Module Code:	**CIS 2343**
Module Title:	**OBJECT-ORIENTED SYSTEMS DEVELOPMENT**

Scheme:	Undergraduate Computing
Module Rating:	Intermediate, 20 credits
Delivery Method:	**Lectures:** 24 hrs
	Tutorials/Practicals: 24 hrs
	Directed Unsupervised Activity: 152 hrs
School(s) involved:	Computing and Mathematics
Module author(s):	Anne Grundy
Prerequisites:	None
Recommended prior learning:	Competence in a procedural programming language and exposure to object-encapsulation concepts.
Co-requisites:	None
Barred Combinations:	None
Professional Body Requirements:	None
Module Status:	Dedicated
	Core (BScCSD, BScSD with …, MEng, BScCompSci)

Module Aims:

1. To define the paradigm of object orientation as it may be applied to the specification and implementation of software systems.
2. To equip the learner with an awareness of and skills in the application of techniques for the development of object-oriented system descriptions and specifications.
3. To acquire skills in a programming language which enables object-oriented software implementation.

Module Synopsis:

Modern programming languages include features geared towards the production of re-usable software built out of separate software components. Modern analysis techniques view systems as the interaction between ‘objects’ of different types. Object-oriented design seeks to specify reusable software components, which implement the desired behaviour of object-based systems.

This module examines how object-oriented analysis, design and implementation may be blended together.

Learning Outcomes:

1. Knowledge Outcomes
   On completion of this module the learner will be able to:
   1.1 Explain the concepts embodied in the object paradigm.
   1.2 Discuss the nature of the relationships between objects within a system from a variety of perspectives.
   1.3 Understand the aims, scope and core semantics of an object modelling language.
   1.4 Describe informally the semantics of a programming language, which enables the writing of object based software.
   1.5 Evaluate the impact of the object-based approach on the software development process.

2. Ability Outcomes
   On completion of this module the learner will be able to:-
   2.1 Produce object-based specifications for proposed software systems.
   2.2 Implement object-based specifications in software using appropriate programming language constructs.

Outline syllabus of topics to be covered:

- Software systems as interacting objects - modelling perspectives.
- Object modelling languages - semantics and visualisation, e.g. the Unified Modelling Language (UML).
- Object class relationships, object states and transitions, object interaction and collaboration.
- Interface objects and system event handling.
- Object-based software development methodologies - stages, activities and deliverables.
- Mapping of object-based system models onto software implementation components - concerns and techniques - design patterns.
- Object-oriented programming languages - semantics for object class implementation - encapsulation, information hiding, inheritance, polymorphism, genericity.
- Active objects in concurrent system modelling and implementation.
- Storage and retrieval of persistent objects, object request brokers.
- Object-based software quality issues - notion of contract, responsibilities of client and server, validation and verification of behaviour, invariance, type substitutability, metrics.
- Software reuse - identification of reusable components - tools and techniques.

Indicative learning strategy:
Topics will largely be introduced in the lecture programme. It is recommended that the introduction and development of software implementation skills should take place in parallel with the exposition of object-modelling semantics, languages and techniques. The tutorial/practical programme will be used for exercises to clarify lecture materials, study of worked examples and support time for courseworks. Some laboratory support is needed for initial introduction to the software implementation environment and language.

Indicative references/learning materials:

Eliens	Principles of Object-Oriented Software Development Addison-Wesley, 1994
Fowler	UML Distilled Addison-Wesley, 1998
Quatrani	Visual Modelling with Rational Rose and UML Addison-Wesley, 1998

Resources required:
Object-based software implementation environment for implementation of object classes and clients e.g. C++ or JAVA.
CASE tool support for object modelling e.g. Select Enterprise or Rational Rose.
For a maximum of 3 hours per week per student (1 hour supervised, 2 hours unsupervised).

Assessment Strategy:
This module will be assessed by one coursework and a two-hour time-constrained examination.
Both elements of assessment are equally weighted.
Assessed coursework: Object-oriented analysis of a problem scenario, followed by design and implementation of an object-based software solution, with consideration of validation and verification issues. Assesses learning outcomes 2.1, 2.2 (ability outcomes)

Time-constrained examination: Assesses learning outcomes 1.1, 1.2, 1.3, 1.4, 1.5 (knowledge outcomes)