Module Code  | CSU33D06
Module Name  | Software Design and Implementation.
ECTS Weighting¹ | 5 ECTS
Semester taught | Semester 2
Module Coordinator/s | Stephen Barrett

Module Learning Outcomes

On successful completion of this module, students will be able to:

LO1. Undertake software design and construction as members of teams of various sizes using modern methodologies.
LO2. Choose, learn, and use new languages, tools, and techniques
LO3. Define a problem specification and devise a solution.
LO4. Implement a software system of reasonable complexity.
LO5. Test the solution using standard techniques.
LO6. Present the solution to stakeholders.

Module Content

This module aims to inculcate practical skills in team driven software engineering through small and large group programming projects. It places students in the team setting to offer practical learning opportunity in how to work effectively with others throughout project lifecycles. Students conceive of, develop and present creative work in a competitive setting. Key practical skills include the day-to-day use of:

- Git source code management system.
- Team communication tools.
- Testing techniques for quality assurance.
- Agile development methodology.

Teaching and Learning Methods

Lectures and Development Lab sessions.

Assessment Details²

<table>
<thead>
<tr>
<th>Assessment Component</th>
<th>Brief Description</th>
<th>Learning Outcomes Addressed</th>
<th>% of total</th>
<th>Week set</th>
<th>Week due</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development Contribution</td>
<td>Students are assessed based on their practical development contribution to their team work.</td>
<td>All</td>
<td>100%</td>
<td>n/a</td>
<td>n/a</td>
</tr>
</tbody>
</table>

¹ TEP Glossary
² TEP Guidelines on Workload and Assessment
An individual practical development task will be set, to be completed by the end of the normal reassessment period. There is no reassessment by written examination.

### Contact Hours and Indicative Student Workload

<table>
<thead>
<tr>
<th>Contact Hours (scheduled hours per student over full module), broken down by:</th>
<th>44 hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>lecture</td>
<td>11 hours</td>
</tr>
<tr>
<td>laboratory</td>
<td>33 hours</td>
</tr>
<tr>
<td>tutorial or seminar</td>
<td>0 hours</td>
</tr>
<tr>
<td>other</td>
<td>0 hours</td>
</tr>
</tbody>
</table>

**Independent study (outside scheduled contact hours), broken down by:**

<table>
<thead>
<tr>
<th></th>
<th>72 hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>preparation for classes and review of material (including preparation for examination, if applicable)</td>
<td>6 hours</td>
</tr>
<tr>
<td>completion of assessments (including examination, if applicable)</td>
<td>66 hours</td>
</tr>
</tbody>
</table>

**Total Hours**

| | 116 hours |

### Recommended Reading List

Relevant material will be introduced from the Internet. References will be provided as the module proceeds.

### Module Pre-requisites

**Prerequisite modules:** none.

**Other/alternative non-module prerequisites:** Students should have programming experience in one or more modern languages and should have completed a personal or class programming project of reasonable scale (eg. Module programming assignment).

### Module Co-requisites

### Module Website

### Last Update

19/09/2019 by Stephen Barrett