<table>
<thead>
<tr>
<th>Module Code</th>
<th>CS3012</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module Name</td>
<td>Software Engineering</td>
</tr>
<tr>
<td>Module Short Title</td>
<td>N/a</td>
</tr>
<tr>
<td>ECTS weighting</td>
<td>5</td>
</tr>
<tr>
<td>Semester/term taught</td>
<td>First Semester</td>
</tr>
</tbody>
</table>
| Contact Hours | Lecture hours: 3  
Lab hours:  
Tutorial hours:  
Total hours: 3 |
| Module Personnel | Lecturing staff: Dr. Edgar Galvan-Lopez |
| Learning Outcomes | This course provides students with a solid grounding in various aspects related to building large, important software systems. The overall aim of this course is for students to learn the fundamental skills for building large, important software systems. This entails (i) to recognise the general software lifecycle and its stages from domain analysis to maintenance, (ii) to analyse software in the problem domain, (iii) to identify the fundamental approaches to managing software projects and teams, (iv) to distinguish the roles of stakeholders in a software project in general and in software teams in particular, (v) to recognise architectures for building large-scale distributed software systems. This course covers various aspects related to building software systems ranging from the use of software lifecycle models, to project management, to large-scale software architectures. Specifically, software lifecycle models, including variations of the waterfall and spiral models as well as extreme programming and agile, are introduced along with concepts that are relevant to the specific model stages. These concepts include UML-based O-O, and domain analysis, requirements and specification analysis, testing and debugging, and version control. Moreover, strategies for managing large software projects and their contracts as well as project teams are presented and contrasted. |
| Module Learning Aims | When students have successfully completed this module they will be able to:  
- explain and compare the key concepts of the waterfall, spiral, and agile/extreme programming lifecycle models;  
- apply strategies for the gathering of software requirements and the generation of software specification;  
- analyse software by means of a set of UML diagrams;  
- compare and apply techniques for managing software projects and teams;  
- explain the philosophy for designing architectures for software systems.  
- Introduction |
| Module Content | • Software lifecycles  
|                | • O-O Analysis and UML  
|                | • Domain Analysis  
|                | • Requirements and Specification  
|                | • Project Management  
|                | • Design Issues  
|                | • Small and Larger Scale Design  
|                | • Debugging and Testing  
| Module Pre Requisite | -  
| Module Co Requisite | -  
| Assessment Details | Exam: %75  
|                    | Coursework: %25  
|                    | In the supplemental examinations, assessment is by written examination only, which contributes 100% of the overall mark.  
| Module approval date | N/a  
| Approved By | N/a  
| Academic Start Year | N/a  
| Academic Year of Data | N/a  

Page 2 of 2