modules with gaps between them, as long as it meets all other constraints. Courses do not have to follow the same order as the syllabus.

The syllabus contains references to established standards. The use of referenced standards in the preparation of training material is mandatory. Each standard used must be the version quoted in the current version of this syllabus.

## Terminology Used

Terminology used in this document is from the current version of the ISTQB Glossary of Testing Terms or from BS 7925-1, BS 7925-2, IEEE Std. 829-1998, IEEE Std. 610-1990 or other referenced source as appropriate to the specialist topic. Standards override the glossary in cases of conflict. This syllabus may override both standards and glossary. Versions of standards and glossary change from time to time; the version current at the time of publication of this syllabus is the version referred to in this syllabus as the latest version.

## Other Syllabuses

Any references to other ISEB syllabuses refer to the latest published version at the date of issue of this syllabus.

## Bloom's Taxonomy

Learning objectives in this syllabus are given indicators from K1-K6. These are based on Bloom's taxonomy of knowledge in the cognitive domain (ref *Taxonomy of Educational Objectives, Handbook 1 – The Cognitive Domain*, Bloom et al., New York 1956), and can be broadly interpreted as follows: K1 – Recall; K2 – Comprehension; K3 – Application; K4 – Analysis; K5 – Synthesis; K6 – Evaluation. Bloom's taxonomy is explained in greater detail in Appendix A, where examples are given. All topics in this syllabus have learning objectives associated with them, each of which has an associated K level. The language used in this syllabus mirrors as closely as possible the language used in defining Bloom's taxonomy to provide candidates with consistent pointers to the expected level of knowledge and a consistent way of expressing that level in words.

## Structure of this Document

This syllabus is structured into sections relating to major subject headings and numbered with a single digit section number. Each section is allocated a minimum contact time for presentation. Learning objectives are identified at the beginning of each section. The K level for each learning objective is identified at the lowest level of breakdown in the learning objectives list.

The breakdown of content matches the structure of the learning objectives, so that the material related to a given learning objective appears in a paragraph bearing the same numerical reference as that of the related learning objective. The content associated with each learning objective may include non-examinable explanatory commentary in italics as well as the examinable content associated with the learning objective.

## Change History

| Version 2.0 – January 2008 |  |
|----------------------------|--|
| 1.1                        | Influence of context – timing changed from 2 hours to 1 hour. Learning objectives amended to reflect the reduced time available  |
| 1.2                        | Analysis of Test basis – timing changed from 2 hours to 1 hour. Learning objectives amended to reflect the reduced time available  |
| 1.3                        | Supporting Activities – timing changed from 2 hours to 1.5 hours. Learning objectives amended to reflect the reduced time available  |
| 2.1                        | Testing in Action – timing changed from 3 hours to 1 hour. Learning objectives amended to reflect the reduced time available – in particular the detail about the “advisory role of Test Analyst” has been removed   |
| 2.2.1                      | Test Techniques – timing changed from 7 hours to 12 hours, to reflect the emphasis on the number and specific detail of the techniques where examination candidates are expected to have practical knowledge. It would not be possible to teach to the required depth in the previously allotted 7 hours |
| 2.2.2                      | Test Types – timing changed from 2 hours to 1 hour. Learning objectives amended to reflect the reduced time available  |

# Syllabus for Practitioner Certificate in Test Analysis

## Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Choices (3.5 hours)</b> .....  | <b>6</b>  |
| 1.1      | Influence of Context (1 hour) .....                                       | 7         |
| 1.2      | Analysis of Test Basis (1 hour).....                                      | 8         |
| 1.3      | Supporting Activities (1.5 hours).....                                    | 8         |
| <b>2</b> | <b>Techniques (14.5 hours)</b> .....                                      | <b>9</b>  |
| 2.1      | Testing in Action (1.5 hours).....  | 10        |
| 2.2      | Use of Test Techniques and Test Types (13 hours) .....                    | 12        |
|          | <b>Appendix A: Levels of Knowledge</b> .....                              | <b>17</b> |
|          | <b>Appendix B: Rules for ISEB SWT Practitioner Level Syllabuses</b> ..... | <b>19</b> |
|          | <b>Appendix C: Syllabus References</b> .....                              | <b>20</b> |

