

Program Guide

Bachelor of Software Engineering

1. Program Details

Title	Bachelor of Software Engineering
Abbreviation	BSoftEng
RMIT Program Code	BP096
Credit Points	384
Career	undergraduate
Duration/length	4 years full-time or equivalent part-time study
Campuses	City Campus, Vietnam
Location	Onshore and offshore
Owning School	140H – School of Computer Science and IT (www.rmit.edu.au/csit)
Partnered offering / corporate client	Not applicable
ASCED code:	020100 – Computer Science
CRICOS code: (If known)	061069G
Proposed Introduction	Semester 1, 2008
Contact Details	Santha Sumansekara (santha.sumansekara@rmit.edu.au)

2. Plan Details

RMIT Plan Code	BP096P8
Title	Bachelor of Software Engineering
Award Title	Bachelor of Software Engineering
ASCED code:	020100 – Computer Science
CRICOS code:	061069G

3. Program Map

The program structure for the software engineering is given below.

Year	Semester	BP096 Bachelor of Software Engineering			
1	Sem 1	Programming 1	Database Concepts	Computer Organisation	Mathematics for Computing
	Sem. 2	Programming 2	Web Programming	Data Communication and Net-centric Computing	Software Engineering Fundamentals
2	Sem 1	Programming Techniques	Software Engineering: Process and Tools	Computing Theory	Student Elective
	Sem. 2	Algorithms and Analysis	Operating Systems Principles	Professional Computing Practice	Computer Science Elective
3	Sem 1	Approved Industry Experience 1 (36CP)			Software Engineering Principles and Practice 1
	Sem. 2	Approved Industry Experience 2 (36CP)			Software Engineering Principles and Practice 2
4	Sem 1	Software Engineering Project A (24CP)		Software Engineering Elective	Computer Science Elective
	Sem. 2	Software Engineering Project B (24CP)		Software Engineering Elective	Student Elective

Year One

Total Credit Points = 96

Complete Eight (8) Courses from:			
Subject Area	Catalogue Number	Course Title	Credit Points
COSC	1073	Programming 1	12
ISYS	1057	Database Concepts	12
COSC	1082	Computer Organisation	12
MATH	1074	Mathematics for Computing	12
COSC	1076	Programming 2	12
COSC	2413	Web Programming	12
COSC	1111	Data Communication and Net-centric Computing	12
ISYS	1118	Software Engineering Fundamentals	12

Year Two

Total Credit Points = 96

Complete Six (6) Course from:			
Subject Area	Catalogue Number	Course Title	Credit Points
COSC	1284	Programming Techniques	12
COSC	2299	Software Engineering: Process and Tools	12
COSC	1107	Computing Theory	12
COSC	2123	Algorithms and Analysis	12
COSC	1114	Operating Systems Principles	12
COSC	1147	Professional Computing Practice	12

AND

Complete One (1) Course from:			
Subject Area	Catalogue Number	Course Title	Credit Points
ISYS	2403	Advanced Distributed Systems	12
COSC	1204	Agent-oriented Programming and Design	12
COSC	2269	AI Concepts and Applications	12
COSC	1235	Broadcast Network Applications	12
COSC	2404	Database Administration	12
COSC	2271	Digital Media Computing	12
COSC	2104	Document Markup Languages	12
COSC	2353	E-Commerce and Enterprise Systems	12
COSC	1207	Evolutionary Computing	12
COSC	1187	Interactive 3D Graphics and Animation	12
COSC	1197	Distributed Systems	12
INTE	2425	Introduction to Network Security	12
ISYS	1073	Knowledge and Data Warehousing	12
COSC	1208	Mathematical Logic and Logic Programming	12
COSC	1179	Network Programming	12
COSC	1254	Object Oriented Programming	12
COSC	2391	Software Architecture: Design & Implementation	12
COSC	1226	Real-time Rendering and 3D Games Programming	12
COSC	1093	Scripting Language Programming	12

INTE	1071	Secure E-Commerce	12
INTE	2402	Secure Programming Environments	12
COSC	2412	Unix Essentials	12
COSC	1133	Unix Systems Administration	12
COSC	1221	User Interface Programming	12
INTE	1113	Web 3D Technologies	12
ISYS	1126	Web Database Applications	12
COSC	2276	Web Development Technologies	12
COSC	1301	Web Servers and Web Technology	12
COSC	2424	Windows Systems Administration	12

