Systems Development Syllabus

Version 1.1

January 2010
## Change History

<table>
<thead>
<tr>
<th>Version 1.1.</th>
<th>Re-branded and re-formatted. Added in a contents page, change history and the examination format on final page.</th>
</tr>
</thead>
</table>
Foundation Certificate in Systems Development

Contents

1. Introduction to Systems Development (5%) ................................................................. 5
   1.1 What is systems development 5
   1.2 The scope of systems development 5
   1.3 Relationship with other disciplines such as project management, programming, testing, service management, change and configuration management 5

2. Life Cycle types and their Rationales (10%) .................................................................... 5
   2.1 Systems Development Lifecycles 5
   2.2 For each lifecycle 5
   2.3 Adaptation and customisation of the lifecycles 5
   2.4 Project management and the lifecycles 5

3. Business Analysis (5%) .................................................................................................. 6
   3.1 Definition of Business Analysis 6
   3.2 Place of Business Analysis in the development lifecycle 6
   3.3 Outcomes from Business Analysis 6

4. Requirements Engineering (10%) .................................................................................. 6
   4.1 Key areas of Requirements Engineering 6
   4.2 Techniques for requirements elicitation 6
   4.3 For each elicitation technique 6
   4.4 Types of requirements 6
   4.5 Prioritising requirements 6
   4.6 Managing requirements 7
   4.7 Analysing and validating requirements 7

5. Making a Business Case (10%) .................................................................................... 7
   5.1 Feasibility checking 7
   5.2 Elements of a business case 7
   5.3 Identifying, evaluating and selecting options 7
   5.4 Principles of cost/benefit analysis 7
   5.5 Principles of impact and risk analysis 7

6. Programming and Development Approaches (10%) ...................................................... 7
   6.1 Types of development approaches (these are not necessarily mutually-exclusive) 7

7. Systems Modelling and Specification Techniques (10%) .............................................. 8
   7.1 Reasons for modelling 8
   7.2 Modelling from different perspectives 8
   7.3 Cross-referencing different modelling perspectives 8
   7.4 Documentation and specification 8

