Programme Specification[1]

1. Programmes:

Programme Title	UCAS GU Code Code
BSc Honours in Software Engineering	G430 G430-2208
BSc Honours in Software Engineering (Faster Route)	G601-2208

2. Attendance Type:

Full Time

2.1 SCQF Level:

10

2.2 Credits:

490

3. Awarding Institution:

University of Glasgow

4. Teaching Institutions:

5. College:

College of Science and Engineering

6. School/Institute:

Computing Science [REG30200000]

7. Programme Accredited By:

[1] 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 full advantage is taken of the learning opportunities that are provided. More detailed information on the learning outcomes, content and teaching, learning and assessment methods of each course can be found in course handbooks and other programme documentation and online at www.gla.ac.uk

The accuracy of the information in this document is reviewed periodically by the University and may be checked by the Quality Assurance Agency for Higher Education.

British Computing Society (BCS)
IET (formerly IEE)

8. Entrance Requirements:

Please refer to the current undergraduate prospectus at:
http://www.gla.ac.uk/undergraduate/

8.1 ATAS Certificate Requirement:

ATAS Certificate not required

9. Programme Aims:

The focus of the Software Engineering degree is on topics directly relevant to the development of large and complex software systems. Initially this programme shares the fundamentals with the Computing Science Single Honours Degree. It becomes more specialised from year 3 with a focus on software design and implementation in the Team Project and subsequently in the choice of Level 4 Electives and Individual Project. This programme also includes a formally supervised and assessed industrial summer placement between Level 3 and Level 4.

This degree programme aims to:

- provide students with a deep understanding of the theory and practice of computing;
- give students the opportunity to study a broad range of core computing science topics;
- encourage students to discover the connections among these topics and to understand their common theoretical foundations;
- produce graduates fit to occupy responsible positions in the software industry;
- expose students to software engineering in an industrial context via summer work placement;
- give students the opportunity to choose selected Software Engineering topics to study in considerable depth thereby equipping the best graduates to enter research programmes;
- emphasise unchanging principles in computing science;
- encourage independent study habits that will stand graduates in good stead throughout their professional careers;
- enable students to enhance their transferable and interpersonal skills, particularly written and oral communication and team working.

10. Intended Learning Outcomes of Programme:

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

At the end of the programme students should be able to demonstrate a deep understanding of advanced topics chosen by the student, such as:

Information Engineering	Students should be able to: - relate how humans interact with computers; - identify design, construction and evaluation techniques for human-computer interfaces; - reproduce details related to the collection, organisation, manipulation, communication, and display of information by computers; - discuss the design and operation of database systems; - use database technologies;

	- apply human-computer interaction principles, information retrieval techniques; - relate information systems principles; and - define multimedia usage details.
Systems Architecture	Students should be able to: - explain computer architecture and networking principles, at all levels of abstraction from the system level down to the gate level; - employ concurrent and distributed computation techniques and practice their specification and implementation in hardware and software; - discuss and diagram hardware architecture, computer-based systems, computer communications, computer networks, concurrency and parallelism, distributed computer systems and operating systems.
Programming Languages	Students should be able to: - recall the fundamental concepts underlying imperative, object-oriented, functional and concurrent programming languages; principles of language design and specification; knowledge of implementation techniques; - apply programming fundamentals, analyse comparative programming languages, compilers and use syntax-directed tools.
Software Engineering	Students should be able to: - recall the principles underlying the building and maintenance of large software artefacts; - apply modern software engineering methods and to use specific software engineering tools; - relate the strengths and weaknesses of formal and informal software engineering methods; - define middleware; - perform systems analysis and design.
Theoretical Foundations	Students should be able to: - recall discrete mathematics principles; - analyse algorithms and their complexity; - tell how these underpin other areas of the subject; - experiment with data structures and algorithms.

At the end of the programme students should be able to demonstrate advanced skills dependent on topics chosen by the student, such as the ability to:

- program in several imperative, object-oriented, functional and concurrent programming languages; a thorough mastery of a least one of these languages;
- engineer substantial software systems through all stages of their life cycle, namely problem analysis, requirements, design, specification, construction, testing and modification;
- evaluate systems in terms of general quality attributes and possible trade-offs presented within the given problem;
- recognise any risks or safety aspects that may be involved in the operation of computing equipment within a given context;
- specify and implement concurrent and distributed computation.

