PROGRAMME SPECIFICATION

Programme title:	MSc Software Systems Engineering (SSE)
Final award (BSc, MA etc): (where stopping off points exist they should be detailed here and defined later in the document)	MSc
UCAS code: (where applicable)	N/A
Intake cohort(s) to which this programme specification is applicable: (e.g. from 2001 intake onwards)	From 2005 intake onwards
Awarding institution/body:	University College London
Teaching institution:	University College London
Faculty:	Engineering Sciences
Parent Department: (the department responsible for the administration of the programme)	Computer Science
Web page address: (if applicable)	http://www.cs.ucl.ac.uk/teaching/mscsse/index.htm
Method of study: Full-time/Part-time/Other	Full-time
Length of the programme: (please note any periods spent away from UCL, such as study abroad or placements in industry)	One calendar year
Level on Framework for Higher Education Qualifications (FHEQ) (see Guidance notes)	Master's level (MSc Award)
Relevant subject benchmark statement (SBS) (see Guidance notes)	Not applicable. There is currently no Master's level benchmark statement for this subject area.
Brief outline of the structure of the programme / its assessment: (see guidance notes)	Please see http://www.cs.ucl.ac.uk/teaching/mscsse/programme.htm
Board of Examiners:	Board of Examiners for the MSc in Software Systems Engineering
Professional body accreditation (if applicable):	BCS and IET Date of next scheduled accreditation visit: 2010/11

EDUCATIONAL AIMS OF THE PROGRAMME:

The programme will train students in the principles and techniques of engineering large, complex software systems and will give them an opportunity to apply their understanding of the principles and techniques in a realistic group project setting. The training will be at an intellectually demanding level and will cover not only the state-of-the practice in software systems engineering, but also the most significant trends, problems and results in the study of complex software systems.

The programme is aimed at graduates in computing and engineering who wish to gain a greater depth of understanding about the engineering of complex software systems. The programme will be particularly well suited to software systems engineering professionals who wish to refresh, enhance and advance their understanding of the principles and techniques that they apply in their work. The programme could also be undertaken as a bridge to a more research-oriented programme of study for students who are as yet uncertain about their desire to embark upon a research career.

PROGRAMME OUTCOMES:

The programme provides opportunities for students to develop and demonstrate knowledge and understanding, qualities, skills and other attributes in the following areas:

A: Knowledge and understanding

Knowledge and understanding of:	→	**Teaching/learning methods and strategies:**
1. The nature of large, complex software systems and how that nature varies according to the context in which a system is situated. 2. The lifecycle of software systems engineering, its different phases, the relationships and dependencies between phases, and the problems, issues, techniques, mechanisms and solutions that are relevant to each phase. 3. The state-of-the practice in software systems engineering as well as the most significant trends, problems and results in the study of complex software systems. 4. Capturing and articulating requirements for a software system in consultation with appropriate stakeholders. 5. Constructing informal, semi-formal and formal models of the structure, static relationships, and dynamic behaviour of a software system at different levels of abstraction and suitable for different phases of the software lifecycle. 6. Manual and automated techniques for analyzing and testing the properties of a software model. 7. Manual and automated techniques for identifying inconsistencies between software models expressed at different levels of abstraction. 8. Applying tools and environments to the construction of software models. 9. Management of software systems engineering projects, the processes defined to guide such projects, and the metrics used to measure the progress of such projects. 10. The most important formal and de facto standards for software systems engineering, including important modelling notations, component models, middleware, tools and environments. 11. The advantages and disadvantages of the approaches to software systems engineering learned in the programme.		Knowledge and understanding in all areas will be imparted through taught lectures that present fundamental principles and techniques in a general fashion and also illustrate them to the greatest extent possible through synthetic examples drawn from the scientific literature and, where appropriate, real-world examples drawn from the popular press or the instructors' knowledge and experience. Knowledge and understanding in all areas will be enhanced through supplemental readings drawn from the scientific literature. A greater intuitive understanding of areas 2, 5, 6, 7, 8 and 10 will be imparted through hands-on experience acquired first in small-scale written or laboratory exercises that support the taught courses, and ultimately in a group project in which students will act as team members carrying out the engineering lifecycle for a significant software system. The group project will also impart a greater intuitive understanding of areas 4 and 9.

