Module Specification

Collaborative Software Project

Version: 2023-24, v2.0, 16 Mar 2023

Contents

Part 1: Information

Module title: Collaborative Software Project

Module code: UFCFFN-30-3

Level: Level 6

For implementation from: 2023-24

UWE credit rating: 30

ECTS credit rating: 15

Faculty: Faculty of Environment & Technology

Department: FET Dept of Computer Sci & Creative Tech

Partner institutions: None

Delivery locations: Not in use for Modules

Field: Computer Science and Creative Technologies

Module type: Module

Pre-requisites: None

Excluded combinations: None

Co-requisites: None

Continuing professional development: No

Professional, statutory or regulatory body requirements: None

Part 2: Description

Overview: Not applicable

Features: Not applicable

Educational aims: See Learning Outcomes

Outline syllabus: The primary role of a software engineer is to be able to design, build and test high-quality software solutions following best practices and industry standards. They will typically be working as part of a larger collaborative team, in which they will have responsibility for significant elements of the overall project. The developer will need to be able to interpret requirements, specification documentation and designs in order to develop and test software that meets its requirements, even when these requirements may change.

Using industry recognised techniques and language apprentices will:

Roles and responsibilities that are required from a software engineer at every stage of the development lifecycle.

Issues and constraints that could appear for a software company looking to build or edit software.

Collaborate as a team to apply systems analysis and design to a project specification creating artefacts (e.g. use case).

Interpret and implement a design that's compliant security requirements (e.g. functional and non-functional).

Collaboratively create a program based on user requirements, embracing an industry based methodology (e.g. Agile).

Create software using industry standard build processes, and tools for configuration management, version control and software build (e.g. GitHub), release and deployment into enterprise environments.

Create effective and secure software solutions using contemporary software development languages, producing high quality code with sound syntax applying secure and robust development techniques to increase code resilience.

Collaboration as cross functional teams to achieve a common goal.

Code reviews, debugging and refactoring to improve code quality and efficiency.

Thoroughly test a solution to ensure resilience of code and that it meets the functional and non-functional requirements (e.g. black box, white box, unit testing).

Part 3: Teaching and learning methods

Teaching and learning methods: Introductory lectures are supported by seminars, case studies, visits and practical workshops. In addition this module will be supported by interactive forums and learning tools.

300 hours study time of which 72 hours will represent scheduled learning. Scheduled learning includes lectures, seminars, tutorials, demonstration, practical classes and workshops; external visits; supervised time in studio/workshops.

Independent learning includes hours engaged with essential reading, case study preparation, assignment preparation and completion. Apprentice study time will be organised each week with a series of both essential and further readings and preparation for practical workshops.

This unit practically based and designed to ensure that apprentices understand and develop their skills in advanced programming techniques. Apprentices will use the object-oriented facilities within C++ as a vehicle for this.

Module Learning outcomes: On successful completion of this module students will achieve the following learning outcomes.

MO1 Identify, justify and operate at all stages of the software development lifecycle.

MO2 Work collaboratively to effectively develop software solutions embracing Agile and other development approaches.

MO3 Identify, justify and use software analysis and design approaches.

MO4 To interpret and implement a design, compliant with functional, non-functional and security requirements.

MO5 How to perform functional and unit testing.

MO6 Use and apply the range of software tools used in software engineering.

MO7 Critically appraise the business environment and business issues related to software development.

MO8 Create effective and secure software solutions using contemporary software development languages to deliver the full range of functional and non-functional requirements using relevant development methodologies.

MO9 Undertake analysis and design to create artefacts, such as use cases to produce robust software designs.

MO10 Produce high quality code with sound syntax in at least one language following best practices and standards.

MO11 Perform code reviews, debugging and refactoring to improve code quality and efficiency.

MO12 Test code to ensure that the functional and non-functional requirements have been met.

MO13 Deliver software solutions using industry standard build processes, and tools for configuration management, version control and software build, release and deployment into enterprise environments.

MO14 Work collaboratively and professionally with others in cross functional teams.

MO15 Apply secure and robust development principles to ensure software resilience.

Hours to be allocated: 300

Contact hours:

Independent study/self-guided study = 228 hours

Face-to-face learning = 72 hours

Total = 300

Reading list: The reading list for this module can be accessed at readinglists.uwe.ac.uk via the following link https://uwe.rl.talis.com/lists/34CEA324-A7C5-DF21-7596-AB10614A3262.html

Part 4: Assessment

Assessment strategy: This module is assessed by a combination of techniques: a design document and a practical build.

Design Documentation
Apprentices will need to complete documentation on how they are going to create their project against an industry recognised lifecycle model (e.g. Agile). This should clearly identify their roles and responsibilities within each stage of the lifecycle model.

Apprentices will be required to perform appropriate software design using their own chosen methodology and paradigm (e.g. object oriented, event driven) for a chosen business specification. They will need to complete a robust and sound design process using various designs that should include multiple versions.

Practical Build
Apprentices will be given multiple business specifications from which they will produce a solution. This should be based on their designs from element 1; however, changes can be made to the design process and modified to provide documentation.

Team work will be evident throughout all stages of the development so there will be a large emphasis on the usage of tools to provide team-working at all stages of the development lifecycle. There will need to be clear evidence of the individual work that each team member has completed, but also a record of the collaboration that has been performed as a team.

Apprentices will incorporate development, implementing, testing and debugging. During development, this would include tools such as GitHub for version control and collaboration.

The testing of the program will need to be robust and thorough, using techniques such as white and black box testing and unit testing. The program must be fully documented and following the paradigm/methodology chosen, conforming to industry standards. After the testing, apprentices will be required to modify their existing code base and apply new fixes or modules from their requirements and testing.

Apprentices will be required to show their project at an end of year meeting to go through; how the completion of the project went, the finished project, and the evaluation of the overall project.

Opportunities for formative assessment exist for the assessment strategy used. Verbal feedback and written feedback is given to all apprentices providing a personal platform for improvement.

Assessment components:

Portfolio (First Sit)
Description: Design portfolio
Weighting: 40 %
Final assessment: No
Group work: Yes
Learning outcomes tested: MO1, MO2, MO3, MO4, MO5, MO6, MO7

Practical Skills Assessment (First Sit)
Description: Practical assessment
Weighting: 60 %
Final assessment: Yes
Group work: Yes
Learning outcomes tested: MO10, MO11, MO12, MO13, MO14, MO15, MO5, MO6, MO8, MO9

Portfolio (Resit)

Description: Design portfolio

Weighting: 40 %

Final assessment: No

Group work: Yes

Learning outcomes tested: MO1, MO2, MO3, MO4, MO5, MO6, MO7

Practical Skills Assessment (Resit)

Description: Practical assessment

Weighting: 60 %

Final assessment: Yes

Group work: Yes

Learning outcomes tested: MO10, MO11, MO12, MO13, MO14, MO15, MO5, MO6, MO8, MO9

Part 5: Contributes towards

This module contributes towards the following programmes of study:

Digital and Technology Solutions (Software Engineer) {Apprenticeship-UCW} [Sep][FT][UCW][4yrs] BSc (Hons) 2020-21