1. **Title of the module**

COMP6610 (CO661) – Theory and Practice of Concurrency

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

Division of Computing, Engineering, Mathematical Sciences (CEMS)

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

Level 6

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

15 credits (7.5 ECTS)

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

Autumn or Spring

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

Pre-requisite: COMP5450 Functional and Concurrent Programming

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

BSC Computer Science (all variants) including year in industry

1. **The intended subject specific learning outcomes.  
   On successfully completing the module students will be able to:**
   1. Have a critical understanding of the principles of concurrent programming, as well as its advantages and challenges;
   2. Reason on the properties of a distributed process (e.g., safety and liveness), and compare the behaviour of different processes.
   3. Design and implement processes satisfying given properties.
   4. Apply the acquired knowledge to real scenarios e.g. application-level protocols, Web services.
   5. Be familiar with advanced concepts of Web Services.
2. **The intended generic learning outcomes.  
   On successfully completing the module students will be able to demonstrate:**
   1. Systematic and rigorous reasoning,
   2. Application of abstract concepts to concrete scenarios,
   3. Ability of presenting and discussing state of the art topics.
3. **A synopsis of the curriculum**

This module is aimed at introducing the principles of concurrency theory (1, 2, 3) and demonstrating how these can be applied to design and implement distributed applications (4). Advanced concepts of Web services will be studied and placed in the perspective of these principles (5, 6).

The following is an indicative list of topics:

* Message passing primitives for concurrency: synchronous versus asynchronous message passing, the actor model.
* Reasoning on processes: temporal logic, safety and liveness properties, bisimulation.
* Channel passing and mobility.
* Design and implementation of application–level protocols.
* Web services: from stateless services to distributed business processes (also known as service orchestrations).
* Transaction protocols on the Web: two-phase commit, long running transactions.

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

Armstrong, J, Virding, R, Williams,W.M, Wilkstrom, (1996). *C. Concurrent Programming in ERLANG*. Prentice-Hall.  
Lynch, N.A. (1996). *Distributed Algorithms*. Morgan Kaufmann Publishers In (Section 7)

Milner, R. (1989). *Communication and Concurrency*. Prentice-Hall,

1. **Learning and teaching methods**

Total contact hours: 40 hours

Private study hours: 110 hours

Total study hours: 150 hours

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

Concurrent programming in Java Coding assignment (20%)

Concurrent programming in GO Coding assignment (20%)

Seminar (Presentation and oral discussion) 10%

2-hour unseen written examination (50%)

13.2 Reassessment methods

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* | *9.1* | *9.2* | *9.3* |
| **Learning/ teaching method** |  |  |  |  |  |  |  |  |
| **Private Study** | **X** | **X** | **X** | **X** | **X** | **X** | **X** |  |
| Lectures | **X** | **X** |  |  | **X** | **X** | **X** |  |
| Supervised terminal classes |  | **X** | **X** |  |  | **X** | **X** |  |
| Student seminars |  |  |  | **X** |  |  |  | **X** |
| **Assessment method** |  |  |  |  |  |  |  |  |
| *Assignment* | **X** | **X** | **X** | **X** |  | **X** | **X** |  |
| *Presentation* | **X** |  |  | **X** | **X** |  |  | **X** |
| *Examination* |  | **X** | **X** |  | **X** | **X** |  |  |

1. **Inclusive module design**

The Division 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.

**DIVISIONAL 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 delivery of revised version | Section revised | Impacts PLOs (Q6&7 cover sheet) |
| 10/12/2020 | Major | September 2021 | 6,7 | No |
|  |  |  |  |  |