	→	**Assessment:** Knowledge and understanding in all areas will be assessed through written exercises with modelling notations, laboratory exercises with tools and environments, unseen examination papers, and a significant, comprehensive group project.
B: Skills and other attributes		
Intellectual (thinking) skills: 1. Translation of understanding into models. 2. Logical and algorithmic reasoning at multiple levels of abstraction. 3. Selection of appropriate solution for problem at hand. 4. Analysis and interpretation. 5. Critical evaluation. 6. Precision of thought. 7. Independence of thought. 8. Scientific integrity.	→	**Teaching/learning methods and strategies:** The intellectual skills are acquired through the teaching/learning programme described above. The lectures provide the foundation for developing the skills, while the written and laboratory exercises and group project plus associated feedback provide the opportunity for individuals to develop and hone their skills.
	→	**Assessment:** The intellectual skills are assessed through written and laboratory exercises, unseen examinations, and the group project. The group project in particular provides an extensive opportunity for students to demonstrate the intellectual skills, to the extent that it realistically simulates a software systems engineering project requiring the application of creativity, judgment and critical thinking in the design, analysis and refinement of software models and artefacts.

C: Skills and other attributes

Practical skills (able to): 1. Locate and evaluate relevant scientific literature. 2. Plan and undertake written and laboratory exercises. 3. Present the results of exercises in written form. 4. Communicate technical information about software systems in an effective manner both orally and in writing. 5. Design and analyze software models at multiple and appropriate levels of abstraction. 6. Critically evaluate problem solutions and work performance. 7. Apply project management skills	→	**Teaching/learning methods and strategies:** Skill 1 will be acquired primarily through self-motivated efforts to read beyond the provided supplemental readings, and through background reading for group project reports. Skills 2, 3 and 5 are acquired through the written and laboratory exercises. Skills 4, 5, 6 and 7 are acquired throughout the group project through periodic oral progress presentations to the project supervisor and through written reports submitted at significant milestones.
	→	**Assessment:** The practical skills are assessed through assessments of the work outputs as described immediately above. Skills 4 and 5 are assessed through unseen examinations.

D: Skills and other attributes

Transferable skills (able to): 1. Prepare and present effective oral presentations. 2. Present technical material effectively in written form. 3. Solve problems that are presented or stated in abstract terms. 4. Critically analyze and evaluation solutions to problems. 5. Relate abstract concepts to concrete phenomena. 6. Manage time effectively.	→	**Teaching/learning methods and strategies:** A programme in software systems engineering presents students with an intellectually demanding array of highly abstract concepts and principles that are difficult to appreciate and apply in the absence of hands-on experience. The programme will therefore provide frequent opportunities through written and laboratory exercises, and ultimately through the group project, to become comfortable and conversant with the precepts of software systems engineering, and to exhibit this comfort and conversance through the transferable skills.

	→	**Assessment:** Skill 6 is implicitly required in order to successfully complete the programme. The other skills are all assessed through the group project, while Skills 2, 3, 4 and 5 are assessed through the exercises and the unseen examinations.

The following reference points were used in designing the programme:

- the Framework for Higher Education Qualifications (http://www.qaahe.org.uk/academicinfrastructure/FHEQ/EWNI/default.asp);
- the programme specifications for UCL degree programmes in relevant subjects (where applicable);
- UCL teaching and learning policies;
- staff research.

Please note: This specification provides a concise summary of the main features of the programme and the learning outcomes that a typical student might reasonably be expected to achieve and demonstrate if he/she takes full advantage of the learning opportunities that are provided. More detailed information on the learning outcomes, content and teaching, learning and assessment methods of each course unit/module can be found in the departmental course handbook. The accuracy of the information contained in this document is reviewed by UCL and may be checked by the Quality Assurance Agency for Higher Education.

Programme Organiser(s) Name(s):	Dr E Letier
Date of Production:	2005
Date of Review:	April 2008
Date approved by Head of Department:	TBA
Date approved by Chair of Departmental Teaching Committee:	TBA
Date approved by Faculty Teaching Committee	TBA