Unit 21: Software Applications Testing

Unit code:	**L/601/1984**
QCF Level 4:	**BTEC Higher National**
Credit value:	**15**

Aim

To provide learners with an understanding of the principles of software applications testing as an essential element in the development of commercial applications for delivery to customers.

Unit abstract

Linking into all programming units, this unit supports the detailed exploration, development and deployment of a functional commercial application. Taking the designed and implemented application and ensuring that it is tested and documented to a commercially viable standard.

Learners taking this unit will be able to work on a range of systems, being able to apply the testing techniques to procedural event driven and object oriented systems. There is no particular programming platform or preferred language inherent in the unit, it will support existing programming solutions as well as new developments.

A learner may work on GUI-based systems, a web-based application, a multimedia resource, a mobile (or handheld) application as well as a 'traditional' procedural programming environment to meet the outcomes of this unit. The assurance is that in any of the applications being tested the learner must be systematic and ensure the quality of the system being developed.

Learning outcomes

On successful completion of this unit a learner will:

1 Understand the principles of software application testing

2 Be able to design test strategies

3 Be able to implement test plans

4 Be able to evaluate test plans.

Unit content

1 Understand the principles of software application testing

Specification: user needs eg analysis of requirements, expected outcomes, expected timeline

Dry run of design: testing eg given data, expected outcomes

Implementation: testing techniques eg black box, functional, white (or glass) box; sub-system testing eg integration, whole system, interface

Methodology: testing method eg top down, bottom up, component based, Graphical User Interface (GUI), code, event only, pre-alpha, alpha, beta

Maintenance: procedures eg following changes, reviews, time based, stress/overload

User evaluation: user testing eg against requirements, actual outcomes, acceptance, alpha participation, beta participation

Requirements: resources eg software, hardware, tester time, user time, system access

Documentation: technical documentation eg system and program specifications, user requirements, plans and logs

2 Be able to design test strategies

Test strategy: contents eg timing, justification, functionality, maintainability

Test plan: example data eg normal, erroneous, extreme; expected outcomes eg valid, invalid, information gained; prioritisation

Techniques: types of testing eg black box, functional, white (or glass) box, validation, verification, creation of test cases, reuse of test cases

Versioning: alpha testing in closed test; beta test in open environment; version cycles; bug fixing

3 Be able to implement test plans

Reporting: procedures eg manage reporting process, bug collection protocol, bug response and fix protocol

Tools: testing tools eg Bugzilla

Fault identification: procedures eg prioritisation, categorisation, response

Monitoring: procedures eg adjusting timelines, time management, allocation of resources, feedback to customer, managing adherence

4 Be able to evaluate test plans

Evaluation: of features eg functionality, accuracy, effectiveness, alterations to tests carried out, timeliness; of possible improvements eg program specification and design, self-reflection on product, aspects of test management; of maintainability eg usefulness to self, usefulness to others, usefulness for customers

Learning outcomes and assessment criteria

Learning outcomes **On successful completion of this unit a learner will:**	**Assessment criteria for pass** **The learner can:**
LO1 Understand the principles of software application testing	1.1 evaluate testing techniques applicable to the testing opportunity 1.2 compare the relative benefits of different testing methodologies 1.3 justify a proposed testing methodology
LO2 Be able to design test strategies	2.1 design a test strategy for a given testing opportunity 2.2 design a test plan for a given testing opportunity 2.3 justify the test plan proposition and testing strategy
LO3 Be able to implement test plans	3.1 implement a test plan based on a given testing opportunity
LO4 Be able to evaluate test plans	4.1 critically review the test outcomes 4.2 justify the validity of the test and identify any potential issues.

Guidance

Links to National Occupational Standards, other BTEC units, other BTEC qualifications and other relevant units and qualifications

The learning outcomes associated with this unit are closely linked with:

Level 3	Level 4	Level 5
Unit 6: Software Design and Development	Unit 18: Procedural Programming	Unit 35: Web Applications Development
Unit 14: Event Driven Programming	Unit 19: Object Oriented Programming	Unit 39: Computer Games Design Development
Unit 15: Object Oriented Programming	Unit 20: Event Driven Programming Solutions	Unit 40: Distributed Software Applications
Unit 16: Procedural Programming	Unit 22: Office Solutions Development	Unit 41: Programming in Java
	Unit 23: Mathematics for Software Development	Unit 42: Programming in .NET

This unit has links to the Level 4 and Level 5 National Occupational Standards for IT and Telecoms Professionals, particularly the areas of competence of:

- Software Development
- IT/Technology Solution Testing.

Essential requirements

Learners must have access to facilities, which allow them the opportunity to fully evidence all of the criteria of the unit. If this cannot be guaranteed then centres should not attempt to deliver this unit.

Employer engagement and vocational contexts

Working with a local programming-based organisation or using internet-based open source projects would enhance the learners' experience and offer a relevant vocational context.