2016 Suite

OCR
Oxford Cambridge and RSA

Cambridge **TECHNICALS LEVEL 3**

Cambridge TECHNICALS 2016

IT

Unit 14

Software engineering for business

H/507/5017

Guided learning hours: 60

Version 1 September 2015

ocr.org.uk/it

LEVEL 3

UNIT 14: Software engineering for business

H/507/5017

Guided learning hours: 60

Essential resources required for this unit: A programming language installed and available for all learners to use and a tutor that is proficient with this language.

This unit is internally assessed and externally moderated by OCR.

UNIT AIM

The aim of this unit is to give you practical experience of writing computer programs for specific requirements, such as those found in a business. Programmers' first jobs tend to be coding against specific requirements or maintaining an existing program. You need only a basic appreciation of the full system life cycle at this level, freeing up more time for practical programming experience.

This unit focuses on developing code for a single customer with specific requirements. If you are on the application developer pathway, you could follow this unit with the games design and prototyping unit, which focuses on developing for a mass market, allowing you to further develop your coding skills and experience.

This unit is optional within the Applications Developer and Data Analyst specialist pathways. Application developers create, test and program applications software and therefore use the knowledge skills and understanding associated with software engineering. Data analysts require a good understanding of computers and software and an insight into software engineering can assist them in their design and analysis of systems and data.

TEACHING CONTENT

The teaching content in every unit states what has to be taught to ensure that learners are able to access the highest grades.

Anything which follows an i.e. details what must be taught as part of that area of content. Anything which follows an e.g. is illustrative, it should be noted that where e.g. is used, learners must know and be able to apply relevant examples in their work, although these do not need to be the same ones specified in the unit content.

For internally assessed units you need to ensure that any assignments you create, or any modifications you make to an assignment, do not expect the learner to do more than they have been taught, but must enable them to access the full range of grades as described in the grading criteria.

Learning outcomes **The Learner will:**	**Teaching content** **Learners must be taught:**
1. Understand universal programming constructs	1.1 Programming constructs, i.e.: • encapsulation (e.g. modules, procedures, functions, classes, properties and methods) • predefined subroutines (e.g. string manipulation, screen output). • the use of parameters and return values (one technique is sufficient and depends on chosen language e.g. functions and return value, byref and byval parameters, getters and setters) • declaring and using variables with fundamental data types: o integer o floating point o string (or equivalent e.g. array of chars) o boolean • converting between data types • selection (if, then, elseif, else, endif) • iteration (fixed loop, conditional loop e.g. pre-condition, post condition) • GUI objects and their different applications (e.g. text box, label, button, check box, radio button, dropdown list, combo box, multi-line lists, menus)
2. Be able to investigate business requirements for programming solutions	2.1 A modern incremental system life cycle in which stakeholder feedback feeds back into the next round of development (e.g. Rapid application development (RAD), Agile, Spiral, Scrum) 2.2 Requirements specification, i.e.: • requirements (e.g. customer requirements of the new system) • investigation (e.g. existing system observations, data used, interview with the stakeholder) • analysis (e.g. tasks the existing system carries out, mission critical tasks, good and bad points about the existing system, integration with other systems, any specific processing known). • feasibility (e.g. scope, cost/benefit analysis) • phased development (which requirements will be in which versions)

Learning outcomes The Learner will:	Teaching content Learners must be taught:
3. Be able to develop software solutions to meet business requirements	3.1 Design specification, i.e.: • what is it going to look like? (UI design) • what is it going to do? (e.g. flowchart, JSD, pseudocode) • processing (e.g. algorithms, mathematical formulae, input process output) • coding • methodology (e.g. top-down with stubs, bottom-up with reusable modules) • meaningful names (e.g. modules, procedures, variables) • house style (e.g. Hungarian notation) • debug tool (e.g. trace comments, profilers, breakpoints, stepping and watches) • testing • against requirements, to prove project completion and avoid requirements creep (stakeholders adding more requirements into the project as it progresses) • robustness (e.g. invalid data range, invalid data type, data extremes, data presence) • repeatable test data (data that is used in the tests) 3.2 Adaptations to design, i.e.: • obtain feedback from stakeholders • analyse feedback • confirm viability of changes • make viable changes to design based on analysed feedback 3.3 Design evaluation, i.e.: • how and why does the design meet the stakeholder requirements?
4. Be able to propose software solutions to meet business requirements	4.1 Presenting to an audience, i.e.: • personal skills (e.g. speech clarity and speed, appearance, gait, personal cleanliness) • interpersonal skills (e.g. facing audience, eye contact, verbal confidence, non-verbal signals such as smiling, hand gestures, nerves and rocking, handling difficult questions) • communication skills (e.g. summarising concepts, adapting content to audience composition, polite addressing of strangers, use of appropriate speech, the dangers of verbose slides, the dangers of auto-timed presentations) • preparation (e.g. prompt cards, practice, knowing what is wanted from a meeting before it starts, preparation for equipment failure, potential questions) 4.2 Adaptations to prototype, i.e.: • obtain feedback from stakeholders • analyse feedback • confirm viability of changes • make viable changes to prototype based on analysed feedback

