Unit 51: Software Design Fundamentals

Unit code: L/601/3251

QCF Level 3: BTEC Specialist

Credit value: 10

Guided learning hours: 80

Aim and purpose

This unit introduces the principles of software design and the application of software design techniques.

Unit introduction

To develop a programmed software solution that meets business and user needs, it is necessary to understand the problem in question and be clear in terms of user requirements. Issues are often caused by poor understanding of user need as well as poor planning. A wide range of different development programming languages and paradigms is available to developers with quite different characteristics and features.

This unit focuses on the design and development process, following the software development lifecycle and investigating software structures and looking at the features of programming paradigms.

A major part of learners' time will be spent on familiarising themselves with fundamental software development processes and concepts. This will give them a foundation in programming techniques.

Learning outcomes and assessment criteria

In order to pass this unit, the evidence that the learner presents for assessment needs to demonstrate that they can meet all the learning outcomes for the unit. The assessment criteria determine the standard required to achieve the unit.

On completion of this unit a learner should:

Learning outcomes	Assessment criteria
1 Understand the principles of software design	1.1 describe the role of software design and computer programming in the IT Systems Development Life Cycle (SDLC) 1.2 describe the application and limits of programming paradigms procedural, object oriented and event driven and the available supporting tools and environments (eg case tools, IDEs) 1.3 explain sequence, selection and iteration as used in computer programming 1.4 explain abstraction of data and code and the use of predefined data and code in computer programming 1.5 explain the importance of the readability and understandability of code and how these can be improved by naming, comments and layout 1.6 describe how the following factors contribute to the quality of code: efficiency, reliability, robustness, usability, portability and maintainability
2 Apply the techniques of software design	2.1 develop algorithms to represent problems 2.2 identify and define data and file storage requirements including predefined data items 2.3 identify and define program structures including predefined code items 2.4 identify and represent required inputs and outputs 2.5 use tools (eg Pseudo code) to express software designs.

Unit content

1 Understand the principles of software design

Software development life cycle: stages eg determination of scope, requirements gathering and specification, software design, coding, testing, maintaining

Programming paradigms: procedural; object oriented; event driven; supporting tools and environments eg CASE tools, IDE; application of each eg when appropriate; limits eg when not appropriate

Features: sequence; selection eg case, if … then … else; iteration eg repeat – until, while … do

Software structures: abstraction of data and code eg lookup tables, objects; pre-defined data and code eg common data structures and statements; readability and understandability eg comments, appropriate names for variables, indentation

Quality of code: efficiency eg logical, structured, no redundancy; reliability eg consistent, always works; robustness eg copes with extreme data; usability; portability eg user interaction, prototyping, multi-platform, compatible frameworks; maintainability eg understandable, adaptable, modular

2 Apply the techniques of software design

Algorithms: sequence of instructions to solve a problem eg Pseudo code, structured English, flow chart

Tools: algorithms; other eg structure diagrams, DFDs, ERM

Data types: text; integer; floating point; byte; date; Boolean; other eg char, smallint; predefined data items; benefits of appropriate choice of data type eg additional validation, efficiency of storage

Program structures: eg functions, procedures, classes, objects; data; predefined code; input and output; data and file storage

Essential guidance for tutors

Delivery

Although this unit does not require learners to produce a working computer program, much of the theory can come from programming activities and learners will appreciate a hands-on approach.

The software development cycle should be constantly referred to as the structure for developing programmed solutions. Learners will need to appreciate the difference between the different programming paradigms and when they are used. Preferably learners will be able to try suitable languages themselves, using simple exercises and looking at the supporting tools such as CASE. Understanding the features of sequence, selection and iteration is best done through practical exercises in a chosen language. The items listed under *software structures* can be extracted from practical work and examples of good practice.

The various design tools, such as data flow diagrams, may have been introduced to learners in other units. Note that these are given as examples of design tools and the tools chosen should be appropriate to the task. Data types will come from programming activities as will program structures.

Outline learning plan

The outline learning plan has been included in this unit as guidance and can be used in conjunction with the programme of suggested assignments. The outline learning plan demonstrates one way in planning the delivery and assessment of this unit.

Topic and suggested assignments/activities and/assessment
Introduction to the unit
Principles of software design • Development life cycle – tutor led, case studies/examples • Programming paradigms – tutor led, examples of each language, practical • Tools – eg CASE, examples, practical • Sequence, selection, iteration – examples, practical • Software structures – examples, practical • Quality – tutor led, examples, practical.
Assignment 1 - Good Practice Guide
• Techniques of software design • Tools – DFDS etc; examples, practical • Algorithms – eg Pseudo code; examples, practical • Data types, structures – practical.
Assignment 2 - Designing a Solution

Assessment

It is suggested that this unit is assessed using the two assignments summarised in the *Programme of suggested assignments* table.

