MODULE SPECIFICATION

Part 1: Information			
Module Title	Object-Oriented Systems Development		
Module Code	UFCFB6-30-2	Level	Level 5
For implementation from	2019-20		
UWE Credit Rating	30	ECTS Credit Rating	15
Faculty	Faculty of Environment & Technology	Field	Computer Science and Creative Technologies
Department	FET Dept of Computer Sci & Creative Tech		
Module type:	Standard		
Pre-requisites	Introduction to OO Systems Development 2019-20		
Excluded Combinations	None		
Co- requisites	None		
Module Entry requirements	None		

Part 2: Description

Educational Aims: See Learning Outcomes

Outline Syllabus: 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 and waterfall methods e.g. agile (the agile manifesto), reuse and integration, test-driven development.

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: an UML modelling tool (e.g. ArgoUML) with cognitive assistance wizards. A java-based Interactive Development Environment (IDE) e.g. NetBeans, providing application projects, packages, classes, build scripts, deployment, execution, debugging, testing (e.g. JUnit), 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: File input / output. Java Database Connectivity (JDBC), serialisation, introductory database 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.

Distributed systems – network architectures and protocols, Sockets, Datagram and RMI.

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

Teaching and Learning Methods: Contact time: 72 hours

Assimilation and development of knowledge: 148 hours

Exam preparation: 40 hours

Coursework preparation: 40 hours

Total study time: 300 hours

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 Blackboard Virtual Learning Environment (VLE). All lecture slides, recommended articles, videos and tutorial notes are available on the Blackboard 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 Blackboard VLE offers podcasts and presentations for students to interact with at their own pace. Also, high quality, robust java-based Open Source software tools (e.g. ArgoUML and NetBeans) 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 Blackboard 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.

Part 3: Assessment

The assessment strategy for this module is a combination of written examination and coursework assignment. The written examination is of three hours duration and comprises questions to examine cognate and practical skills via a range of essay, multi-choice questions (MCQs), and appropriate analysis and design technique exercises. Where appropriate, partial UML diagrams, source code fragments or partial text cases may be provided as the basis for the examination question.

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; use 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 an in-class 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.

First Sit Components	**Final Assessment**	**Element weighting**	**Description**
In-class test - Component B		50 %	A group coursework software design and development assignment – (submitted on-line). Assessment by an in-class demonstration.
Examination - Component A	✓	50 %	Written Examination (3 hrs)
Resit Components	**Final Assessment**	**Element weighting**	**Description**
Report - Component B		50 %	Design and Implementation of a software system. Submitted as a report with supporting software. The report should detail about how the development would be achieved when working in a team of developers. This should be incorporated into the agile development section of the document.
Examination - Component A	✓	50 %	Written Exam (3 hrs)

Part 4: Teaching and Learning Methods

Learning Outcomes

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

Module Learning Outcomes	**Reference**
Describe the theory and concepts of the object-oriented paradigm	MO1
Explain the essential characteristics of the object-oriented software systems development life cycle, including testing	MO2
Apply object-oriented analysis and design techniques for a number of problem domains scoped at level 2 complexity	MO3
Understand and use a Unified Modelling Language (UML) modelling tool and a Java-based Interactive Development Environment (IDE) to develop object oriented software implementations appropriate to level 2 complexity	MO4
At an introductory level, apply object-oriented software patterns and architectures for increasingly large scale distributed software systems	MO5
Use an appropriate development environment to design and implement persistence within a distributed architecture at an introductory level	MO6
Design and implement concurrent systems and distributed systems	MO7
Design and implement Graphical User Interfaces (GUIs)	MO8

	Develop the necessary professional skills considering Agile approach while working in a group.	MO9

Contact Hours		
	Independent Study Hours:	
	Independent study/self-guided study	228
	Total Independent Study Hours:	228
	Scheduled Learning and Teaching Hours:	
	Face-to-face learning	72
	Total Scheduled Learning and Teaching Hours:	72
	Hours to be allocated	300
	Allocated Hours	300
Reading List	*The reading list for this module can be accessed via the following link:* https://uwe.rl.talis.com/modules/ufcfb6-30-2.html	

Part 5: Contributes Towards

This module contributes towards the following programmes of study:

Software Engineering for Business [Sep][SW][Frenchay][4yrs] BSc (Hons) 2018-19
Computer Science [Sep][SW][Frenchay][4yrs] BSc (Hons) 2018-19
Software Engineering [Sep][SW][Frenchay][4yrs] BSc (Hons) 2018-19
Software Engineering [Jan][FT][Northshore][3yrs] BSc (Hons) 2018-19
Software Engineering {Dual} [Aug][FT][Taylors][3yrs] BSc (Hons) 2018-19
Software Engineering {Dual} [Mar][FT][Taylors][3yrs] BSc (Hons) 2018-19
Software Engineering [Sep][FT][Frenchay][3yrs] BSc (Hons) 2018-19
Computer Science [May][FT][Villa][3yrs] BSc (Hons) 2018-19
Computer Science [Jan][FT][Villa][3yrs] BSc (Hons) 2018-19
Computer Science [Sep][FT][Villa][3yrs] BSc (Hons) 2018-19
Computer Science [Sep][FT][Frenchay][3yrs] BSc (Hons) 2018-19
Software Engineering for Business [Sep][FT][Frenchay][3yrs] BSc (Hons) 2018-19
Business Computing [Sep][FT][Frenchay][3yrs] BSc (Hons) 2018-19
Business Computing [Sep][SW][Frenchay][4yrs] BSc (Hons) 2018-19