At the end of the programme students should be able to demonstrate a deep understanding of advanced topics chosen by the student, which will lead to the ability to:

- debate the strengths and weaknesses of different programming languages;
- distinguish social, professional and ethical implications of the use of computers and software development;
- model computer-based systems for the purpose of comprehension, communication, prediction and the understanding of trade-offs;

- critically evaluate and test the extent to which a computer-based system meets the criteria defined for its current use and future development;
- make design decisions based on appropriate correctness and efficiency considerations;
- use formal and semi-formal methods in the analysis and verification of software.

At the end of the programme students should be able to:

- work individually, including managing learning and development, making use of time management and organisational skills;
- work in teams, recognising the different roles team members adopt;
- present succinctly rational and reasoned arguments (orally or in writing);
- effectively perform information-retrieval tasks (e.g. using search engines and catalogues);
- present cases involving a quantitative dimension thus demonstrating basic numeracy; and
- effectively use general IT facilities

11. Assessment Methods:

- Knowledge and understanding

  Unseen examinations, consisting principally of short-answer questions with some essay-style questions. Assessed coursework in the form of tutorial exercises and reports of laboratory activity.
- Practical, discipline-specific, skills

  Practical seen examination. Assessed coursework exercises will each assess various subsets of these skills. Major project work is assessed using a final written report, a demonstration and an oral presentation.
- Intellectual (thinking) skills

  Major project report assesses all skills. Assessed coursework may assess strengths and weakness of different programming paradigms, modelling computer based systems for the purposes of comprehension, communication, prediction and understanding trade-offs and risks. An awareness of social, professional and ethical implications of the use of computer applications and software development is assessed via individual performance in group work, and by unseen essay on topical issues.
- Transferable skills

  Major team and individual projects, as well as some coursework exercises, assess the ability to work individually or in teams, including managing learning and development, time management and organisation skills and the different roles team members adopt. The Professional Skills & Issues course assesses reflection and communication and effective information-retrieval skills.

  Reporting on the industrial placement assesses the ability to describe an independent piece of work both in writing, via the placement report, and verbally, through the presentation. It also assesses the student's ability to reflect on how their academic work relates to the practice of software engineering.

12. Learning and Teaching Approaches:

- Knowledge and understanding

  Contact with teaching staff is through lectures, large and small-group tutorials, workshop and laboratory sessions. The majority of staff provide handouts in lectures. Students are expected to augment these with their own notes, using these as a basis for further regular study during the course. Formatively assessed tutorial and laboratory exercises give students the opportunity to exercise their developing knowledge and understanding. Feedback on exercises is given individually or collectively.
- Practical, discipline-specific, skills

  Demonstrations are given and case-studies examined in lectures, workshops and tutorials.

  Extensive coursework exercises in the early years support the development of programming skills. Major team and individual project work develops particularly the ability to evaluate systems, recognise risks or safety aspects in the operation of computing equipment within a given context, and the ability to engineer substantial software systems through all stages of their life cycle. Subject coursework, in many different styles, often encourages development of all these skills.

  The summer placement gives the students the opportunity to exercise core skills learned, on the first three years of the programme, in a commercial environment. It also motivates their further study of software engineering in the final year of the programme.

- Intellectual (thinking) skills

  Lectures introduce these skills, which are developed using major project work, coursework, workshops. An awareness of social, professional and ethical implications of the use of computer applications and software development is developed using a series of group exercises, including debates, IT news analysis.

- Transferable skills

  Lectures and workshops introduce time management, reflection and communication, and organisation and planning, and these are developed extensively in major project work in particular.

  Numeracy in both understanding and presenting cases is developed as required in major and minor project work. Effective information retrieval skills are developed during major project work and coursework exercises. The effective use of IT facilities is developed as a side effect of all practical work.

  The industrial placement provides practical experience of time management, working to deadlines and developing interpersonal skills by working with professional software engineers on real projects.

13. Relevant QAA Subject Benchmark Statements and Other External or Internal Reference Points:

The following web links introduce the benchmarks that are used to guide and assess our programmes. We monitor our courses against these on a regular basis, further information about this process and about recent developments in these benchmarks can be obtained direct from the school.

http://www.qaa.ac.uk/Publications/InformationAndGuidance/Pages/Subject-benchmark-statement-Computing.aspx
http://www.theiet.org/careers/profreg/
http://www.bcs.org/server.php?show=nav.7065

14. Programme Structure and Features:

The Single Honours degree programme extends over four years of full-time study.
A candidate for the Honours degree must obtain a minimum of 490 credits, 240 of which must be awarded for Honours courses.