# 1 Choices (3.5 hours)

*Test Analysts take decisions and make recommendations about what testing to do, and which test cases to do first. While these decisions may be driven primarily by an analysis of the test basis, candidates should be able to place their analysis in a broader context when justifying their decisions.*

## Learning Objectives

### 1.1 Influence of Context

- i. Select test approaches and techniques (K3)
- ii. Prioritise the tasks resulting from chosen approaches and techniques (K4)
- iii. Justify decisions made (K6)

### 1.2 Analysis of Test Basis

- i. Analyse a test basis to identify test items (K4)
- ii. Select and apply appropriate test techniques (K3)

### 1.3 Supporting Activities

- i. Describe how tools are used to support testing activities (K2)
- ii. Select appropriate tools (K3)
- iii. Describe the use of a defect taxonomy (K2)
- iv. Analyse and interpret incident metrics (K4)
- v. Write an incident report (K5)

# 1 Choices (continued...)

## 1.1 Influence of Context (1 hour)

*This section extends section 5.1 of the Intermediate Certificate syllabus and section 4.6 of the Foundation Certificate syllabus.*

In this syllabus the term 'context' means some combination of the following:

### Requirements

- Business/customer requirements
- Contractual requirements
- Technological / implementation constraints
- Regulatory standards

### Project

- Project strategy
- Test activities already completed on the project
- Test activities currently in progress on the project
- Level of testing being considered
- Project risks
- Available team experience, tools and time available to the project

### Product and relevant technology

- Application domain (web-based, client-server, mainframe, PC-based)
- Product risks
- Design and implementation details specific to product

For a given context:

- Select test approach and test technique.**
- Prioritise tasks related to test approaches and test techniques.**
- Justify decisions made in choosing test approach, test techniques and prioritising tasks.**

# 1 Choices (continued...)

## 1.2 Analysis of Test Basis (1 hour)

*This section extends section 5.1 of the Intermediate Certificate syllabus. A test basis may consist of (any or all of) requirements, business scenarios, state diagrams and use cases. Candidates are expected to be able to use these artefacts in combination. Where no test basis is documented it may be necessary to construct a document from available information to act as a test basis.*

- i. **Analyse a test basis to identify test items.**
- ii. **Use the test basis analysis to make a reasoned choice and application of test techniques, for the purpose of deriving executable test cases.**

## 1.3 Supporting Activities (1.5 hours)

*Activities such as incident management, defect logging, test estimation and participation in reviews are covered in supporting syllabuses. This section extends sections 4.2 and 4.5 of the Intermediate Certificate syllabus. A defect taxonomy is a classified list of known problems. Although defect taxonomies may be built by projects or organisations, candidates are expected to be familiar with a known, publicly-available taxonomy. Two such taxonomies are given in the References in Section 3.*

- i. **Tool support.**
  - Describe typical tool support for Static Analysis and Dynamic Analysis.
  - Describe how tools are used in test execution and logging.
  - Describe how tools are used in performance and monitoring.
- ii. **For a given scenario, select an appropriate tool, and describe how that tool would help.**
- iii. **Describe, with examples, how a 'defect taxonomy' may be used in test design.**
- iv. **Interpret graphs showing incidents found over time.**
- v. **Write (or re-write) an incident report.**

## 2 Techniques (14.5 hours)

### Learning Objectives

*The learning objectives listed here relate to the topic as a whole and are provided as an overall guide to the knowledge levels required and to maintain consistency of presentation across Practitioner syllabuses. Within the subsections identified below more detailed learning objectives are given relating to individual areas. The learning objectives embedded within the text are those directly related to the content of that section of the syllabus and achievement of those learning objectives implies achievement of the higher level objectives listed here.*

