

Module Title	<b>Systematic Software Development</b>
Level	5
Reference No.	BIF-5-SSD
Credit Value	20
Student Study Hours	Total: 200 Contact hours: 65 Student managed learning hours: 135
Pre-requisite Learning	None
Co-requisites	None
Excluded combinations	None
Module co-ordinator	P Rosner
Faculty/Department	Informatics
Short Description	This module will deepen your understanding of software development processes and their associated products. You will learn how to develop software systematically from initial requirements through to acceptance testing, and how to achieve high quality in terms of reliability, robustness and maintainability. You will enhance your programming skills and develop new skills in the areas of program design utilising patterns, user interface design and implementation, and rigorous testing.
Aims	The module aims to extend your understanding of the issues involved in software development. It covers a range of good practices, many of which are reinforced by the practical work that you undertake. The emphasis is on developing high quality software through the embodiment of key principles such as the separation of concerns and the “keep it simple” maxim. You will also develop new skills in the area of GUI design and implementation as well as in systematic development and testing.
Learning Outcomes	<p><b>Knowledge and understanding.</b> After completing this module you should be able to:</p> <ul style="list-style-type: none"> <li>• Demonstrate an understanding of software development using a variety of software engineering techniques.</li> <li>• Critically evaluate the quality of a software artefact.</li> </ul> <p><b>Intellectual skills.</b> You should be able to:</p> <ul style="list-style-type: none"> <li>• Use sources of information to improve your knowledge and understanding.</li> </ul> <p><b>Practical skills.</b> You should be able to:</p> <ul style="list-style-type: none"> <li>• Demonstrate an understanding of using IDE’s to build and test software systems.</li> </ul> <p><b>Transferable skills.</b> You should be able to:</p> <ul style="list-style-type: none"> <li>• Effectively plan the development of a practical project from design to implementation</li> </ul>
Employability	Sound knowledge of the wide range of issues, principles and good practices that come under the umbrella of “software engineering” is to be expected of all professional software developers. The emphasis on visual application development within this module will also extend the students’ knowledge and expertise in this key area of software development, giving them hands-on experience.
Teaching and learning pattern	There will be two hours of lectures each week, typically introducing new concepts in the first hour and then explaining how to implement these concepts with software examples in the second hour. There will also be one three-hour lab session each week in which students will work on the development of software artefacts and carry out some paper-based exercises.
Indicative content	The content of the module will include: <ul style="list-style-type: none"> <li>- software development lifecycle models,</li> <li>- requirements engineering and prototyping,</li> </ul>

	<ul style="list-style-type: none"> <li>- expressing specifications graphically and textually</li> <li>- aspects of software quality and quality assurance techniques,</li> <li>- principles of testing,</li> <li>- software maintenance and evolution.</li> </ul> <p>A central aspect of the approach to the above topics will be how to construct software that embodies the engineering principles addressed.</p>
<p>Assessment <i>Elements &amp; weightings</i></p>	<p>The module will be assessed via a combination of coursework (60%) and exam (40%). The coursework will typically be divided into a series of assessments that will include initial specification of an interactive system and a subsequent engineered implementation of the system.</p>
<p>Indicative Sources <i>(Reading lists)</i></p>	<p>No single textbook has been found that is suitable as the core text for this module, however the syllabus can be covered by a combination of textbooks such as:</p> <p>Bell, D. (2005) <i>Software Engineering: A Programming Approach</i>. 4<sup>th</sup> ed. Addison-Wesley.</p> <p>Liang, D (2012) <i>Introduction to Java Programming</i>. 9<sup>th</sup> ed. Pearson</p> <p>Further background reading includes</p> <p>Koskela, L. (2013) <i>Effective Unit Testing, A Guide for Java Developers</i>, Manning</p> <p>Freeman, E. Robson, E. Bates, B. Sierra, K.(2004) <i>Head First Design Patterns</i>, O'Reilly Media</p> <p>Hunt, A, Thomas, D. (2002) <i>The Pragmatic Programmer</i>, Addison Wesley</p> <p>Leff, A. Rayfield, J. (2001), <i>Web Application Development using the Model View Controller Design Pattern</i>, IEEE Enterprise Distributed Object Computing Conference pp 116-127</p> <p>Fowler, M (2006) <i>GUI Architectures</i>,  <a href="http://martinfowler.com/eaDev/uiArchs.html">http://martinfowler.com/eaDev/uiArchs.html</a></p> <p>In addition to textbooks, students will be expected to refer to other web-based information sources as required.</p>