Module Specification

Enterprise Systems Development

Version: 2021-22, v3.0, 01 Jul 2021

Contents

Part 1: Information

Module title: Enterprise Systems Development

Module code: UFCF85-30-3

Level: Level 6

For implementation from: 2021-22

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: Frenchay Campus, Global College of Engineering and Technology (GCET), Northshore College of Business and Technology, Taylors University, Villa College

Field: Computer Science and Creative Technologies

Module type: Standard

Pre-requisites: Object-Oriented Systems Development 2021-22

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 syllabus includes:

Enterprise-scale software systems development:
Enterprises of planetary scale and complexity, virtualisation. Software systems by acquisition, integration, configuration and customisation, and subsequent interoperability. The economics of various software acquisition strategies, from DIY to components-of-the-shelf (COTS), to software packages, to service-based capabilities in the Cloud.

Theory and concepts of components, interfaces and services:
Specification, test specification, implementation and deployment of components and services, and their various modelling notations, including the Unified Modelling Language (UML).

Developmental processes:
Agile versus plan-based, impact of scale and complexity on the process. Challenges to agile, test-driven and incremental approaches for contemporary software development. Issues associated with planetary scale stakeholder analysis, requirements capture, virtualisation and testing. Processes for systems development by software reuse – software discovery and evaluation, acquisition, integration, customisation and configuration.

Architectures and patterns:
The notion of software architecture and its significance to ESD, separation of concerns (‘divide and conquer’) in global scale software systems via architectures and patterns. Logical versus physical focus i.e. business-driven, logical separation of concerns via cohesive groupings of components and services minimising coupling dependencies, versus the technology capabilities provided by large scale technical platform infrastructures. The application of architectures and patterns in relation to analysis and design techniques for enterprise-scale software systems development, including security aspects.

Enterprise scale development frameworks:
Review of the state-of-the-art frameworks for ESD (e.g. comparison of Microsoft

.NET, Java Enterprise Edition, Spring, etc). Analysis, design and development of multi-tier, distributed web-based applications by the reuse, configuration, customisation and deployment of framework components and services, e.g. servlets and JSPs. Also persistence via design, implementation and use of Java Database Connectivity (JDBC) and relational databases, and security aspects.

Computing paradigms and models for ESD with emphasis on Cloud Computing: Its significance (competing) definitions, available capabilities and contemporary technologies. Current example applications. Benefits and drawbacks, especially in relation to the economics and risks of utility computing. Drift of applications and data from localised processing to virtual environments, and likely consequences. State-of-the-art research findings on the potential of Cloud Computing, e.g. via multi-channel, asynchronous and adaptive 'systems of systems', to serve mankind in the future.

Part 3: Teaching and learning methods

Teaching and learning methods: Scheduled learning includes interactive lectures and tutorials, wherein the state-ofthe-art of enterprise systems development is demonstrated, discussed and critically evaluated. At lectures, questions from students are proactively encouraged and freely discussed. Questions from the lecturer are prominent at the start of lectures to clearly establish the learning context and obtain the undivided focus of the student cohort. Further questions and answers are initiated during the lecture. As appropriate for Level 3 students, discussions at the end of lectures promote formative feedback, evaluation and deep reflection upon the learning outcomes of the lecture. Audio recordings of the lecture and interactive discussions are taken and made available via the Blackboard Virtual Learning Environment (VLE). Indeed, all lecture slides, recommended articles (both research and trade), videos, URLs and tutorial notes are available on the Blackboard VLE.

At tutorials, students are encouraged to attempt scoped activities (e.g. problem analysis and solving, appraisal, design, implementation, and validation) and then articulate and present their findings to their peers. Activities relate to realistic case studies of Level 3 complexity, and may also include the production of implementation

artefacts such a deployment descriptors, execution log trails, source code implementation, etc. Interactive peer-review (directed by the tutor) is essential for student reflection to achieve the deep learning appropriate for Level 3. From time to time, students also participate in “pub-quiz”-like sessions in tutorials (For example, MCQs are provided to small groups of students, enabling immediate and rich formative feedback on the level and extent of student knowledge and understanding at that point in time). Not sure that we leave it as example or to be definitive.