#### 2.1 Testing in Action

- i. Calculate metrics to identify test coverage and test progress (K3)
- ii. Compile a test summary report (K5)
- iii. Compare and contrast different test approaches (K2)

#### 2.2 Use of Test Techniques and Test Types

*This list of learning objectives provides the overall expected levels of understanding (K levels) of the topics in this section of the syllabus. Individual sub-sections identify the application of the K levels to specific test techniques and test types.*

- i. Recognise, describe and illustrate different test techniques/test types with examples
- ii. Compare and contrast different test design techniques/test types (K2)
- iii. Select test design techniques and test types appropriate to given context (K4)
- iv. Apply given test design techniques/test types to derive test cases (K3)
- v. Analyse test output and suggest new test cases and test approaches (K5)
- vi. Prioritise test activity (K4)
- vii. Justify selection of techniques/types and prioritisation of tests (K6)

## 2 Techniques (continued...)

### 2.1 Testing in Action (1.5 hours)

Note that paragraphs i – iii below have multiple learning objectives defined within them.

#### i. Coverage and metrics:

- Describe the application of the 'subsumes' model (see BS 7925-2 annex C) to testing based on structure of code.
- Recognise and be able to calculate Defect Detection Percentage.
- Understand and calculate coverage for each of the test techniques described below.

#### ii. Analysis of overall test output and preparation of a test summary report.

*Candidates are expected to analyse test results, graphs of coverage over time, timelines of key events, incident reports and observation logs. These artefacts may be given singly or in combination.*

- Analyse information to arrive at a reasoned interpretation of events.
- Write a Test Summary Report.

#### iii. Compare and contrast different test approaches.

##### Static and Dynamic Analysis

*Static analysis is used to find problems by analysing the system as an artefact, rather than its behaviour as a working entity. Static analysis can be performed on code, and on non-code artefacts, such as requirements and state models. Typical tools include compilers, code analysis tools, and HTML validation tools. Static analysis can detect errors in syntax and possible faults such as unreachable code, undeclared variables, parameter type mismatches, uncalled functions and procedures, array boundary violations. Static analysis can produce metrics on the complexity and volume of code.*

*Dynamic analysis provides run-time information on the state of working software, and is supported by tools that monitor the running program and its environment, and may involve instrumenting the program under test. Typical tools include memory, CPU and I/O monitors, tools to monitor memory usage and reveal memory leaks, tools that report on object-code statement / decision coverage for a given set of tests.*

- Recognise complexity metrics and describe typical uses.
- Describe the differences between Static and Dynamic Analysis, and the ways each can be used in the context of a testing project.

## 2 Techniques (continued...)

### Scripted and Unscripted Test Approaches

*Much testing follows a pre-written test procedure specification. To the extent that the test activity follows those instructions, and those instructions limit the test, such testing is scripted. To the extent that the test activity deviates from or expands upon those instructions, it is unscripted. Testing is often a balanced combination of these two complementary approaches.*

*Test techniques such as those given below typically allow a script to be developed before the software is available for testing – and the initial investment may be offset by the speed, accuracy, and depth of coverage that can result. Some scripted tests are written to persist for the life of the product, and may be automated with test execution tools to run frequently and in bulk throughout the life of the product. As such, they are helpful in showing that a system continues to work to specification. Automated unit testing and regression testing use predominantly scripted approaches.*

*Diagnostic and exploratory approaches to testing require greater flexibility and responsiveness, and tests may be constructed incrementally, with the design of the next test frequently dependent on the outcome of the current test. Unscripted testing is assisted by the use of tools which record test activity and system behaviour, and is helpful in producing novel observations and determining the impact or extent of an observed problem. Test types such as performance and reliability testing use scripted test cases in bulk within a broadly unscripted framework of experiment and exploration.*

- Describe ways in which scripted and unscripted approaches can be used in the context of a testing project.