Assignment 1

For learning outcome 1 learners can be asked to produce a Good Practice Guide for new programmers covering the assessment criteria 1.1 to 1.6. The guide should be illustrated and use examples wherever possible.

Assignment 2

Learners need to be provided with a scenario outlining a business problem requiring a programmed solution eg an online business wants to give discount on its products dependent on the value of the order and the quantity of goods ordered.

Following the assessment criteria 2.1 to 2.5, learners should design a solution to the problem. This could be done in practice and the evidence presented as annotated program listings with the exception of 2.5, which is a design element.

Programme of suggested assignments

The table below shows a programme of suggested assignments that cover the assessment criteria in the assessment and grading grid. This is for guidance and it is recommended that centres either write their own assignments or adapt any Edexcel assignments to meet local needs and resources.

Criteria covered	Assignment title	Scenario	Assessment method
1.1, 1.2, 1.3, 1.5, 1.6	Good Practice Guide	You are to produce a good practice guide for trainee programmers.	Booklet.
2.1, 2.2, 2.3, 2.4, 2.5	Designing a Solution	You have been asked to produce a programmed solution for a business requiring a method of deciding discounts on goods for different categories of customer.	Design documentation.

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

This unit forms part of the BTEC in IT sector suite. This unit has particular links with:

Level 1	Level 2	Level 3
		Systems Analysis and Design

This unit maps to some of the underpinning knowledge from the following areas of competence in the Level 3 National Occupational Standards for IT (ProCom):

4.6 Human Computer Interaction/Interface (HCI) Design

5.2 Software Development.

Essential resources

Learners will need individual access to a particular programming language and development environment. They will also need a more limited access, possibly through demonstration, to other different types of languages.

Learners will require access to computer equipment to enable them to gain a practical awareness and enable them to apply their knowledge and understanding in a practical situation.

Employer engagement and vocational contexts

The use of vocational context is essential in the delivery and assessment of this unit.

There is a range of organisations that may be able help centres to engage and involve local employers in the delivery of this unit, for example:

- Learning and Skills Network – www.vocationallearning.org.uk
- Local, regional business links – www.businesslink.gov.uk
- National Education and Business Partnership Network – www.nebpn.org
- Network for Science, Technology, Engineering and Maths Network Ambassadors Scheme – www.stemnet.org.uk
- Work-based learning guidance – www.aimhighersw.ac.uk/wbl.htm
- Work experience/workplace learning frameworks – Centre for Education and Industry (CEI University of Warwick) – www.warwick.ac.uk/wie/cei

Indicative reading for learners

Textbooks

Bowman K – *Systems Analysis: A Beginner's Guide* (Palgrave Macmillan, 2003) ISBN-10 033398630X, ISBN-13 978-0333986301

Flanagan D – *JavaScript Pocket Reference, 2nd edition* (O'Reilly, 2002) ISBN-10 0596004117, ISBN-13 978-0596004118

Knuth D – *The Art of Computer Programming: Volumes 1–3, 2nd Edition* (Addison Wesley, 1998) ISBN-10 0201485419, ISBN-13 978-0201485417

Wang W – *Visual Basic 6 for Dummies* (John Wiley & Sons, 1998) ISBN-10 0764503707, ISBN-13 978-0764503702

Wender K – *Cognition and Computer Programming* (Ablex Publishing Corporation, 1995) ISBN-10 1567500951, ISBN-13 978-1567500950

Willis T, Crossland J and Blair R – *Beginning VB.NET, 3rd edition* (John Wiley & Sons, 2004) ISBN-10 0764556584, ISBN-13 978-0764556586

Websites

www.guidetoprogramming.com/joomla153

www.profsr.com

www.vbexplorer.com/VBExplorer/VBExplorer.asp

www.visualbasic.about.com

Functional Skills – Level 2

Skill	When learners are …
ICT - Using ICT	
plan solutions to complex tasks by analysing the necessary stages	describing the stages of the software development lifecycle
ICT - Developing, presenting and communicating information	
use appropriate software to meet the requirements of a complex data-handling task	designing a software solution
evaluate the selection, use and effectiveness of ICT tools and facilities used to present information	reviewing the design against the original requirement