GRADING CRITERIA

LO	Pass	Merit	Distinction
	The assessment criteria are the Pass requirements for this unit.	To achieve a Merit the evidence must show that, in addition to the pass criteria, the candidate is able to:	To achieve a Distinction the evidence must show that, in addition to the pass and merit criteria, the candidate is able to:
1. Understand universal programming constructs	P1: Explain the purpose of programming constructs when designing software for business		
2. Be able to investigate business requirements for programming solutions	P2*: Outline the requirements for a software solution to meet an identified business need *(*Synoptic assessment from Unit 1 Fundamentals of IT, Unit 2 Global information and Unit 3 Cyber security)*	M1: Propose a requirements specification for a software solution	D1: Discuss the feasibility of the proposed software solution
3. Be able to develop software solutions to meet business requirements	P3: Prepare a design specification for an identified software solution P4: Create the software solution prototype based on the design specification P5: Test the prototype software solution, rectifying issues	M2: Make adaptations to final design specification following negotiations with stakeholders	D2: Evaluate the software solution prototype against stakeholder requirements
4. Be able to propose software solutions to meet business requirements	P6: Demonstrate the software solution prototype to relevant stakeholders	M3: Adapt the software solution prototype based on stakeholder feedback	

SYNOPTIC ASSESSMENT

When learners are taking an assessment task, or series of tasks, for this unit they will have opportunities to draw on relevant, appropriate knowledge, understanding and skills that they will have developed through other units. We've identified those opportunities in the grading criteria (shown with an asterisk). Learners should be encouraged to consider for themselves which skills/knowledge/understanding are most relevant to apply where we have placed an asterisk.

ASSESSMENT GUIDANCE

LO1 Understand universal programming constructs

P1: Learners should explain the purpose of the different constructs in the teaching content. The explanations should include what they do and could be supported by examples. This could be in the format of a booklet, teaching resource, technical guide or report detailing and explaining the purpose.

LO2 Be able to investigate business requirements for programming solutions

P2: Learners are required to outline the requirements for a software solution to meet an identified business need. Learners should use the headings listed in the teaching content. Rather than describing generally what each section might contain, this must be related to the given business need. The evidence will be the outline requirements for the software solution.
M1: Learners are required to propose a requirements specification for a software solution. The outline requirements specification from P2 should be extended to clearly show where the new system will fit into the existing business, what it will replace and the existing processes with which it needs to integrate. The requirements will be achievable and provable. The evidence will be the proposed requirements specification.
D1: Learners are required to discuss the feasibility of the proposed software solution. The feasibility should be sufficiently detailed to confirm that all aspects of the business need and proposed software solution have been considered. The evidence could be presented as a report or as a presentation with detailed speaker notes.

LO3 Be able to develop software solutions to meet business requirements

P3: Learners are required to prepare a design specification based on the outline solution produced for P2. This could be a UI design, a form of flow design and described processing. The UI can be designed directly onto the IDE using the GUI design tools, or drawn either on paper or electronically. The flow can be pseudo code, flowchart, JSD or equivalent design method. There should also be a description of processing needed to turn the expected input into the desired output. This can be an algorithm or mathematical formula.
M2: Learners are required to make adaptations to their designs based on stakeholder feedback. Some requirements may change; some may be dropped or rephrased to avoid requirements creep. This may give learners an opportunity to identify with the stakeholders which requirements are critical, which are only desirable and in which release each requirement will appear. Any discussion should then be documented along with changes to the design specification.
P4: The program does not have to successfully meet all requirements but the prototype should follow the required specification. The main form of evidence will be the created prototype. The program code does not need to be printed out but must be available for inspection on request.