8. Systems Design (10%) ................................................................................................ 8
   8.1 The location of systems design in the systems development lifecycle 8
   8.2 The objectives and constraints of systems design 8
   8.3 Input design, input technologies and their application 8
   8.4 Output technologies and their application (screen and form design is covered in the interface design and usability section of the syllabus) 9
   8.5 The objectives and principles of process design 9
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.6</td>
<td>The objectives and principles of data design</td>
<td>9</td>
</tr>
<tr>
<td>8.7</td>
<td>The design of codes</td>
<td>9</td>
</tr>
<tr>
<td>8.8</td>
<td>The scope and principles of security design</td>
<td>9</td>
</tr>
<tr>
<td>9.</td>
<td>System Architecture (5%)</td>
<td>9</td>
</tr>
<tr>
<td>9.1</td>
<td>Types of architecture</td>
<td>9</td>
</tr>
<tr>
<td>9.2</td>
<td>Objectives and principles of systems architectures</td>
<td>9</td>
</tr>
<tr>
<td>9.3</td>
<td>Stakeholders and roles in architecture</td>
<td>10</td>
</tr>
<tr>
<td>9.4</td>
<td>Management of the architecture</td>
<td>10</td>
</tr>
<tr>
<td>9.5</td>
<td>The tiered architecture approach to IT system development</td>
<td>10</td>
</tr>
<tr>
<td>9.6</td>
<td>Service Oriented Architecture and Service Oriented Development Applications</td>
<td>10</td>
</tr>
<tr>
<td>10.</td>
<td>Quality and Testing (10%)</td>
<td>10</td>
</tr>
<tr>
<td>10.1</td>
<td>The definition of software quality</td>
<td>10</td>
</tr>
<tr>
<td>10.2</td>
<td>The objectives and limitations of testing</td>
<td>10</td>
</tr>
<tr>
<td>10.3</td>
<td>The structure and purpose of the static test stages of the V model</td>
<td>10</td>
</tr>
<tr>
<td>10.4</td>
<td>The purpose and content of the dynamic test stages of the V model</td>
<td>10</td>
</tr>
<tr>
<td>10.5</td>
<td>Static testing</td>
<td>10</td>
</tr>
<tr>
<td>10.6</td>
<td>Dynamic testing</td>
<td>11</td>
</tr>
<tr>
<td>10.7</td>
<td>Re-testing (confirmation testing)</td>
<td>11</td>
</tr>
<tr>
<td>10.8</td>
<td>Regression testing</td>
<td>11</td>
</tr>
<tr>
<td>11.</td>
<td>Implementation and changeover (5%)</td>
<td>11</td>
</tr>
<tr>
<td>11.1</td>
<td>The task of file and data conversion</td>
<td>11</td>
</tr>
<tr>
<td>11.2</td>
<td>The principles and problems of data mapping</td>
<td>11</td>
</tr>
<tr>
<td>11.3</td>
<td>Plan, test and undertake data conversion</td>
<td>11</td>
</tr>
<tr>
<td>11.4</td>
<td>The role of supporting documentation, including user manuals</td>
<td>11</td>
</tr>
<tr>
<td>11.5</td>
<td>Approaches to training</td>
<td>11</td>
</tr>
<tr>
<td>11.6</td>
<td>Define training needs and evaluate training effectiveness</td>
<td>11</td>
</tr>
<tr>
<td>11.7</td>
<td>Systems implementation</td>
<td>11</td>
</tr>
<tr>
<td>12.</td>
<td>Evaluation and maintenance (5%)</td>
<td>12</td>
</tr>
<tr>
<td>12.1</td>
<td>The location of maintenance in the systems development life cycle</td>
<td>12</td>
</tr>
<tr>
<td>12.2</td>
<td>The range of metrics which might be used to evaluate a delivered software product</td>
<td>12</td>
</tr>
<tr>
<td>12.3</td>
<td>The purpose and conduct of a post-implementation review</td>
<td>12</td>
</tr>
<tr>
<td>12.4</td>
<td>The purpose and conduct of a post-project review</td>
<td>12</td>
</tr>
<tr>
<td>12.5</td>
<td>The distinction between corrective, adaptive and perfective maintenance</td>
<td>12</td>
</tr>
<tr>
<td>13.</td>
<td>Software Support Tools (5%)</td>
<td>13</td>
</tr>
<tr>
<td>13.1</td>
<td>Computer Aided Software Engineering tools (CASE)</td>
<td>13</td>
</tr>
<tr>
<td>13.2</td>
<td>Computer Aided Software Testing tools (CAST)</td>
<td>13</td>
</tr>
<tr>
<td>13.3</td>
<td>Configuration Management tools (CM)</td>
<td>13</td>
</tr>
</tbody>
</table>
Background

This ISEB Foundation Certificate is designed for anyone involved in or affected by the development of IS/IT systems: this extends to Business and Systems Analysts, Designers, Developers, Testers and other users and practitioners who want an understanding of the coverage of Systems Development. Candidates will be able to demonstrate an understanding of the principles systems development and delivery, including, life cycle approaches, architecture, business analysis, requirements engineering, systems modelling, design, development, testing, implementation and communication between the various roles involved in systems development and delivery.

This new exam sits below the range of ISEB modular Certificates in Systems Development and the ISEB Diploma in Solutions Development. In addition it can be used to provide a foundation in the subject for specialists in other disciplines.

1. Introduction to Systems Development (5%)

Objective: To understand the scope of systems development work and its relationship to other associated disciplines.

1.1 What is systems development
1.2 The scope of systems development
1.3 Relationship with other disciplines such as project management, programming, testing, service management, change and configuration management

2. Life Cycle types and their Rationales (10%)

Objective: To understand the range of systems development lifecycles, their application, advantages and disadvantages.

2.1 Systems Development Lifecycles
   • Waterfall model
   • V model
   • Incremental model
   • Spiral model
   • Unified Process
2.2 For each lifecycle
   • Principles and rationale
   • Structure and stages
   • Advantages
   • Disadvantages
   • Selection criteria
   • Team roles and responsibilities
2.3 Adaptation and customisation of the lifecycles
2.4 Project management and the lifecycles
   • Responsibilities of project managers in systems development
   • Difference between project life cycles and systems development life cycles
3. **Business Analysis (5%)**

**Objective:** To understand the objectives, activities and deliverables of business analysis work.

3.1 **Definition of Business Analysis**
- Holistic discipline
- Focus on business problems and opportunities
- Bridge between business and IT

3.2 **Place of Business Analysis in the development lifecycle**
- Feasibility study
- Requirements analysis
- User acceptance testing
- Implementation
- Post-implementation review/benefits realisation

3.3 **Outcomes from Business Analysis**
- People change
- Process change
- IT/IS change
- Organisation change

4. **Requirements Engineering (10%)**

**Objective:** To understand the core activities of the requirements engineering approach and the techniques used to define, document and manage requirements.