Level 1
There are three sets of courses currently offered at level 1. Either set enables students to continue to Honours level:

Set 1: aimed at students with prior programming experience:
Computing Science 1P (COMPSCI1001) (Level 1, 20 credits) (semesters 1 and 2)
Computing Science 1Q (COMPSCI1002) (Level 1, 20 credits) (semesters 1 and 2)
Other subjects (Level 1, 80 credits)

Set 2: aimed at students with no prior programming experience:
Computing Science 1CT (COMPSCI1016) (Level 1, 20 credits) (semester 1)
Computing Science 1PX (COMPSCI1017) (Level 1, 10 credits) (semester 2)
Computing Science 1S (COMPSCI1018) (Level 1, 10 credits) (semester 2)
Other subjects (Level 1, 80 credits)

Set 3: aimed at students with no prior programming experience:
Computing Science 1CT (COMPSCI1016) (Level 1, 20 credits) (semester 1)

Computing Science 1PX (COMPSCI1017) (Level 1, 10 credits) (semester 2)
Computing Science 1Q (COMPSCI1002) (Level 1, 20 credits) (semesters 1 and 2)
Other subjects (Level 1, 80 credits)

*Students will be strongly encouraged to include 40 credits of Level 1 Mathematics in year 1 or 2.

Level 2
Algorithmic Foundations (COMPSCI2003) (Level 2, 10 credits) (semester 1)
Computer Systems (COMPSCI2005) (Level 2, 10 credits) (semester 1)
Java Programming (COMPSCI2001) (Level 2, 10 credits) (semester 1)
Algorithms and Data Structures (COMPSCI2007) (Level 2, 10 credits) (semester 2)
Information Management (COMPSCI2006) (Level 2, 10 credits) (semester 2)
Object-Oriented Software Engineering (COMPSCI2008) (Level 2, 10 credits) (semester 2)
Other subjects (Level 1 or 2, 60 credits)

Computing Fundamentals (COMPSCI2002) (level 2, 10 credits) (semester 1) is required to be taken by any student who has done set 3 in level 1.

Level 3
Entry to the Honours years of the programme is at the discretion of the Head of School of Computing Science. Entry to Software Engineering (SE3H) may be competitive due to the limited number of summer placements available. Entry will be guaranteed to students who have demonstrated exemplary programming skills during Levels 1 and 2 Computing Science. A grade point average of B3 in Algorithms and Data Structures 2, Java Programming 2 and Object-Oriented Software Engineering 2 and at least a grade point average of C3 (i.e. C average) over all of six Level 2 Computing Science courses at the first attempt is expected. The College of Science and Engineering have a minimum level of performance that must be met before a student may enter any Honours programme.

Students in Year 3 take a fixed curriculum designed to give breadth in the subject.
Advanced Programming (COMPSCI4010) (Level 3, 10 credits) (semester 1)
Algorithmics (COMPSCI4009) (Level 3, 10 credits) (semester 1)
Interactive Systems (COMPSCI4014) (Level 3, 10 credits) (semester 1)
Programming Languages (COMPSCI4016) (Level 3, 10 credits) (semester 1)
Database Systems (COMPSCI4013) (Level 3, 10 credits) (semester 2)
Distributed Information Management 3 (COMPSCI4048) (Level 3, 10 credits) (semester 2)
Networked Systems (COMPSCI4012) (Level 3, 10 credits) (semester 2)
Operating Systems (COMPSCI4011) (Level 3, 10 credits) (semester 2)
Professional Software Development (COMPSCI4015) (Level 3, 20 credits) (semesters 1 and 2)
Team Project H (COMPSCI4047) (Level 3, 20 credits) (semesters 1 and 2)

Summer between Years 3 and 4
Software Engineering Summer Placement (COMPSCI4046) (Level 3, 10 credits)

Assessment of the placement is based on a written report and presentation by the student, in conjunction with input from a visit to the student during the placement by a member of academic staff.

Level 4
Entry to the 4th year of the BSc (Hons) Software Engineering is dependent on the student achieving an aggregate score of 9 (equivalent to D3) at the end of their 3rd year. Students failing to achieve the minimal level for progression will be assessed for the early exit qualification of BSc in Software Engineering based on their results in level 3, including the industrial summer placement (370 credits in total).