### Goals of testing

*Where the purpose of testing is to show that a system works as expected, the test analyst typically uses scripts to specify acceptable behaviour. Test suites are designed to be repeated, and may be automated, and test cases are designed to produce well-determined output that can be judged simply against a test oracle. Tests may be based on requirements and design documents.*

*Where the purpose of testing is to find unexpected problems and risks, the test analyst typically investigates using knowledge of previous problems of the designed system and potential failures of the underlying technology. Observations are judged using business and technical knowledge.*

*Where the purpose of testing is to diagnose problems and risks, the test analyst typically starts with reports of problems, and designs tests to reproduce the symptoms or the predicted causes of those problems. Further tests incrementally help to refine diagnosis and assess impact.*

- Describe the differences between testing designed to show that software works as expected, testing designed to reveal unexpected problems, and testing designed to diagnose known problems.

## **2 Techniques (continued...)**

### **2.2 Use of Test Techniques and Test Types (13 hours)**

#### **2.2.1 Test Techniques (12 hours)**

*Note that BS 7925-2 contains references to Syntax Testing, Cause-Effect Graphing, and other techniques not mentioned explicitly in this section.*

Describe and recognise the applicability of all test techniques from BS 7925-2 (including those not explicitly listed in this section); compare and contrast the techniques in given contexts.

The following learning objectives apply to all test techniques listed in this sub-section:

- i. Recognise, describe and illustrate with examples.**
- ii. Compare and contrast within a given context.**
- iii. Demonstrate reasoned judgement in selection of test techniques and justify selections.**
- iv. Apply techniques singly and in combination to derive test cases, with associated preconditions and expected results as necessary.**
- v. Analyse test output and exit criteria to find out the effectiveness of techniques leading to that output. Make appropriate adjustments to test techniques and test cases to achieve desired outcomes. Suggest possible alternative approaches.**
- vi. Prioritise test cases in a given context.**

## 2 Techniques (continued...)

### **Specification-based (Black-box) Testing**

#### **Equivalence Partitioning**

*Within an equivalence partition, a single value is selected to act as a proxy for a range of equivalent values – and a set of such values acts as a proxy for testing all possible values of a variable. These sets of proxies may be combined when designing test cases for multiple, independent variables. Equivalence partitions may also be considered in relation to multiple, dependent variables.*

#### **Boundary Value Analysis**

*Boundary analysis may also be considered in relation to multiple variables – note that where the variables are dependent, the boundary will be tilted or non-linear.*

*For information on domain analysis and testing, see Beizer's Black Box Testing, Ch 7.*

- Apply Equivalence Partitioning to situations with up to 2 dependent or independent variables.
- Apply Boundary Value Analysis to situations with up to 2 dependent or independent variables.

*For the purposes of this syllabus boundary values may be specified using either 2 values (as in the Foundation Certificate syllabus) or 3 values (as in BS 7925-2); in all cases the convention in use will be clear from the context or explicitly stated.*

#### **State Transition Testing**

*State diagrams and state tables are models of parts of a system. State models may be nested (i.e. a secondary state model may be invoked from a single state primary state model), or may operate in parallel (i.e. to study the interaction between two independent but simultaneous models). Path-testing techniques can be applied to state diagrams and state tables.*

- Analyse and construct state diagrams and state tables.
- Derive test cases from state diagrams and state tables.
- Apply Chow's n-Switch coverage measures.

#### **Decision Table Testing**

*In a decision table, conditions and actions may be expressed as true or false at their intersections with rules. Conditions can also be expressed as ranges, and actions as values or formulae. Some rules may be infeasible where conditions are dependent and exclusive.*

- Analyse and construct decision tables from specifications.
- Derive test cases from decision tables, including selecting actual values for input data and expected output.

