S.08-52

SIMON FRASER UNIVERSITY

Senate Committee on University Priorities Memorandum

TO:	Senate	**FROM:**	John Waterhouse [illegible] Chair, SCUP Vice President, Academic
RE:	Faculty of Applied Sciences: Full Program Proposal for a Software Systems Major in the School of Computing Science (SCUP 08-08)	**DATE:**	March 6, 2008

At its February 27, 2008 meeting SCUP reviewed and approved the full program proposal for a Software Systems Major in the School of Computing Science from the Faculty of Applied Sciences.

Motion

That Senate approve and recommend to the Board of Governors, the Full Program Proposal for a Software Systems Major in the School of Computing Science in the Faculty of Applied Sciences.

encl.

c: T. Shermer

Full Program Proposal: Software Systems Major

Tom Shermer, Associate Director for Surrey, School of Computing Science

December 7, 2007

Revised February 28, 2008

Credential to be Awarded

B.Sc. (Bachelor of Science)

Location

SFU's Surrey Campus

School/Faculty Offering Program

Computing Science/Faculty of Applied Sciences

Anticipated Program Start Date

September 2008

Description of Proposed Program

A computing major focused on software development and software engineering.

Aims, goals and/or objectives

The aim of this program is to provide students with the skills, knowledge, and thought-processes necessary for the production of professional software, while at the same time giving them a broad background in the types of computing systems used in industry, both currently and in the near future.

Anticipated contribution to the mandate and strategic plan of the institution

This program directly fits into the applied, high-tech focus of the Surrey campus, and we anticipate strong ties and natural joint program development with other Surrey programs, including Mechatronics Engineering, Business, and Interactive Arts and Technology.

Target audience

The program is targeted at students with a concrete, hands-on learning style, entering university from high school.
The program is designed for students who are interested in the creation and design of technology, but have a more concrete learning style and want more immediately applicable software development skills. Our existing major focuses more on abstract concepts, and some students have difficulties with this approach. We also hope to attract students who have a greater interest in software development and software engineering than can be met by our existing major.

Content

The curriculum of the program is divided into three broad areas: Software Engineering, Fundamentals, and Systems. Software Engineering, including the study of computer languages and compilers, has 24 required credit hours of material. Fundamentals, which includes mathematics, writing, and theoretical computing, has 18 required credits. Systems has 15 required credits, and in addition to traditional systems areas it will include a strong emphasis on the emerging field of distributed embedded systems. In addition, students must complete a 9-credit specialization in 3rd- and 4th- year courses. Students must also meet the university's breadth (WQB) requirements, which will require approximately 18 credits outside of the other program requirements, leaving approximately 36 credits of completely free electives. This elective count allows for students to easily combine Software Systems with minors or concentrations in other disciplines. The curriculum has been designed in close accordance with Software Engineering standards established by the major professional computing and engineering bodies (ACM and IEEE).

Delivery methods

It will be delivered with a combination of lecture and laboratory work, with ample project- and case- based pedagogy to appeal to the learning style of the target audience.

Linkages between the learning outcomes and the curriculum design, including an indication whether a work experience/work place term is required for degree completion

The program and curriculum have been designed without a required internship or workplace experience. However, given the strong demand and projections in the software industry, it is anticipated that any student wanting an internship will have one available to them. Currently, our co-op department has more computing science internships available than we have qualified students to fill them.

Distinctive characteristics

We anticipate that this program will see significant demand from students who prefer the pedagogical approach taken by this program, in contrast to our existing major. We also hope to attract students who have a greater interest in software development and software engineering than can be met by our existing major.

Anticipated completion time in years or semesters

The program is designed to be completed in 8 semesters of full-time study.

Enrolment plan for the length of the program

Our target student intake is 30-50 students per year.

Policies on student evaluation

Students will be evaluated on homework, projects, class participation, and examinations, in accordance with the current practices in the School of Computing Science and Simon Fraser University.

Policies on faculty appointments (minimum qualifications)

Courses will be taught by a combination of tenured and tenure-track faculty (Ph.D. required) and lecturers (M.Sc. required), with occasional use of sessional instructors. Normal qualifications for a sessional instructor are an M.Sc., although, given the applied nature of this program, we will also seek out candidates who have significant experience in the software industry.

