### CO2006 Software Engineering and System Development

**Credits:** 20  **Convenor:** tbc  **Semester:** 1st

<table>
<thead>
<tr>
<th>Prerequisites:</th>
<th>Essential: CO1003, CO1005, CO1019</th>
<th>Desirable: CO1001, CO1012</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assessment:</td>
<td>Coursework: 40%</td>
<td>Three hour exam in January: 60%</td>
</tr>
<tr>
<td>Lectures:</td>
<td>30 hours</td>
<td>Problem Classes: 0 hours</td>
</tr>
<tr>
<td>Surgeries:</td>
<td>10 hours</td>
<td>Private Study: 90 hours</td>
</tr>
<tr>
<td>Laboratories:</td>
<td>20 hours</td>
<td></td>
</tr>
</tbody>
</table>

#### Subject Knowledge

**Aims**  
According to a report of British Computer Society, only about 16% of IT projects can be considered truly successful and over 60% experience severe problems. The difficulties of software development led to the coining of the phrase “the software crisis” and the birth of software engineering as a discipline. However, in many companies, software is still developed in an ad hoc way. The purpose of this module is to teach object-oriented methods for analysis, specification, design and implementation of software systems.

**Learning Outcomes**  
At the end of this course, students should be able to:

- Specify customer requirements in a structured requirements document
- Analyze customer requirements and produce an object-oriented system design
- Use UML for consistent specification of software systems and business processes
- Implement and test the system
- Use appropriate tools for software development and testing
- Understand the software development processes

**Methods**  
Lecture notes, class sessions, recommended textbooks, work-sheets, supervised laboratories, feedback from markers and extensive web support.

**Assessment**  
Programming tests, small implementation projects, problem-based worksheets, lab tests, examination.

#### Skills

**Aims**  
The student should be able to analyse problems, formulate solution strategies and solve the problems. They will be also able to effectively work in a group and to communicate ideas to other group members.

**Learning Outcomes**  
Students will learn how to develop object-oriented software systems, how to plan the development and how to work in groups. This will involve requirements specification and analysis as well as system design, implementation and testing.

**Methods**  
Class sessions, individual and group work-sheets, feedback from module convenors.

**Assessment**  
Marked coursework, lab test.

#### Explanation of Prerequisites

A sound knowledge of basic algorithms, data structures and programming is required. An understanding of logic and discrete structures is important for rigorous specification of software systems. Some knowledge of database systems is desirable.

**Course Description**  
This module teaches engineering principles and methods needed to specify, design and implement a large system using object-oriented techniques.
Detailed Syllabus

Introduction: Software crisis and historical background of Software Engineering; software products and their characteristics: maintainability, dependability, efficiency and usability.

Introduction to OO Development: The inherent complexity of software; mastering complex systems; examples of complex systems; function oriented vs object-oriented methods.

Requirements specification: requirements documents; use cases and scenarios; model of problem domain; functional and non-functional requirements.

Object-oriented analysis: analysis class and sequence diagrams.

Object-oriented design: design class and sequence diagrams; object diagrams; state machines; design patterns; consistency of multi-view models; software architectures.

Implementation: mapping object-oriented designs to code; container/collection classes in code; testing and JUnit.

Reading List


Resources Course notes, module web page, study guide, worksheets, handouts, lecture rooms with two OHPs, past examination papers, past tests.

Module Evaluation Course questionnaires, course review.