4.1 **Key areas of Requirements Engineering**
- Requirements Elicitation
- Requirements Analysis
- Requirements Negotiation
- Requirements Documentation
- Requirements Validation

4.2 **Techniques for requirements elicitation**
- Workshops
- Interviews
- Observation
- Questionnaires
- Scenarios
- Prototyping
- Document Analysis

4.3 **For each elicitation technique**
- Purpose of the technique
- Advantages
- Disadvantages

4.4 **Types of requirements**
- Functional
- Non-functional

4.5 **Prioritising requirements**
- Reasons for prioritisation
- Approach to prioritisation
4.6 Managing requirements
- Recording requirements documentation
- Change Control
- Version control
- Traceability
- CASE tools

4.7 Analysing and validating requirements
- Feasibility checking
- Ensuring the consistency and correctness of the requirements
- The validation process and roles
- Responsibilities of the reviewers

5. Making a Business Case (10%)

Objective: To understand the purpose of producing a business case and the structure and contents of a business case.

5.1 Feasibility checking
- Business feasibility
- Technical feasibility
- Financial feasibility

5.2 Elements of a business case
- Background and context to the business case
- Options
- Costs and benefits for each option
- Impacts of each option
- Risks of each option
- Recommended actions

5.3 Identifying, evaluating and selecting options
5.4 Principles of cost/benefit analysis
5.5 Principles of impact and risk analysis

6. Programming and Development Approaches (10%)

Objective: To appreciate the different approaches to programming and development of software solutions and identify the key features of each.

6.1 Types of development approaches (these are not necessarily mutually-exclusive)
- Agile Approaches
  - Extreme programming
  - Iterative and incremental approaches (Dynamic Systems Development Method)
  - Exploratory/empirical approaches (Scrum, Adaptive Systems Development, Crystal, Lean Development)
  - Feature driven development
  - Test driven development
- Procedural
  - Waterfall lifecycle
  - Structured Programming
- Object Oriented approaches
- Service Oriented
On Demand Software
Application Service Provider (ASP)
- Open source development
- Commercial off-the-shelf packages (COTS)

7. **Systems Modelling and Specification Techniques (10%)**

Objective: To understand the importance of modelling and documentation in the systems development process, to identify the different types models and be aware of the various perspectives they address.

7.1 Reasons for modelling
- To aid Communication between actors
- As a basis for rigorous development
- To provide a standard approach
- To ensure consistency across the development
- To assist in the identification of re-use
- To compare the current situation with the required

7.2 Modelling from different perspectives
- Examples of Models
  - Modelling static data (top down and bottom up)
  - Modelling process and business rules
  - Modelling dynamic behaviour
  - Modelling user interface

7.3 Cross-referencing different modelling perspectives
- Cross-referencing process and data (e.g. CRUD)
- Cross-referencing process and objectives
- Cross-referencing user role and processes

7.4 Documentation and specification
- Importance of documentation
- Documentation configuration management and version control
- Keeping documentation up to date

8. **Systems Design (10%)**

Objective: To recognise the fundamental objectives and principles of good systems design.