## 2 Techniques (continued...)

### **Use Case Testing**

*A use case has various paths through it; each path represents a scenario. There is typically one normal scenario, and a selection of alternate or error scenarios.*

*Note: Use case testing is not found in BS 7925-2. A suitable reference can be found at Appendix C. This section follows on from section 4.3.5 in the Foundation syllabus.*

- Analyse user scenarios from textual and graphical use cases.
- Identify scenarios corresponding to normal and alternate/error paths through a use case.
- Derive test cases from scenarios with appropriate choice of data.

### **Classification Tree Method**

*Classification trees are often used with information derived from domain analysis. A suitable reference can be found at Appendix C.*

- Analyse and construct classification trees for identified test objects.
- Derive test cases from a classification tree, including choice of data and prediction of expected output. The classification tree may include sub-classes. Test cases may include those that trigger error-handling.

### **Structure-based (white-box) Testing**

*Structure-based test design techniques apply to object code – but because testers rarely encounter object code, its application in this syllabus is to intermediary/representational artefacts such as control flow diagrams, flow charts and pseudo-code. The concepts may be extended to state transition diagrams, lifecycles and data flow diagrams.*

#### **Statement Testing**

- Apply statement testing to artefacts with loops and/or nested conditions

#### **Decision Testing**

- Apply decision testing to artefacts with loops and/or nested conditions

## 2 Techniques (continued...)

### 2.2.2 Test Types (1 hour)

The following learning objectives apply to all test types listed in this sub-section:

- i. **Recognise, describe and illustrate the given test types with examples.**
- ii. **Compare and contrast the given test types within a given context.**
- iii. **Prioritise test activity as guided by strategy.**
- iv. **Demonstrate reasoned judgement in selection of test type and justify selections.**

#### **Reliability Testing**

*Reliability testing determines the ability of a software product to perform its required functions under stated conditions for a specified period of time, or number of operations. Reliability requirements typically have a level of tolerance to go with the specified time/number. Candidates are expected to recognise that multiple measurements are needed, typically leading to a calculation of mean time between failure, and that there are different ways that this can be calculated.*

*Reliability testing tends to use test execution tools to exert the 'stated conditions', and monitoring tools to observe failures or otherwise unacceptable behaviours. Some reliability testing may use random input to stress the software (fuzz testing) and reveal problems that might occur under adverse conditions.*

*Note that candidates should be able to calculate mean time between failures, but candidates are not expected to directly calculate errors, probabilities or statistical measures, nor to compare against binomial distributions.*

- Measure and extrapolate from measures of mean time between failures (MTBF).
- Use random input in constructing tests.

#### **Usability Testing**

*Usability testing determines the extent to which the software product is understood, easy to learn, easy to operate and attractive to the users under specified conditions. Testers can assess usability in a variety of ways - by using the product themselves, by observing others using the product, and by using questionnaires given to users after exposure to the product.*

*An example of a usability issue could be slow response times, and examples of context could include the impact of long or unreliable response times on electronic braking or on access to a web page.*

*Usability is a measurement from the user's point of view, and different users will have different experiences. Accessibility standards typically give guidance to improve usability for those with physical handicap, and may form a regulatory requirement against which to test.*

- Describe and compare approaches to usability assessments.
- Demonstrate familiarity with known accessibility standards.

## 2 Techniques (continued...)

### **Testing Under Load**

*Volume, performance and stress testing all require load to be applied to a system or component, but differ in the way that load is applied, and in the measurements that are made. Volume testing measures system behaviour under high load, performance testing measures the behaviour of a system under a variety of loads (load profile) in an attempt to model the system, and stress testing takes a system beyond its normal operating limits, typically to observe failures.*

*Load on a system is typically generated by a performance testing tool within a designed scenario, which simulates or generates numbers of transactions, of processes, of users, volume of data etc. The system's resources – memory, CPUs, licenses, bandwidth – may be constrained purposefully, or as a result of test environment limitations. A load profile describes the combination of load that will be applied for a given test, and how it will vary over time.*

*Testing under load typically makes many measurements of different types of response times, which may be averaged to give an overall value, or displayed as a scatter graph to show populations, trends and emergent behaviours. Measurements may be interpreted with graphs of memory, CPU or network usage to give context – such measurements are often made with a monitoring tool.*

- Select load profile to fit different goals of volume, performance, and stress tests.

