1. **Title of the module**

COMP5100 (CO510) - Software Engineering

1. **School or partner institution which will be responsible for management of the module**

School of Computing

1. **The level of the module (Level 4, Level 5, Level 6 or Level 7)**

Level 5

1. **The number of credits and the ECTS value which the module represents**

30 credits (15 ECTS*)*

1. **Which term(s) the module is to be taught in (or other teaching pattern)**

Autumn and Spring

1. **Prerequisite and co-requisite modules**

Pre-requisite: COMP3340: People and Computing

COMP3200: Introduction to Object-Oriented Programming

COMP5200: Further Object-Oriented Programming

1. **The programmes of study to which the module contributes**

Computer Science, CS(AI), CS(Networks), CS(Consultancy)

Applied Computing joint honours programmes,

Computing and Business Administration,

Web Computing

Plus year in industry variants of these programmes.

1. **The intended subject specific learning outcomes.  
   On successfully completing the module students will be able to:**

8.1 Understand the principles and practice of the development of software systems (broadly defined) – from requirements specification, design, validation, implementation, and evolution [A2, A4];

8.2 Apply design principles and patterns while developing software systems [B5, C1];

8.3 Create UML diagrams for modelling aspects of the domain and the software [A2, B1];

8.4 Design and implement test plans, and apply a wide variety of testing techniques effectively and efficiently [B4, C2];

8.5 Demonstrate the vital role of planning, documentation, estimation, quality, time, cost and risk evaluation in the business context [B1];

8.6 Show an understanding of system design, including, design simplicity, appropriateness, and styles of system thinking and focused problem solving [C1, B3, B4, B5];

8.7 Show an understanding of the professional and legal duties software engineers owe to their employers, employees, customers and the wider public [B6];

8.8 Use the appropriate tools and techniques when working in groups [C4, D1, D5].

1. **The intended generic learning outcomes.  
   On successfully completing the module students will be able to:**

9.1 Take a professional approach to software development [B6];

9.2 Enhance their ability to work successfully in a group [A2, D1].

9.3 Effectively use IT facilities [D3];

9.4 Manage their own learning and time [D5];

9.5 Be able to analyse a problem specification and to design and implement a solution [B3, B4, D3].

1. **A synopsis of the curriculum**

This module provides an introduction to basic design principles of systems, including modelling principles and the use of tools, and design patterns. It also looks into different software processes, and introduces software testing. Regarding software project management, topics All the issues cover in the module will form the basis of the group project, which entails the design, implementation and evaluation of a simple software system.

This module provides an introduction to basic design principles of systems, including modelling principles and the use of tools, and design patterns. It also looks into different software processes, and introduces software testing. Regarding software project management, topics like risk management, quality assurances are covered. Under professional practice the module covers codes of ethics and professional conduct. All the issues cover in the module will form the basis of the group project, which entails the design, implementation and evaluation of a simple software system.

1. **Reading list (Indicative list, current at time of publication. Reading lists will be published annually)**

K. Beck. Extreme Programming Explained: Embrace Change. Addison Wesley. Upper Saddle River, NJ, USA. 2000.

G. Booch, J. Rumbaugh, I. Jacobson. The Unified Modeling Language Users Guide. Addison Wesley. 1999

G. Booch, J. Rumbaugh, I. Jacobson. The Unified Software Development Process. Addison Wesley. 1999.

P. Coad, E. Lefebvre, J. De Luca. JAVA Modeling in Color with UML: Enterprise Components and Process. Prentice Hall. 1999.

A. Cockburn. Writing Effective Use Cases. Addison-Wesley. Boston, Ma, USA. 2001.

E. M. Hall. Managing Risk: Methods for Software Systems Development. Addison-Wesley. Reading, MA, USA. 1998.

D. G. Johnson, H. Nissenbaum. Computers, Ethics and Social Values. Prentice-Hall. 1995

E. A. Kallman, J. P. Grillo. Ethical Decision Making and Information Technology: An Introduction with Cases. 3rd Edition. McGraw-Hill. 1999

D. Kulak, E. Guiney. Use Cases: Requirements in Context. Addison-Wesley. Boston, Ma, USA. 2000.

J. Newkirk, R. C. Martin. Extreme Programming in Practice. Addison Wesley. Upper Saddle River, NJ, USA. 2001.

Mauro Pezze, Michal Young. Software Testing and Analysis: Process, Principles and Techniques. John Wiley & Sons. 2007.

R. Pooley, P. Stevens. Using UML Software Engineering with Objects and Components. Addison-Wesley. 2001.

G. Schneider, J. P. Winters. Applying Use Cases: A Practical Guide. Addison-Wesley. 2001.

I. Sommerville. Software Engineering.9th Edition. Harlow, England, UK. 2010.

1. **Learning and teaching methods**

Total contact hours: 70

Private study hours: 230

Total study hours: 300

1. **Assessment methods**
   1. Main assessment methods

3-stage modelling portfolio – 10%

5-stage development in groups – 40%

Examination (2 hours) – 50%

13.2 Reassessment methods

Reassessment Instrument: Like for like.

1. **Map of module learning outcomes (sections 8 & 9) to learning and teaching methods (section12) and methods of assessment (section 13)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Module learning outcome** | *8.1* | *8.2* | *8.3* | *8.4* | *8.5* | *8.6* | *8.7* | *8.8* | *9.1* | *9.2* | *9.3* | *9.4* | 9.5 |
| **Learning/ teaching method** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Lectures/Classes | X | X | X | X | X | X | X | X | X | X | X | X | X |
| Private Study | X | X | X | X | X | X | *X* |  | X | X | X | X | X |
| **Assessment method** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Portfolio | X | X | X |  |  | X |  |  | X |  | X | X | X |
| Development | X | X | X | X | X | X | *X* | X | X | X | X | X | X |
| Exam | X | X | X | X | X | X | *X* |  | X |  |  |  | X |

1. **Inclusive module design**

The School recognises and has embedded the expectations of current equality legislation, by ensuring that the module is as accessible as possible by design. Additional alternative arrangements for students with Inclusive Learning Plans (ILPs)/declared disabilities will be made on an individual basis, in consultation with the relevant policies and support services.

The inclusive practices in the guidance (see Annex B Appendix A) have been considered in order to support all students in the following areas:

a) Accessible resources and curriculum

b) Learning, teaching and assessment methods

1. **Campus(es) or centre(s) where module will be delivered**

Canterbury

1. **Internationalisation**

The topics addressed by this module relate to a field which is of international importance, given the global role of computers in today's technological innovation. The topics covered by this module are international in nature, being identical worldwide and independent of traditional spoken language.

**FACULTIES SUPPORT OFFICE USE ONLY**

**Revision record – all revisions must be recorded in the grid and full details of the change retained in the appropriate committee records.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date approved | Major/minor revision | Start date of the delivery of revised version | Section revised | Impacts PLOs (Q6&7 cover sheet) |
|  |  |  |  |  |
|  |  |  |  |  |

Revised FSO Jan 2018