Module Specification

Object-Oriented Systems Development

Version: 2026-27, v2.0, Approved

Contents

Part 1: Information

Module title: Object-Oriented Systems Development

Module code: UFCFB6-30-2

Level: Level 5

For implementation from: 2026-27

UWE credit rating: 30

ECTS credit rating: 15

College: College of Arts, Technology and Environment

School: CATE School of Computing and Creative Technologies

Partner institutions: None

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: This module extends students' understanding of the object-oriented paradigm. It deepens knowledge of class abstraction, and develops the use of the Unified Modeling Language (UML) for modelling classes, interactions, and system behaviour. Students explore and apply a range of design patterns to develop robust, reusable, and maintainable code.

Features: Not applicable

Educational aims: The aim of this module is to further develop the object-oriented programming skills of students, enabling them to develop and document complete applications.

Outline syllabus: Topics are likely to include but are not limited to:

Theory and concepts of the object-oriented paradigm: objects and classes, encapsulation and visibility, cohesion and coupling, instance and class scope attributes and methods, inheritance and polymorphism, leading to abstract classes – overriding and overloading.

Object-oriented systems software development lifecycles: phases (requirements capture and analysis, design, code, test), iterative / incremental and waterfall methods.

Object-oriented analysis and design techniques: use cases (user stories), verb-noun analysis, abstraction, Class-Responsibility Collaboration (CRC), and heuristics for design evaluation and test case generation.

Tools: Unified Modeling Language (UML) tools. An Integrated Development Environment (IDE) supporting projects, packages, classes, build scripts, deployment, execution, debugging, testing, and version control.

Software Patterns and Architecture: Model-View-Controller (MVC), analysis patterns (e.g. Arlow and Neustadt), design patterns (e.g. Gamma et al.), architectural patterns e.g. separation of concerns for multi-tier distributed software systems and interoperability for systems of increasing scale, complexity, multiple interactive channels and virtualisation.

Persistence: Input / output, serialisation, database connectivity, entity design and implementation (tables, inserts, queries, result sets etc.)

Concurrency and Distributed Systems: Theory and potential solutions relating to the changes of concurrency, design and implementation of concurrent systems.

Graphical User Interfaces: user-centric design, usability testing, event-driven paradigms – graphical components, events, listeners, handlers.

Part 3: Teaching and learning methods

Teaching and learning methods: Scheduled learning includes interactive lectures, wherein the theory and practice of object-oriented systems design and development are demonstrated; questions are invited and freely discussed. Audio recordings are taken of all lectures and made available as podcasts via the Virtual Learning Environment (VLE). All lecture slides, recommended articles, videos and tutorial notes are available on the VLE. Also within the interactive discursive tutorials, students are encouraged to articulate and present their analysis and design models of a realistic industrial case study, as well as implementation source code and associated tests, to their peers in small groups. Interactive cohort peer-review is then encouraged as a mechanism for self-evaluation and reflection upon software design and code artefacts. In addition, continual tutor and peer feedback provides the essential component of the deep learning environment provided to students.

Independent learning includes hours engaged with essential reading, case study preparation, assignment preparation and completion etc. Explicit direction is given to students with respect to selected reading materials for self-study – more information is provided in the following section. Self-study is crucial to this module as individual student reflection is a key technique for learning software design and development. The VLE offers podcasts and presentations for students to interact with at their own pace. Also, high quality, robust Open Source software tools have been selected to enable maximum portability and so ease of installation on a variety of students' own laptop platforms for self-study, and are available free of charge. Having the same tools consistently available on faculty workstations and student laptops, when taken together with the VLE, enables great interoperability between development artifacts, promoting virtualisation of learning location. The learnings achieved in self-study are then brought forward by the student and reinforced at the interactive tutorials

wherein their knowledge and understanding are deepened by directed articulation, presentation and evaluation with their peers and tutors.

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

MO1 Apply Object-Oriented principles to analyse and design software systems.

MO2 Demonstrate proficiency in creating UML diagrams including use case, class and sequence diagrams.

MO3 Develop skills in designing Object-Oriented systems to apply design patterns and best practices in the implementation of a robust and maintainable software application.

MO4 Design and implement tests to ensure the correctness of software applications and understand the importance of testing in the software development lifecycle.

Hours to be allocated: 300

Contact hours:

Independent study/self-guided study = 228 hours

Face-to-face learning = 72 hours

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/modules/ufcfb6-30-2.html

Part 4: Assessment

Assessment strategy: The assessment strategy for this module is based on a coursework assignment. In the coursework assignment the students will design and implement a moderately realistic object-oriented system. They will produce detailed object models and designs from system requirements using the modelling concepts provided by UML. The students will then map the designs into code and perform unit testing using automated testing tools. Assessment of this will include a demonstration and submission of a portfolio. In this assessment apart from

determining the technical progress of the students, their professional approach when dealing with group dynamics will also be assessed.

Students will have the opportunity for formative feedback during practical lab/tutorial sessions.

The resit corresponds to the main sit work and will be assessed by a demonstration and the submission of a portfolio that contains the problem analysis, requirements specification, design etc. In the event that only a single student is referred, the groupwork aspect of the referral will require the student reflect on how the project would have benefited or been hindered by being undertaken and managed as a group.

Assessment tasks:

Portfolio (First Sit)
Description: Portfolio
Weighting: 100 %
Final assessment: Yes
Group work: Yes
Learning outcomes tested: MO1, MO2, MO3, MO4

Portfolio (Resit)
Description: Portfolio
Weighting: 100 %
Final assessment: Yes
Group work: Yes
Learning outcomes tested: MO1, MO2, MO3, MO4

Part 5: Contributes towards

This module contributes towards the following programmes of study:

Software Engineering for Business {Foundation} [Frenchay] BSc (Hons) 2024-25

Software Engineering for Business {JEP} [Neusoft] BSc (Hons) 2024-25

Software Engineering for Business [Frenchay] BSc (Hons) 2025-26

Software Engineering for Business [Frenchay] BSc (Hons) 2025-26

Software Engineering for Business {Foundation} [Frenchay] BSc (Hons) 2024-25

Business Computing {Foundation} [Frenchay] - WITHDRAWN BSc (Hons) 2024-25