P5: There must be a test plan mapping every requirement against at least one test and it should include tests for robustness. There should be evidence of real data for repeatable tests, though not all tests are required to have actual data. Tests should be accompanied with evidence that the test was carried out, such as screen shots or a video of the program running them. Though not all tests need to be passed, in the case of a failed test, there should be evidence of how learners attempted to find the problem and rectify it.
D2: Learners will evaluate how each requirement was met or attempted. They should use their M1, M2 and D1 to feed into D2. For each requirement, learners will note how successful their solution was in satisfying the requirement. The evidence will be the documented evaluation and could be in the form of a report or a presentation with detailed speaker notes.

LO4 Be able to propose software solutions to meet business requirements

P6: Learners must demonstrate the software solution prototype to their stakeholders, formal panel, or a person unfamiliar to the project. Slides and hand-outs can be used if desired, but the presentation must feature a demonstration of the program running. The evidence could be in the form of a video of the learner delivering the presentation; presentation slides with detailed speaker notes should be supported by detailed witness statements and the actual prototype.
M3: Learners should analyse the feedback from the stakeholders and make adaptations to their software solution prototype based on the feedback. The evidence will be the feedback from the stakeholders, the adaptations to the software solution design and the prototype.

Feedback to learners: you can discuss work-in-progress towards summative assessment with learners to make sure it's being done in a planned and timely manner. It also provides an opportunity for you to check the authenticity of the work. You must intervene if you feel there's a health and safety risk.

Learners should use their own words when producing evidence of their knowledge and understanding. When learners use their own words it reduces the possibility of learners' work being identified as plagiarised. If a learner does use someone else's words and ideas in their work, they must acknowledge it, and this is done through referencing. Just quoting and referencing someone else's work will not show that the learner knows or understands it. It has to be clear in the work how the learner is using the material they have referenced **to inform their** thoughts, ideas or conclusions.

For more information about internal assessment, including feedback, authentication and plagiarism, see the centre handbook. Information about how to reference is in the OCR *Guide to Referencing* available on our website: http://www.ocr.org.uk/i-want-to/skills-guides/.

EMPLOYABILITY SKILLS

Employability skills	Learning outcome
Communication	P6, M2, M3
Problem solving/decision making	P2, P3, P4, P5, M1, M2, M3
Time management	P4, P6
Critical thinking	M2, M3, D1, D2

MEANINGFUL EMPLOYER INVOLVEMENT - a requirement for the Diploma (Tech Level) qualifications

The ‘Diploma’ qualifications have been designed to be recognised as Tech Levels in performance tables in England. It is a requirement of these qualifications for centres to secure for every learner employer involvement through delivery and/or assessment of these qualifications.

The minimum amount of employer involvement must relate to at least one or more of the elements of the mandatory units.

Eligible activities and suggestions/ideas that may help you in securing meaningful employer involvement for this unit are given in the table below.

Please refer to the *Qualification Handbook* for further information including a list of activities that are not considered to meet this requirement.

Meaningful employer involvement	Suggestion/ideas for centres when delivering this unit
1. Learners undertake structured work experience or work placements that develop skills and knowledge relevant to the qualification.	Learners could have work experience with local software development organisation where they job shadow and support industry practitioners in the development of software. This would provide them with an insight into the importance of the development lifecycle and the need to meet strict deadlines as well as the stages taken when developing software.
2. Learners undertake project(s), exercises(s) and/or assessments/examination(s) set with input from industry practitioner(s).	Local industry practitioners could assist you in developing realistic scenarios based on the types of development activities that are required within the sector.
3. Learners take one or more units delivered or co-delivered by an industry practitioner(s). This could take the form of master classes or guest lectures.	Industry practitioners could deliver the technical elements of the unit for example, programming constructs, how to investigate business requirements or how to develop software solutions to meet business requirements.
4. Industry practitioners operating as ‘expert witnesses’ that contribute to the assessment of a learner’s work or practice, operating within a specified assessment framework. This may be a specific project(s), exercise(s) or examination(s), or all assessments for a qualification.	If an Industry practitioner worked with you on point 2, they could assist in the assessment of the learners’ evidence based on the scenario that they assisted in developing.

To find out more
ocr.org.uk/it
or call our Customer Contact Centre on **02476 851509**

Alternatively, you can email us on **vocational.qualifications@ocr.org.uk**

A DIVISION OF
CAMBRIDGE ASSESSMENT

Oxford Cambridge and RSA