**Please refer to the AM4 Guidance Notes as you complete this descriptor.**

**Part One: Module Leader’s section: core module details**

**1. Module Title: Fundamentals of Parallel Systems**

**2. SCQF Level: Level 9**

**3. SCQF Credit Value: 20 4. ECTS Credit Value: 10**

**5. Module Code: SET09109** [Admin Use Only]

**6. Module Leader: Jon Kerridge**

**7. School: Computing**

**8. Edinburgh Napier Subject Group Area: Software Engineering**

|  |  |
| --- | --- |
| **9. Prerequisites –** To study this module you will need the learning equivalent to the module listed or have passed this module | |
| Is a Prerequisite Required? | Yes  No |
| Module Code | SET08102 |
| Module Title | Software Development 2 |
| Examples of Equivalent Learning | HNC Software Development |

**10. What you will learn and what this module is about**

Parallel systems are becoming ever more prevalent for games engineering, real-time systems’ control and as multi-core processors become commonplace in everyday computing equipment. This module introduces the fundamental concepts of parallel systems. Groovy helper classes are used as a means of learning the underlying principles more quickly and easily.

Starting from four basic concepts, you will develop solutions to increasingly complex problems by adopting a parallel design paradigm. The principal design pattern, client-server, provides a means of ensuring that one can reason about the behaviour of parallel systems. . The module then develops solutions to problems that involve distributed and networked systems as well as multi-core algorithms.

The module uses the Eclipse IDE with Java, Groovy and a library (JCSP) that implements the parallel concepts directly. All the software is available from the Moodle web site for the module and can be downloaded onto students’ own machines.

**11. Description of Module Content**

The basics: Processes, Channels, Alternative Selection and Timers.

Parallel Composition: Build it like Lego

Deadlock and Livelock avoidance: The Client-Server design pattern.

Managing Real-time Events: Parallel Graphical User Interfaces; controlling Lego robots

Shared Memory Synchronisation Primitives: barriers, buckets and CREW

Using net architectures to implement solutions in multi-core and distributed systems

Modify an existing multi-player on-line game so that players are better managed in the interactions within the game.

**12. Learning Outcomes of the module**

Upon completion of this module you will be able to

LO1: Apply the concepts of Process, Parallel Alternative, Channel and Timer to a system design

LO2: Analyse a solution to a problem for deadlock and livelock

LO3: Apply the concepts to the design of multi-core and distributed parallel systems

LO4: Evaluate the benefit of different approaches to parallelisation

**13: Indicative References and Reading List**

T1: Papers from [www.wotug.org](http://www.wotug.org)

T2: Using Concurrency and Parallelism Effectively by Jon Kerridge; www.bookboon.com available as an e-book for free or small charge first published 2014

T3: Concurrent Programming in Java: Design Principles and Patterns by [Doug Lea](http://gee.cs.oswego.edu/dl/) (ISBN 0-201-31009-0). Second edition published by [Addison-Wesley](http://www.awl.com/), November 1999, with supporting material available at <http://gee.cs.oswego.edu/dl/cpj/>

T4: Communicating Sequential Processes, CAR Hoare CACM 1978,21,8,666-677 (copy on Moodle)

**\*16. LTA Approach**

**Learning & Teaching methods including their alignment to LOs**

Lectures will introduce the basic concepts, mostly by means of practical examples which will then be used to explain the underpinning theory based upon Hoare’s CSP (LO1, 2 & 4) Specific parallel patterns will also be introduced and students will be expected to evaluate these through practical examples (LOs 1, 2 & 3).

The practical component of the module will allow students to develop solutions to typical programming problems from embedded systems and user interfaces (LOs 1, 2 & 3). They will also modify a solution to a multi-player on-line game.

**Embedding of employability/PDP/Scholarship Skills**

This module introduces students to parallel programming and in particular to the real-time handling of multiple events from control systems.

**Assessment (formative or summative)**

Assessment comprises a series of practical exercises accompanied by a log book (LOs 1, 2, 3 & 4) and a formal examination (LOs 1, 2 & 4). Students will be expected to complete a log book of their solutions to exercises contained in T2, which will be discussed during practical classes and also a solution to a specific problem created for the module.

**Research / teaching linkages**

The Module Leader is an active researcher in the area of parallel systems. The module will utilise recently published papers in the topic and the students will be encouraged to apply the theory in those papers to the evaluation of the practical examples used throughout the module delivery. The text T2 contains chapters based on published research papers.

**Supporting equality and diversity**

Online learning materials and resources are available to support inclusiveness and accommodate students from a wide variety of backgrounds. By encouraging supported self-study the module has flexibility that allows students to develop their skills at a pace and time appropriate to their prior experience and individual circumstances.

**Internationalisation**

The underpinning theory based on Hoare’s CSP is fundamental to this aspect of computer science. The software tools including Groovy, Java and the Eclipse IDE are used globally and Java programming language is an international standard.

**\*17. Student Activity (NESH)**

|  |  |  |
| --- | --- | --- |
| **Mode of Activity** | **L & T Activity** | **NESH** |
| Face to Face | Lecture | 24 |
| Face to Face | Practical Labs | 24 |
| Assessment | Formal exam | 2 |
| Independent Learning | Individual learning activities | 150 |
| **TOTAL NESH**  **To calculate total, please select the whole table and press F9 key** | | **200** |

**\*18. Assessment**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Week Due** | **Type of Assessment** | **Weighting (%)** | **LOs**  **Covered** | **Length/**  **Volume** |
|  | **Component: Assessment One**  Enter assessment element(s): |  |  |  |
| 11 | Practical Assessment  Other: | 50 | 1,2,3,4 | 50 Hours |
|  | **Component Subtotal**  (will auto-calculate –  see module total box below) | **50%** |  |  |
|  | **Component: Assessment Two**  Enter assessment element(s): |  |  |  |
| 14/15 | Centrally Timetabled Exam  Other: | 50 | 1,2,4 | 2 Hours |
|  | **Component Subtotal**  (will auto-calculate –  see module total box below) | **50%** |  |  |
|  | **Module Total**  **To calculate the total fields, select the whole table and press F9 key** | **100%** |  |  |