Program Resources

When the Software Systems program commences, the School plans to phase out the (currently minimal) offering of the Computing Science major at the Surrey campus. This means that the faculty of the Surrey branch of the School (6 tenured/tenure-track and 3 lecturers) will be mainly teaching in the Software Systems program. (The remainder of their teaching—approximately 1/3 of it— will be graduate teaching and service teaching in support of other majors and joint programs.) The teaching requirements for the program will be met using these existing faculty augmented by resources transferred from the School's Burnaby campus operations. These transferred resources can include instructors (some Burnaby-housed faculty will teach courses in Surrey), budget for sessionals, and any replacement positions that may come open. In any event, the School of Computing Science will undertake the offering of the Software Systems program within current budget and without requiring additional resources.

The faculty currently at the Surrey branch of the School, with qualifications, rank, and area, are:

Dirk Beyer, Ph.D., Assistant Professor, Software Engineering
Robert Cameron, Ph.D, Professor, Software Engineering Languages
Toby Donaldson, Ph.D., Lecturer, Artificial Intelligence
John Edgar, M.Sc., Algorithm Animation
Mohamed Hefeeda, Ph.D, Assistant Professor, Computer Networks
Harinder Khangura, M.Sc., Social Impact of Computers
Thomas Shermer, Ph.D., Professor, Algorithms
Tamara Smyth, Ph.D, Assistant Professor, Digital Signal Processing

Kay Wiese, Ph.D, Associate Professor, Bioinformatics

Policies on program assessment

Internally, the program will be closely monitored, assessed, and adjusted each year during the first several years. The program will also be subject to an external review every 7 years, in accordance with University policy. In addition, the School will keep in contact and consult with industry individuals and groups to monitor the suitability of the Software Systems graduates to the rapidly-evolving software industry.

Level of support and recognition from other post-secondary institutions, (including plans for admissions and transfer within the British Columbia post-secondary education system) and relevant regulatory or professional bodies, where applicable

Software Systems was not explicitly designed as a program suitable for professional certification. We are, however, exploring various accreditation options more broadly in the School of Computing Science.

The program was designed to allow students to easily transfer between the Computing Science and Software Systems majors. Thus, many of the required courses in the first and second year of Software Systems are shared with the existing Computing Science major. Since the School of Computing Science already works closely with the Computer Science course articulation committee and has an extensive articulation framework in place with other British Columbia post-secondary institutions, most articulation issues are already resolved. Transfer admission standards will be set by the School using the process it already uses to set such standards for the Computing Science major.

Evidence of student interest and labour market demand

The graduates from our existing major are currently in high demand in the local, national, and international technology industries. We anticipate that graduates from this program will be similarly successful due to the focus on software development and software engineering.

Related programs in your own or other British Columbia post-secondary institutions.

Currently in British Columbia, university-level Computer Science programs are offered at SFU, UBC, UVic, and UNBC. Additional CS degree programs are offered by the University Colleges, and several colleges offer University-transfer CS courses. UBC offers a certificate in Software Engineering and Quality Assurance; their Electrical and Computer Engineering Department offers a Software Engineering option, as does their Computer Science Department. UVic offers a Bachelor of Software Engineering degree.

For more information, please contact Dr. Thomas Shermer, Associate Director (Surrey) for the School of Computing Science at Simon Fraser University, at (778) 782-7571 or shermer@cs.sfu.ca

Software Systems Program Calendar Entry

[To be inserted immediately before the "SFU-Zhejiang University Dual Degree Program" section, p. 114 of the 2007/8 Calendar.]

Software Systems Program

This program is designed to provide students with the skills, knowledge, and thought-processes necessary for the production of professional software, while at the same time giving them a broad background in the types of computing systems they are likely to encounter over the course of their careers.

For information on course planning, students should visit www.cs.sfu.ca/undergrad/Advising.

Systems Requirement

Students must complete all of

CMPT 150-3 Intro Computer Design
CMPT 250-3 Intro Computer Architecture
CMPT 300-3 Operating Systems I

and two additional courses chosen from

CMPT 170-3 Introduction to Web Application Development
CMPT 371-3 Data Communications and Networking
CMPT 471-3 Networking II
CMPT 354-3 Database Systems I
CMPT 454-3 Database Systems II
CMPT 401-3 Operating Systems II
CMPT 432-3 Real-time Systems
CMPT 433-3 Embedded Systems
CMPT 470-3 Web-based Information Systems

[15 credits]

Fundamentals Requirement

Students must complete all of

MACM 101-3 Discrete Math I
MATH 151-3 Calculus I

STAT 101-3 Introduction to Statistics *
MATH 232-3 Elementary Linear Algebra
CMPT 322-3 Professional Responsibility and Ethics
CMPT 307-3 Data Structures and Algorithms

* STAT 270-3 may be substituted for STAT 101-3.