8.1 The location of systems design in the systems development lifecycle

8.2 The objectives and constraints of systems design
- Objectives reflect many of the software qualities also identified in section ten. They include to deliver required functionality, reliability, maintainability, flexibility, expandability, usability, efficiency, re-usability, testability and adherence to standards
- Constraints on design include budget, time, skills available, influence of current (legacy) systems, target hardware and software platforms and internal politics

8.3 Input design, input technologies and their application
- Keyboard input considerations including data validation and data verification requirements
  - Existence check
8.4 Output technologies and their application (screen and form design is covered in the interface design and usability section of the syllabus)
- Output design technologies including different types of screens, printers and digital media, public display screens (for example, at bus stops) and mobile telephones
- Application of output design technologies within given scenarios

8.5 The objectives and principles of process design
- Stepwise refinement – the process of the elaboration of requirements
- Expressing processes through the constructs of sequence, condition and repetition
- Modularity and the principles of coupling and cohesion
- The principles of abstraction, encapsulation and generalisation

8.6 The objectives and principles of data design
- The aims and principles (but not conduct) of normalisation
- File organisation methods: serial, sequential, index-sequential and random
- File access methods: searches, indexes and algorithms
- Principles of hierarchical, network, relational and object-oriented database management systems

8.7 The design of codes
- Factors affecting the design of a successful code, including uniqueness, stability, expandability and length
- The principles of facetted codes
- The use and definition of check digits

8.8 The scope and principles of security design
- Physical security of the computer environment
- Logical security measures such as passwords
- Firewalls, anti-virus software and spy-ware
- The function and content of audit trails
- Principles of the Computer Misuse Act
- Principles of the Data Protection Act

9. System Architecture (5%)

Objective: To recognise the importance of a well developed architecture as an integral part of good system development.

9.1 Types of architecture
- Enterprise architecture and IT architecture
- Systems and application architecture
- Data architecture

9.2 Objectives and principles of systems architectures
- What is an IT architecture?
- Why is an architecture important?
- Underlying principles of architectures
9.3 Stakeholders and roles in architecture
- Customers and sponsors
- Service and product providers
- Designers and developers
- IT architect

9.4 Management of the architecture
- Monitoring compliance
- Handling change
- Evolution of architecture

9.5 The tiered architecture approach to IT system development
- The components of an IT architecture
- Interface, process and data layers

9.6 Service Oriented Architecture and Service Oriented Development Applications
- Services and technologies
- Examples of services

10. Quality and Testing (10%)

Objective: To recognise the role of testing through the lifecycle.

10.1 The definition of software quality
- The need to meet robustness and reliability requirements
- The need to meet functional requirements
- The need to meet non-functional requirements, particularly usability
- The need for inherent software product qualities such as maintainability, flexibility and efficiency

10.2 The objectives and limitations of testing
- The causes of software defects and the distinction between errors, faults and failures
- The distinction between static and dynamic testing
- General testing principles
  - Testing shows the presence of defects
  - Exhaustive testing is impossible
  - The principle and benefits of early testing
  - The recognition of defect clustering

10.3 The structure and purpose of the static test stages of the V model
- Requirements
- Functional Specification
- Design Specification
- Module specification

10.4 The purpose and content of the dynamic test stages of the V model
- Component (unit) testing
- Component integration testing
- System testing
  - Functional testing
  - Non-functional testing
- User acceptance testing

10.5 Static testing
- Reviews and the test process
- Informal review
- Walkthrough
- Technical review
• Inspection

10.6 Dynamic testing
• Specification-based or black-box techniques
• Structure-based or white-box techniques
• Experience-based techniques

10.7 Re-testing (confirmation testing)

10.8 Regression testing
• Definition and scope of regression testing
• Opportunities for automating regression testing

11. Implementation and changeover (5%)

Objective: To recognise the importance of careful implementation.

11.1 The task of file and data conversion
• Technical feasibility of converting data
• Alternatives to automatically converting data, such as printing data out and re-entering it

11.2 The principles and problems of data mapping
• The principles of data mapping
• Common problems in data mapping, such as field type incompatibility, field length differences, different field structures and absence of required fields in the current system
• Approaches to dealing with different field structures (for example, converting from one address field to three address line fields) and issues arising from populating newly defined fields with valid data

11.3 Plan, test and undertake data conversion
• Plan the steps and the timing of data conversion
• Write and test the data conversion programs
• The possible role of an automated test comparator in the testing process
• Undertake the actual conversion of live data

11.4 The role of supporting documentation, including user manuals
• The role and structure of an on-line Help facility
• The role and structure of a printed user guide or user manual
• The role and structure of printed operational manuals
• The role and structure of technical documentation designed to allow the continuing support of the delivered software

11.5 Approaches to training
• Training delivered through conventional lectures and workshops
• Training delivered through remote mechanisms, such as web-casts and teleconferencing
• Training delivered through computer-based training (CBT) and e-learning initiatives