AND

Complete One (1) Course from:			
Subject Area	Catalogue Number	Course Title	Credit Points
		Student Elective	12

Year Three

Total Credit Points = 96

Complete Four (4) Course from:			
Subject Area	Catalogue Number	Course Title	Credit Points
INTE	2376	Approved Industry Experience 1	36
INTE	2377	Approved Industry Experience 2	36
INTE	2374	Software Engineering Principles and Practice 1	12
INTE	2375	Software Engineering Principles and Practice 2	12

Year Four

Total Credit Points = 96

Complete Two (2) Course from:			
Subject Area	Catalogue Number	Course Title	Credit Points
COSC	2410	Software Engineering Project A	24
COSC	2411	Software Engineering Project B	24

AND

Complete One (1) Course from:			
Subject Area	Catalogue Number	Course Title	Credit Points
ISYS	2403	Advanced Distributed Systems	12
COSC	1204	Agent-oriented Programming and	12

		Design	
COSC	2269	AI Concepts and Applications	12
COSC	1235	Broadcast Network Applications	12
COSC	2404	Database Administration	12
COSC	2271	Digital Media Computing	12
COSC	2104	Document Markup Languages	12
COSC	2353	E-Commerce and Enterprise Systems	12
COSC	1207	Evolutionary Computing	12
COSC	1187	Interactive 3D Graphics and Animation	12
COSC	1197	Distributed Systems	12
INTE	2425	Introduction to Network Security	12
ISYS	1073	Knowledge and Data Warehousing	12
COSC	1208	Mathematical Logic and Logic Programming	12
COSC	1179	Network Programming	12
COSC	1254	Object Oriented Programming	12
COSC	2391	Software Architecture: Design & Implementation	12
COSC	1226	Real-time Rendering and 3D Games Programming	12
COSC	1093	Scripting Language Programming	12
INTE	1071	Secure E-Commerce	12
INTE	2402	Secure Programming Environments	12
COSC	2412	Unix Essentials	12
COSC	1133	Unix Systems Administration	12
COSC	1221	User Interface Programming	12
INTE	1113	Web 3D Technologies	12
ISYS	1126	Web Database Applications	12
COSC	2276	Web Development Technologies	12
COSC	1301	Web Servers and Web Technology	12
COSC	2424	Windows Systems Administration	12

AND

Complete Two (2) Course from:			
Subject Area	Catalogue Number	Course Title	Credit Points
ISYS	1108	Engineering Software Projects	12
ISYS	1084	Object Oriented Software Design	12

ISYS	2405	Software Engineering for Large Scale Systems	12
COSC	2274	Software Requirements Engineering	12
ISYS	2368	Software Reuse	12
ISYS	1087	Software Testing	12
ISYS	1089	Systems Architecture	12
COSC	1183	Usability Engineering	12

AND

Complete One (1) Course from:			
Subject Area	Catalogue Number	Course Title	Credit Points
		Student Elective	12

Program Progression Rules

The program is structured so that capabilities are developed sequentially through the four years. Assumed prerequisite capabilities are listed for each course in the individual course guides. You are strongly advised against enrolling in courses for which you do not have the required prerequisites, unless prior approval has been obtained from the Program Leader. Failure in one or more courses may make it impossible for you to complete the program within the minimum four-year period.

4. External Accreditation and Industry Links

The Computer Science component of this double degree program is accredited at professional level by the Australian Computer Society, which accredits Information and Communication Technology related programs in Australia.

5. Objectives of the Program

The School of Computer Science and Information Technology has a strong tradition of “hands on” teaching, providing students with the opportunity to mix course content and practical experience. This approach, coupled with our close involvement with industry, produces graduates who are highly regarded in the workplace. The curricula used in our various degree programs reflect these needs incorporating cutting-edge technologies while maintaining a good coverage of the theoretical and algorithmic foundations of computer science, information technology, and software engineering.

Software engineering is the discipline of developing and maintaining large software systems that behave reliably and efficiently and are affordable to develop and maintain. It has evolved in response to the increased importance of software in safety-critical applications and to the growing impact of large and expensive software systems in a wide range of applications. Degree programs in computer science and in software engineering have many courses in common. Software engineering students learn more about software reliability and maintenance and focus more on techniques for developing and maintaining software that is correct from its inception. The RMIT Software Engineering degree, in particular, provides a solid theoretical foundation to current software development processes. This program builds necessary skills and provides essential industry experience to develop quality large-scale software systems. It also provides an in-depth coverage in various aspects of software engineering processes including formalisms.