# Appendix A: Levels of Knowledge

The following levels of knowledge are defined as applying to this syllabus. Each topic in the syllabus will be examined according to the learning objectives defined elaborated in the section devoted to that topic. Each learning objective has a level of knowledge (K level) associated with it and this K level defines the nature of any examination questions related to that topic.

Note that each K level subsumes lower levels. For example, a K4 level topic is one for which a candidate must be able to analyse a situation and extract relevant information. A question on a K4 topic could be at any level up to and including K4. As an example, a scenario requiring a candidate to analyse a scenario and select the best risk identification method would be at K4, but questions could also be asked about this topic at K3 and a question at K3 for this topic might require a candidate to apply one of the risk identification methods to a situation.

## Level 1: Remember (K1)

The candidate will recognise, remember and recall a term or concept. All topics in this syllabus require K1 level of understanding.

### Example

Can recognise the definition of “failure” as:

- “non-delivery of service to an end user or any other stakeholder” or
- “actual deviation of the component or system from its expected delivery, service or result”.

## Level 2: Understand (K2)

The candidate can select the reasons or explanations for statements related to the topic, and can summarise, compare, classify and give examples for the testing concept. All topics in this syllabus require K2 level of understanding.

### Examples

Can explain the reason why tests should be designed as early as possible:

- To find defects when they are cheaper to remove.
- To find the most important defects first.

Can explain the similarities and differences between equivalence partitioning (EP) and classification tree method (CTM):

- Similarities: both partition inputs to reduce the required number of test cases to achieve good coverage.
- Differences: EP partitions both input and output while CTM partition only inputs.

## Level 3: Apply (K3)

The candidate can select the correct application of a concept or technique and apply it to a given context. No topics in this syllabus are at the K3 level but those at K4 level and above include a K3 level of understanding.

### Examples

- Can identify boundary values for valid and invalid partitions.
- Can select test cases from a given state transition diagram in order to cover all transitions.

#### **Level 4: Analyse (K4)**

The candidate can separate information related to a concept or technique into its constituent parts for better understanding, and can distinguish between facts and inferences. Some topics in this syllabus require a K4 level of understanding.

##### Examples

- Can identify inconsistencies in a test basis.
- Can select test design techniques for a given context.

#### **Level 5: Synthesise (K5)**

The candidate can identify and build patterns in facts and information related to a concept or technique, and can create new meaning or structure from parts of a concept. Some topics in this syllabus require a K5 level of understanding.

##### Examples

- Can analyse test results and identify further tests to be applied.
- Can identify potential problems from system documentation.

#### **Level 6: Evaluate (K6)**

The candidate can judge the value of information and decide on its applicability in a given situation. Some topics in this syllabus require a K6 level of understanding.

##### Examples

- Can determine the relative effectiveness and efficiency of different testing techniques.
- Can determine the type of information that should be gathered for an incident report.

#### **References**

(For the cognitive levels of learning objectives)

Bloom, B. S. (1956). *Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain*, David McKay, Co. Inc.

Anderson, L. W. and Krathwohl, D. R. (eds) (2001). *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*, Allyn & Bacon.

# Appendix B: Rules for ISEB SWT Practitioner Level Syllabuses

## Syllabus General Rules

SG1. The syllabus should be understandable and absorbable by people who hold the ISEB/ISTQB Foundation Certificate.

SG2. The syllabus should be more practical than theoretical.

SG3. The syllabus should be clear and unambiguous to its intended readers.

## Syllabus Content

SC1. The syllabus should include recent testing concepts and should reflect current best practice in software testing where this is generally agreed and has been published. The syllabus is subject to review every three to five years.

SC2. The syllabus should minimise time-related issues, such as current market conditions, to enable it to have a shelf life of three to five years.