11.6 Define training needs and evaluate training effectiveness
• Identify current and proposed competencies
• Define an appropriate strategy, using approaches to training (11.5), to support the gaining of proposed competencies
• Assess the effectiveness of training in supporting these proposed competencies through post-course questionnaires and tests

11.7 Systems implementation
• The principles of direct changeover/conversion
• The advantages and disadvantages of direct changeover/conversion within a given scenario
• The principles of parallel running
• The advantages and disadvantages of parallel running within a given scenario
• The principles of pilot running
• The advantages and disadvantages of pilot running within a given scenario

12. Evaluation and maintenance (5%)

Objective: to recognise the need to evaluate a delivered system and to enhance it through subsequent maintenance.

12.1 The location of maintenance in the systems development life cycle
• Maintenance in abbreviated waterfall and V models
• The explicit reference to maintenance in the b model
• Maintenance in an iterative environment

12.2 The range of metrics which might be used to evaluate a delivered software product
• Characteristics of good metrics (for example, quantifiable, relevant, easy to collect)
• Metrics associated with the business objectives of the project (for example, concerned with improving profitability of the organisation)
• Metrics associated with the functional fit of the delivered software product (for example, number of corrective maintenance changes raised after implementation)
• Metrics associated with the reliability of the software product (for example, acceptable downtime expressed as a percentage)
• Metrics associated with the usability of the software product (for example, time taken to learn the product to an acceptable level)

12.3 The purpose and conduct of a post-implementation review
• The purpose of the post-implementation review is primarily to examine the success of the software product against pre-defined criteria (12.2) and to discuss user feedback and outstanding issues
• Participants in the post-implementation review
• Outcomes from the post-implementation review concerning the delivered software product

12.4 The purpose and conduct of a post-project review
• The purpose of the post-project review is primarily to examine the conduct of the project that delivered the software product. Focus is on what went badly and well with the aim of implementing lessons learnt into improved project or systems development processes
• Participants in the post-project review
• Outcomes from the post-project review concerning the way projects or systems development is conducted

12.5 The distinction between corrective, adaptive and perfective maintenance
• Corrective maintenance is concerned with fixing software faults and failures. This includes fixing requirements which were incorrectly defined and implemented in the previous (or initial) software product release
• Adaptive maintenance is concerned with implementing new or changed requirements into the software product. For example, implementing changes in payroll rules that have arisen since the previous (or initial) software product release
• Perfective maintenance is concerned with making changes to the software to improve, for example, its usability or maintainability. These changes are not as a
result of faults, failures or changes. They are improvements to the general product quality of the software release.

13. Software Support Tools (5%)

Objective: To appreciate the wide range of software support tools, which support IT development.

13.1 Computer Aided Software Engineering tools (CASE)
- The objectives of CASE tools
- Relationship of the CASE tools elements to the software development life cycle
- Facilities of CASE tools
- Advantages and Disadvantages of CASE tools

13.2 Computer Aided Software Testing tools (CAST)
- The objectives of CAST tools
- Relationship of the CAST tools to the software testing life cycle
- Facilities of CAST tools
- Advantages and Disadvantages of CAST tools

13.3 Configuration Management tools (CM)
- The objectives of CM tools
- The importance of CM within the software life cycle
- Facilities of CM tools
- Advantages and Disadvantages of CM tools

Format of the Examination

This syllabus has an accompanying examination at which the candidate must achieve a pass score to gain the ISEB Certificate in Systems Development.

<table>
<thead>
<tr>
<th>Type</th>
<th>40 Question Multiple Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duration</td>
<td>1 Hour</td>
</tr>
<tr>
<td>Pre-requisites</td>
<td>Accredited training is strongly recommended but is not a pre-requisite</td>
</tr>
<tr>
<td>Supervised / Invigilated</td>
<td>Yes</td>
</tr>
<tr>
<td>Open Book</td>
<td>No</td>
</tr>
<tr>
<td>Pass Mark</td>
<td>26/40</td>
</tr>
<tr>
<td>Distinction Mark</td>
<td>None</td>
</tr>
<tr>
<td>Delivery</td>
<td>Paper based examination</td>
</tr>
</tbody>
</table>