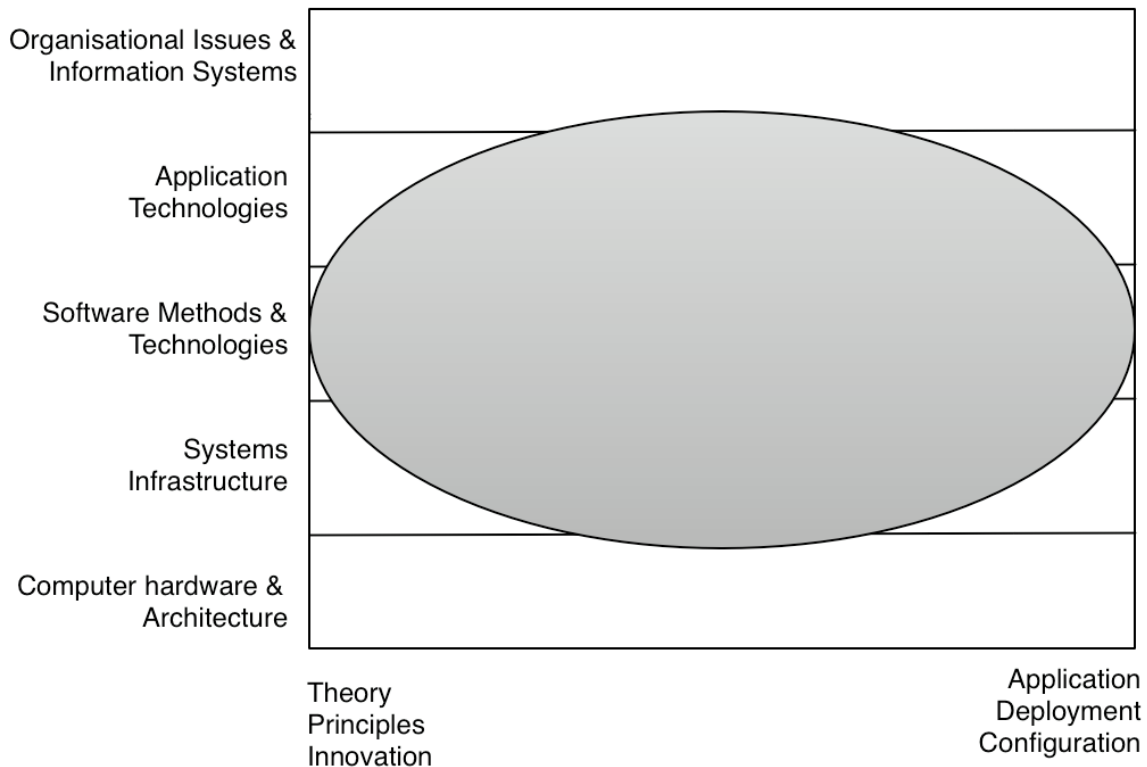
One of the key features of this program is the one-year internship in the third year where you work with an industry partner. This helps you to understand how the software engineering principles you learn in the first two years are applied in the industry, and provides a basis for you to integrate industry practice with theory in the final year. It is important to note that this program is not an engineering degree: “software engineering” is widely recognised as a computing discipline, where use of the term “engineering” is to emphasise its focus on developing and applying robust methods to create reliable products, by *adapting* traditional engineering techniques to the problem of software design and creation. Software engineering is different in character from traditional engineering disciplines, due to the intangible nature of software, software engineering’s focus on human processes rather than

the laws of physics, and the discontinuous nature of software operation. Software engineering differs from traditional engineering in several ways:

- Foundations are primarily in computer science, not in natural sciences.
- The focus is on discrete rather than continuous mathematics.
- The concentration is on abstract/logical entities instead of concrete/physical artifacts.
- There is no manufacturing phase in the traditional sense.
- Software maintenance primarily refers to continued development, or evolution, and not to conventional wear and tear.