Software Engineering Honours students select eight 10-credit subject courses. Of these eight, at least 4

courses should be chosen from the specified group of courses listed below. These courses are designed to provide students with depth in a subject area.

Big Data: Programming and Management (COMPSCI4064) (Level 4, 10 credits) (semester 1)
Component Based Software Engineering (COMPSCI5056) (Level M, 10 credits) (semester 2)
Cyber Security (COMPSCI4062) (Level 4, 10 credits) (semester 2)
Enterprise Computing (COMPSCI5008) (Level M, 10 credits) (semester 1)
Human Computer Interaction (COMPSCI4023) (Level 4, 10 credits) (semester 1)
IT Architecture (COMPSCI5013) (Level M, 10 credits) (semester 2)
Safety Critical Systems (COMPSCI4045) (Level 4, 10 credits) (semester 1)
Software Engineering for Financial Systems (COMPSCI4067), (Level 4, 10 credits) (semester 2)

Also, from the eight courses, at least one must be:
Cyber Security (COMPSCI4062) (Level 4, 10 credits) (semester 2)
Human Centred Security (COMPSCI5060) (Level M, 10 credits) (semester 1)
Safety Critical Systems (COMPSCI4045) (Level 4, 10 credits) (semester 1)

This list will change from year to year, depending on staff availability and resources.

In addition, students undertake the following compulsory courses:
Individual Project (COMPSCI4025P) (Level 4, 30 credits) (semesters 1 and 2)
Professional Skills and Issues 4 (COMPSCI4038) (Level 4, 10 credits) (semesters 1 and 2)

Honours Assessment
Within each year, courses are weighted according to credits. The final Honours assessment is based on 40% of the level 3 aggregated score combined with 60% of the level 4 aggregated score.

A student failing to progress to Level 4 of the Software Engineering Honours degree may be awarded a BSc Designated degree in Software Engineering, provided they meet the general regulations for a designated degree.
The curriculum for a Designated degree in Software Engineering is:
Years 1, 2 and 3 the same as BSc Honours Software Engineering (see above) and also completion of the Software Engineering Summer Placement (COMPSCI4046) (Level 3, 10 credits) (summer)

Alternative Faster Route entry (360 credits)
Students taking this route will come straight into level 2. They will take the following courses totalling 70 credits in Computing Science:
Algorithmic Foundations (COMPSCI2003) (level 2, 10 credits) (semester 1)
Computing Fundamentals (COMPSCI2002) (level 2, 10 credits) (semester 1)
Java Programming (COMPSCI2001) (level 2, 10 credits) (semester 1)
Systems and Networks (COMPSCI4043) (level 4, 10 credits) (semester 1)
Algorithms and Data Structures (COMPSCI2007) (level 2, 10 credits) (semester 2)
Information Management (COMPSCI2006) (level 2, 10 credits) (semester 2)
Object-Oriented Software Engineering (COMPSCI2008) (level 2, 10 credits) (semester 2)
*other subjects from level 1 or 2, totalling 50 credits

It is strongly recommended that 40 credits of level 1 Mathematics are taken, unless the student has an equivalent mathematics qualification on entry.

Faster Route entry to Honours
Students will normally be admitted if they have a grade point average of at least C3 (i.e. C average) over all seven Computing Science courses, including C3 or better in Java Programming 2, Algorithms and Data Structures 2 & Object-Oriented Software Engineering 2 at any diet.

Note that from level 3 onwards, the Faster Route programme is exactly the same as the standard Honours

programme.

For more information on courses, see the course catalogue:
http://www.gla.ac.uk/coursecatalogue/

Regulations
This programme will be governed by the relevant regulations published in the University Calendar. These regulations include the requirements in relation to:

(a) Award of the degree
(b) Progress
(c) Early exit awards
(d) (For undergraduate programmes, where appropriate) Entry to Honours

http://www.gla.ac.uk/services/senateoffice/calendar/

15. Additional Relevant Information:

Support for students is provided by the Postgraduate/Undergraduate Adviser(s) of Studies supported by University resources such as the Effective Learning Adviser located in the Student Learning Service (www.gla.ac.uk/services/tls/sls/), the Student Counselling and Advisory Service (www.gla.ac.uk/services/counselling/), the Student Disability Service (www.gla.ac.uk/services/studentdisability/) and the Careers Service (www.gla.ac.uk/services/careers/).

16. Academic Session:

2013-14

Date of production/revision:	28/10/2013 10:10