## Module Description 2013/14

School of Computer Science and Statistics.

<table>
<thead>
<tr>
<th>2</th>
<th>Module Code</th>
<th>CS4032</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Module Name</td>
<td>Distributed Systems</td>
</tr>
<tr>
<td></td>
<td>Module Short Title</td>
<td>N/a</td>
</tr>
<tr>
<td></td>
<td>ECTS weighting</td>
<td>5 ECTS</td>
</tr>
<tr>
<td></td>
<td>Semester/term taught</td>
<td>Michaelmas</td>
</tr>
<tr>
<td></td>
<td>Contact Hours</td>
<td>Lecture hours: 22  Lab hours: 0  Tutorial hours: 11  Total hours: 33</td>
</tr>
<tr>
<td></td>
<td>Module Personnel</td>
<td>Lecturing staff: Assistant Professor Stephen Barrett</td>
</tr>
</tbody>
</table>
|  | Learning Outcomes | When students have successfully completed this module they should be able to:  
- Describe the basic characteristics, structure and operation of a distributed system, and the issues which a distributed system poses to a systems architect;  
- Identify and evaluate appropriate architectural models for distributed problem scenarios;  
- Design, construct, document and test distributed system solutions to realistic real-world problems;  
- Reason about the performance trade-offs of decentralised architectures;  
- Make use of appropriate documentation and reference material. |
|  | Module Learning Aims | Building distributed applications is a difficult task due to the concurrency, communication latency, and possibility of partial failure that is inherent in distributed systems. As in other areas of computer science, the trend in providing support for building distributed applications has been towards presenting the application developer with ever higher levels of abstraction and, in the particular case of distributed programming, of location transparency. This course takes a critical look at some of the paradigms and architectural issues involved in distributed programming and their likely evolution.  
Students will be given opportunities to develop their problem solving, programming and written communication skills by designing solutions to distributed system programming problems, and implementing those solutions as fully networked distributed systems. |
Specific topics addressed in this module include:

- Network Programming
- Concurrency
- Representational State Transfer
- Remote Procedure Call, Remote Object Technology and Middleware
- Synchronization and coordination
- Distributed transactions
- Authentication
- Replication
- Peer-to-peer and other large scale architectures
- Enterprise Infrastructure and Paradigms – Web services
- Advanced network application frameworks

Key sources include


Additional material will be provided/noted during the module.

The students are expected to be able to be competent in at least one high level programming language (e.g. Python, Java, C++, C# etc.). Previous with concurrent programming is beneficial but concurrency will be reviewed in the module.

A mandatory individual project of significant scope is undertaken over the term which students must work on outside lecture and tutorial times. Continuous assessment is composed of a number of marked tutorial exercises and programming assignments. The final grade awarded will be a simple accumulation of grades achieved in each element.
<table>
<thead>
<tr>
<th>Academic Start Year</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Academic Year of Data</td>
<td>N/a</td>
</tr>
</tbody>
</table>