Independent learning includes hours engaged with essential reading, case study preparation, assignment preparation and completion, etc. Explicit guidance is given to students with respect to the sources of information used in self-study. Library resources such as books, research and trade articles are essential to supplement lectures, and are made available via the Blackboard VLE. Use of library search engines is encouraged. In addition, high quality, robust java-based Open Source modelling and development 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 of faculty workstations and student laptops, when taken together with the Blackboard VLE, enables great interoperability with respect to development artefacts, promoting virtualisation of learning location. The learning achieved from self-study (is) are then brought forward by students to be reinforced at the interactive tutorials wherein their knowledge and understanding are deepened by directed articulation, presentation and critical appraisal with their peers and tutor.

Contact Hours:

Activity:
Contact time: 72 hours
Assimilation and development of knowledge: 156 hours
Exam preparation: 72 hours
Total study time: 300 hours

Module Learning outcomes:

MO1 Describe the essential characteristics of enterprise-scale software systems and their development

MO2 Show a detailed knowledge of software development process models for enterprise-scale software systems development, including agile

MO3 Understand the need for developmental frameworks for developing enterprise systems

MO4 Apply state-of-the-art developmental frameworks to team-based design and development of web-based applications, including security aspects

MO5 Explain the theory and concepts of components, interfaces and services and their various modelling notations

MO6 Discuss in detail the application of the notion of software architecture and software patterns in relation to analysis, design and security techniques for enterprise-scale software systems development

MO7 Understand computing paradigms and associated models for enterprise systems e.g. Cloud Computing and its potential

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/modules/ufcf85-30-3.html

Part 4: Assessment

Assessment strategy: The assessment strategy for this module comprises a written examination and a coursework assignment. Examination questions examine cognate and practical skills via a range of essay, multi-choice questions (MCQs), and

appropriate analysis and design technique exercises. Where appropriate, architectural diagrams, UML diagrams, source code fragments or partial text cases may be provided as the basis for the examination question.

The coursework assignment comprised two elements. The first element is group-based and consists of a demonstration of an appropriately scoped software application developed by the student team using enterprise development frameworks, architectures and patterns.The second element is an individual written assignment in which students conduct a critical review of the modelling framework they have chosen for their software systems modelling of a case study.

Resit opportunities are provided for both examination and coursework assignment. The resit examination is similar in format and approach to the summer examination. The resit coursework assignment assesses the same learning outcomes as previously but by means of an individual written assignment, which allows students to critically reflect on the feedback provided with previous assignments.

Assessment components:

Examination (Online) - Component A (First Sit)
Description: Online Examination (2 hours) 24-hour window
Weighting: 30 %
Final assessment: Yes
Group work: No
Learning outcomes tested: MO1, MO2, MO3, MO5, MO6, MO7

Presentation - Component B (First Sit)
Description: Group-based demonstration of software development
Weighting: 42 %
Final assessment: No
Group work: Yes
Learning outcomes tested: MO3, MO4

Written Assignment - Component B (First Sit)

Description: Individual written assignment (1000 words)
Weighting: 28 %
Final assessment: No
Group work: No
Learning outcomes tested: MO3, MO4

Examination (Online) - Component A (Resit)
Description: Online Examination (2 hours) 24-hour window
Weighting: 30 %
Final assessment: Yes
Group work: No
Learning outcomes tested:

Written Assignment - Component B (Resit)
Description: Individual written assignment (2000 words)
Weighting: 70 %
Final assessment: No
Group work: No
Learning outcomes tested:

Part 5: Contributes towards

This module contributes towards the following programmes of study:

Software Engineering [Sep][FT][Frenchay][3yrs] BSc (Hons) 2019-20

Software Engineering {Dual} [Aug][FT][Taylors][3yrs] BSc (Hons) 2019-20

Software Engineering {Dual} [Mar][FT][Taylors][3yrs] BSc (Hons) 2019-20

Software Engineering [Jan][FT][Northshore][3yrs] BSc (Hons) 2019-20

Computer Science [Jan][FT][Villa][3yrs] BSc (Hons) 2019-20

Computer Science [May][FT][Villa][3yrs] BSc (Hons) 2019-20

Computer Science [Sep][FT][Frenchay][3yrs] BSc (Hons) 2019-20

Computer Science [Sep][FT][Villa][3yrs] BSc (Hons) 2019-20

Software Engineering {Foundation} [Oct][FT][GCET][4yrs] BEng (Hons) 2018-19

Computer Science [Sep][SW][Frenchay][4yrs] BSc (Hons) 2018-19

Software Engineering [Sep][SW][Frenchay][4yrs] BSc (Hons) 2018-19

Computer Science {Foundation} [Sep][FT][Frenchay][4yrs] BSc (Hons) 2018-19

Software Engineering {Foundation} [Feb][FT][GCET][4yrs] BEng (Hons) 2018-19