[18 credits]

Software Engineering Requirement

Students must complete all of

CMPT 126-3 Introduction to CS & Programming *
CMPT 225-3 Data Structures and Programming
CMPT 212-3 Object-oriented Applications Design in C++
CMPT 276-3 Introduction to Software Engineering
CMPT 373-3 Software Development Methods
CMPT 379-3 Principles of Compiler Design
CMPT 473-3 Software Quality Assurance

and one additional course chosen from

CMPT 383-3 Comparative Programming Languages
CMPT 384-3 Symbolic Computing
CMPT 477-3 Formal Verification
CMPT 474-3 Web Systems Architecture

* Either CMPT 120-3 and 125-3 or CMPT 128-3 may be taken instead of CMPT 126-3.

[24-27 credits]

Specialization Requirement

Students are required to take a "specialization" consisting of nine additional CMPT or MACM credits at the 300- or 400-level. This specialization must be approved by the School. Pre-approved sets of courses can be found at www.cs.sfu.ca/undergrad/Advising.

[9 credits]

Depth Requirement

Students must complete at least 9 CMPT or MACM credits at the 400-level. .

WQB Requirements

The Software Systems Program specifies only 66 credit hours of required courses. This leaves 54 credit hours to meet WQB requirements, as well as take other electives or pursue a minor. As such, few of the WQB requirements are specified, but students have many opportunities to fulfill the requirements.

Students in this program will take many quantitative courses, including MACM 101-3 and MATH 151-3.

A discipline-specific upper-division writing course, CMPT 322 (Professional Responsibility and Ethics) is being proposed as part of the program.

The lower-division writing course and breadth courses will be taken as electives.

Preface to Course Proposals

This section contains common material for the course proposals in the Software Systems Program. Included are the calendar entry for the program requirements, and course descriptions for eight new courses required to implement the major. These courses are:

CMPT 170	Introduction to Web Application Development
CMPT 276	Introduction to Software Engineering
CMPT 322W	Professional Responsibility and Ethics
CMPT 373	Software Development Methods
CMPT 432	Real-time Systems
CMPT 433	Embedded Systems
CMPT 473	Software Quality Assurance
CMPT 474	Web Systems Architecture

The remainder of the Software Systems major uses courses already taught by Computing Science or other units.

In the course proposals, we are asked to:

> *Provide details on how existing instructional resources will be redistributed to accommodate this new course. For instance, will another course be eliminated or will the frequency of offering of other courses be reduced; are there changes in pedagogical style or class sizes that allow for this additional course offering?*

and we will address this question here rather than repeating ourselves in each course proposal.

We plan to phase in these new courses over three years, starting with 100- and 200-level courses in 2008, and continuing 300-level courses in 2009 and 400-level courses in 2010. As this happens, we will be phasing out our current (minimal) offering of the standard Computing Science major in Surrey. We are conducting this transition gradually so that students who have already started a standard Computing Science major in Surrey will be able to complete their studies there. Even in Fall 2011, after the CS major is no longer offered in its entirety at Surrey, students will still have some possibility for completing there, as the CS and Software Systems majors have some upper-division overlap, and the new Software Systems courses will be usable in the CS major. Depending on which courses these students have taken by 2011, though, it may be necessary for them to take a few courses in Burnaby.

The phasing out of the CS major in Surrey will free up most of the instructional resources necessary for the Software Systems major. The School of Computing Science will shift resources to cover the remainder. In particular, we lack depth in Systems and in Software Engineering. We should hire one more faculty member in each of these areas, or (given the applied nature of the areas) establish relationships with reliable sessional instructors who are employed in and experienced with specific subareas. In the event that we need to hire new faculty, we will do this by transferring resources (such as replacement positions for departing faculty) from Burnaby to Surrey. No new resources are required.

We will similarly address

> *Any outstanding resource issues to be addressed prior to implementation: space, laboratory equipment, etc.*

in this document rather than in the individual proposals.

Computing Science at Surrey currently has sufficient laboratory space to offer the proposed courses. Most courses require only standard computing laboratory platforms (equipment and software), which we already have in place. Some courses may require specialized platforms; we will settle on these as close to the first offering as possible, in order to avoid early obsolescence. This is the usual situation in the School of Computing Science; we are continually acquiring new platforms for courses, in order to keep up with the rapid evolution of the field. We will handle the acquisition of platforms for these new courses as part of our usual process, and require no additional resources for it.