SC3. Each statement in the Syllabus should clearly state what the candidate is expected to know and therefore what can be examined. (E.g. "Explain that the criteria are" not "Explain the criteria for")

## Learning Objectives

LO1. Learning objectives should distinguish between applicable knowledge levels according to Bloom's taxonomy (K1 to K6).

LO2. The description of the content should be consistent with the learning objectives.

LO3. To illustrate the learning objectives, sample exam questions for each major section should be issued along with the syllabus.

LO4. Each section of the syllabus should include a K-level

## Structure rules

ST1. The structure of the syllabus should be clear and allow cross-referencing to and from other parts, from exam questions and from other relevant documents.

ST2. Overlap between sections of the syllabus should be minimised. Overlap between related syllabuses (Foundation, Intermediate, Analyst & Management) should be minimised, or stated (if intentional)

ST3. Each syllabus and each section of each syllabus should have the same structure and format.

ST4. The syllabus should contain version, date of issue and page number on every page.

ST5. The syllabus should include a guideline for the amount of time to be spent in each major section (to reflect the relative importance of each topic).

ST6. Each statement should be consistent with the ISTQB Foundation Syllabus (where it covers the same area) and should use the same terminology.

## Syllabus References

SR1. Sources and references should be given for concepts in the syllabus to help training providers find out more information about the topic.

SR2. Where there are not readily identified and clear sources, more detail should be provided in the syllabus. For example, definitions are in the Glossary, so only the terms are listed in the syllabus.

# Appendix C: Syllabus References

## Standards and Syllabuses

This syllabus makes reference to the following syllabuses:

- [ISTQB F Syllabus] ISTQB Certified Tester: Foundation Level Syllabus (Version 2007)
- ISEB Intermediate Certificate in Software Testing.

This syllabus makes reference to the following standards:

- [ISTQB Glossary] ISTQB - Veenendaal, Erik van (ed.) (2006), Standard glossary of terms used in Software Testing, Version 2.0
- [BS 7925-2] BS 7925-2:1998, *Software Testing Part 2: Software Component Testing*
- [IEEE 829] IEEE Std 829™ (1998/2005) IEEE Standard for Software Test Documentation (currently under revision)

## Books and Web Sites

*Note that a standard will take precedence over a book. Where common practice differs from standard, candidates will not be penalised for using a standard approach. Nevertheless, candidates should show awareness of the differences from standard.*

Standards may not give an effective view of techniques in use. The following books give worked examples of a range of techniques.

- Beizer, *Black-Box Testing: Techniques for Functional Testing of Software and Systems*, Wiley 1995
- Copeland, *A Practitioner's Guide to Software Test Design*, Artech House 2003;

Use Case testing and the Classification Tree Method are not detailed in BS 7925-2. The following articles are easily available and provide a good starting point for each technique:

- Use Case testing – Heumann *Generating Test Cases From Use Cases*, Rational Edge June 01: <http://www-128.ibm.com/developerworks/rational/library/content/RationalEdge/jun01/GeneratingTestCasesFromUseCasesJune01.pdf>
- Classification tree testing – Grochtmann *Test Case Design Using Classification Trees*, STAR'94: <http://www.systematic-testing.com/documents/star1994.pdf>

A treatment of Defect Detection Percentage may be found in:

- Fewster & Graham *Software Test Automation: Effective Use of Test Execution Tools*, Addison-Wesley (§8.4.1.1)

An example of an accessibility standard is:

- W3C's priority 1 checkpoints provide a minimum, standardised, comprehensible set of accessibility checkpoints for internet browsing. <http://www.w3.org/TR/WAI-WEBCONTENT/full-checklist.html>

Publicly-available defect taxonomies include:

- Kaner, Falk, Nguyen's *Testing Computer Software* (the comprehensive list at the back)
- Beizer/Vinter's taxonomy available from: <http://inet.uni2.dk/~vinter/bugtaxst.doc>