The curriculum guidelines for undergraduate degrees compiled by the joint task force of IEEE Computer Society and the Association of Computing Machinery provide a two-dimensional characterisation of the entire computing discipline. One dimension describes the depth of coverage between more theoretical aspects and more applied aspects. The other dimension describes the breadth of coverage of topics between lower-level computer hardware and architecture focused coverage and higher-level organisational issues and information systems focused coverage¹.

The shaded area of the following diagram represents the coverage domain of the Software Engineering degree. Software Engineering covers a wide range with respect to the systematic development of software. This is because software engineers fill a wide range of needs in large-project software expertise. Software Engineering’s main goal is to develop systematic models and reliable techniques for producing high-quality software, therefore these concepts are covered in both theoretical and practical perspectives. The domain of software engineering also extends downward through systems infrastructure, as software engineers develop software infrastructure that is robust in operation. Its domain also extends upward into organisational systems because software engineers are involved in designing and developing information systems that are appropriate to the client organizations.



This program was developed based on the recommendations of the curriculum development joint task force of IEEE Computer Society and Association of Computing Machinery. It was developed through a formal consultation process with a number of stakeholders, including the School’s Industry Advisory Committee, academic staff of the School, alumni, and current students. This process resulted in the identification of the following graduate capabilities that are required of a computer science graduate to successfully engage in a professional capacity in the relevant field of industry of the 21st century.

¹ http://www.acm.org/education/curric_vols/CC2005-March06Final.pdf

6. Statement of capabilities

The graduate capabilities developed by the Software Engineering degree program are composed of the following dimensions:

- **Enabling Knowledge**
This capability allows one to apply knowledge effectively to new situations and learn from the experience.
- **Critical Analysis**
In general, this capability allows one to examine and consider accurately and objectively any topic, evidence, or situation.
More specifically, this capability allows one to:
 - Analyse and model requirements and constraints for the purpose of designing and implementing software systems;
 - Evaluate and compare designs of such systems on the basis of requirements of the organisational needs.
- **Problem Solving**
In general, this capability allows one to analyse problems and synthesise suitable solutions.
Specifically, this capability allows one to:
 - Design and implement software systems that accommodate specified requirements and constraints, based on analysis or modelling or requirements specification
- **Communication**
In general, this capability allows one to communicate effectively with a variety of audiences through a range of modes and media.
Specifically, this capability allows one to:
 - Present and explain complex software systems solutions, alternative solutions, and decision recommendations to both IT and non-IT personnel via technical reports of professional standard and technical presentations.
- **Team Work**
In general this capability allows one to work as an effective and productive team member in a range of professional and social situations.
Specifically, this capability allows one to:
 - Work effectively in different roles, to form, manage, and successfully produce outcomes from teams, whose members may have diverse cultural backgrounds and life circumstances, and differing levels of technical expertise.
- **Responsibility**
In general this capability refers to accepting responsibility for one's own learning and make informed decisions in judging and adopting appropriate behaviour in professional and social situations. This includes accepting the responsibility for life-long learning.
Specifically, this capability allows one to:
 - Effectively apply relevant standards, ethical considerations, and an understanding of legal and privacy issues to designing software systems.

7. An approach to Teaching and Learning (including a statement on assessment)

RMIT has a commitment to the principle of student-centred learning: that learning is most meaningful when topics are relevant to your life, needs, and interests and when learning activities actively engage you in creating, understanding, and connecting to knowledge². The teaching and learning methods used in this program aim to implement student-centred learning by recognizing that your perceptions of the world are important and relevant, and encouraging you to actively participate in your learning, and to develop solutions in collaboration with your

² McCombs, B. and Whistler, J.S. (1997). *The Learner-Centered Classroom and School: Strategies for Increasing Student Motivation and Achievement*. San Francisco: Josey-Bass Publishers.

peers. Learning activities include practical exercises, case study analysis, oral presentations, technical and business reports, and individual and group project work.

Lectures (some presented by industry experts) are used to convey some of the basic information necessary for each part of the various courses. Smaller tutorials or laboratory sessions are then used to explore the ideas raised in the lectures, or to give you hands-on experience of technologies. In tutorials, you will often work in a smaller group of about 5 students, to ensure there is real scope for genuinely interactive discussions. Most courses use carefully constructed tutorial questions to illustrate key concepts and to help you develop your understanding. Course materials (printed course notes, textbooks and reference books) are available from the RMIT Bookshop; the RMIT Library has copies of the books and also provides online access to electronic books and journals; course web pages contain links that let you download worksheets and assignment specifications, email teaching staff, and access message forums, as well as links to external course-related web sites. Lecturers provide additional suitably formatted electronic files and handouts to visually impaired students upon request.

Assessment

The school views teaching and learning as a cyclic activity, with assessment and evaluation driving planning and teaching. Assessment is an integral part of learning: information derived from assessment activities is used to facilitate student learning and development, and to improve the quality of the school's programs, services and facilities. Assessment activities examine processes as well as products, and are designed to measure your work against standards, not against other students. As no one assessment can capture the full range of student learning and academic growth, courses use multiple assessments to evaluate what you know and are able to do and to inform adjustments to learning activities.

Assessment is developmental and continuous: that is, you have the opportunity to learn by building on what you already know and are able to do and to carry forward these skills and knowledge to expanded and more complex uses. To reflect industry practice in this area, as you progress through assessments at each level of the program, you are expected to demonstrate at increasingly higher levels of complexity and integration, the knowledge and capabilities set forth in the program standards.

Formative assessment progresses from tutorial exercises and self-test quizzes in foundation courses to participation in seminar discussions, moderated by the lecturer, in some elective courses, to seminar-style discussions in key courses, where you present additional topics in the course material, and apply your knowledge of earlier topics to recognize underlying principles and potential applications of new topics. Some courses involve group meetings and discussions relating to assignments, and participation in case study sessions within groups and with key input and guidance from lecturers and industry experts. Tutorial exercises allow you to explore team dynamics, diagnostics, and management issues.

Summative assessment also becomes more demanding as you progress from foundation courses to electives and key courses:

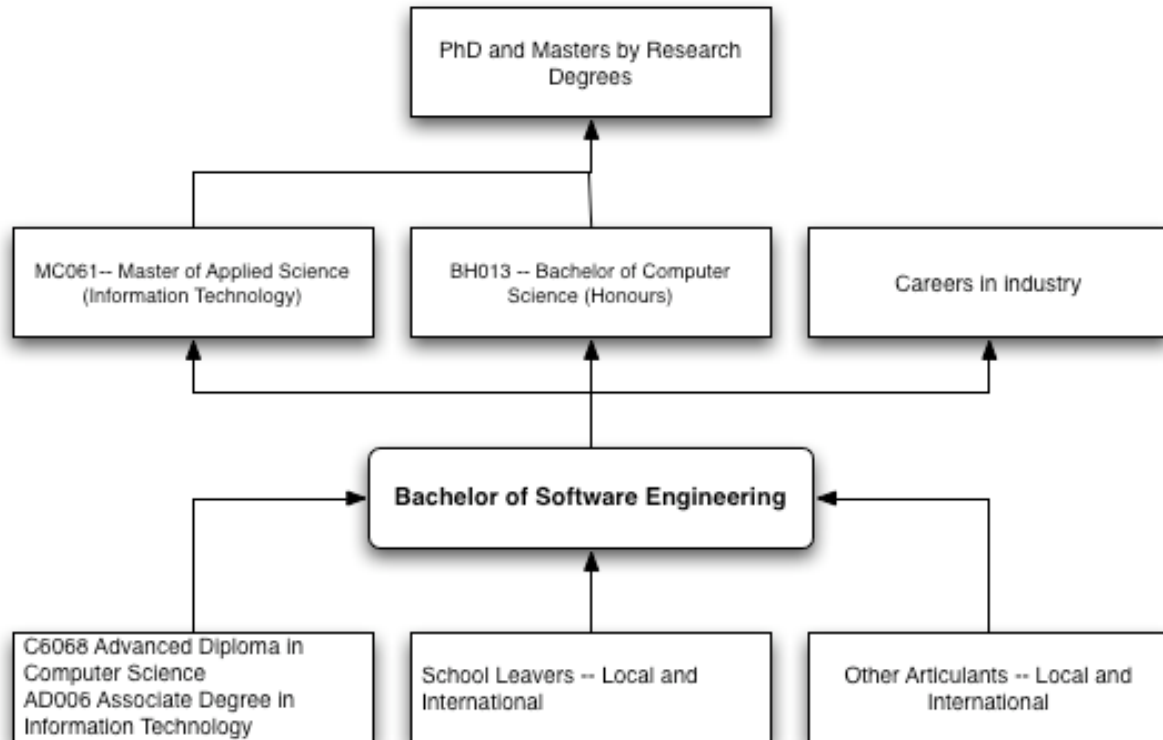
- The core courses focus on key concepts and initial capability attainment: most assessment activities are based on individual skills and capabilities, and ask you to apply fixed "toolsets" to familiar, well-defined problems, to demonstrate that you have grasped the necessary technical foundations and relevant technologies;
- Elective courses require more complex, open problem-solving, with assignments that require you to design or evaluate solutions for problems with complex or conflicting requirements, or to compare alternative solutions for such problems. In most elective courses, assessment activities also emphasize additional graduate capabilities such as written communication, where you demonstrate that you can integrate concepts and arguments into technical or business reports, or literature reviews of relevant standards, ethical considerations, and applicable research. Some elective courses involve group-focussed assessment.
- Assessment of minor courses vary widely among different minor study streams mainly due to substantial differences in the disciplines covered in these minor study areas. You will be advised by an academic advisor about the specific instruments of assessment once you have chosen the minor study area.

In order to be a lifelong learner, you must be able to evaluate your own work. To support this, some group work is peer-assessed, i.e., following criteria specified by the lecturer, or agreed upon by your class, you assess, and are assessed by, the other members of your group. This is in keeping with student-centred learning, and also helps to alleviate a major misgiving about group work - the possibility of some group members being "carried" by the other members.

Most courses in this program also require you to sit a written examination at the end of the semester, worth between 35% (key) and 60% (foundation) of your final result.

A **portfolio** is a collection of evidence that you prepare to demonstrate mastery, comprehension, application, and synthesis of this program's concepts. Many of the learning and assessment activities described in this and the previous section can contribute to your portfolio of evidence, in particular, your individual assignments, and your [journal of your] contributions to group activities (case study analyses, presentations, technical and business reports, and group project work).

8. Articulation and Pathways



The following tables indicate how this program articulates with other programs (TAFE, HE – UG and PG) at RMIT highlighting educational pathways, and indicates to students with existing qualifications the extent of exemptions.

Source Program	Owning school	Credit towards this program		Academic requirement for entry	Terms of entry (guaranteed place, merit, etc)	Date of agreement & expiry
		Courses	Time			
C6068 - Advanced Diploma of Computer Science	School of Life and Physical Sciences	- Programming 1 - Programming 2 - Computer Organisation - Mathematics for Computing - Data Communication and Net-centric Computing - Database Concepts - Two Student Electives	1 yr	Must obtain a CGPA of 3.0 (Distinction Average) or above.	Merit	
AD006 – Associate	School of Life and	- Programming 1 - Programming 2 - Computer	1	Must obtain a	Merit	

Degree in Information Technology	Physical Sciences	Organisation - Mathematics for Computing - Programming Techniques - Software Engineering Fundamentals - Data Communication and Net-centric Computing - Database Concepts	yr	CGPA of 3.0 (Distinction Average) or above.		
----------------------------------	-------------------	---	----	---	--	--

Program Destination	Owning school	Credit from this program towards destination program		Academic requirement for entry	Terms of entry (guaranteed place, merit, etc)	Date of agreement & expiry
		Courses	Time			
MC061 – Master of Information Technology	School of Computer Science & IT	No exemptions	0		Merit	
BH013 – Bachelor of Computer Science with Honours	School of Computer Science & IT	No exemptions	0		Merit	

9. Entrance requirements

VCE Units 3 & 4 English (any) and mathematical methods or specialist mathematics. Students who obtained study scores above 20 for English will earn selection credit.

10. Library, IT and specialist resources

RMIT Library already holds or has ordered all prescribed and recommended books. A limited number of copies of books will be available from the Library; some may be available electronically via Safari Bookshelf or electronic journals. This program will be delivered only in on-campus mode.

No additional IT or specialist resources are required to support the new courses. You will use IT facilities within the school. Special software required, such as Rational Rose, is already licensed and installed.

11. Student expenses and charges in addition to fees

For on-shore students:

None, apart from standard course fees and university charges.

For off-shore students: Students should seek advice from the local provider.

12. Program Transition Plan

The proposed amendments (effective from Sem1, 2008) are for the purpose of inclusion of a few electives into Computer Science elective list and Software Engineering elective list. Therefore, no transition plan is required.

13. Course descriptions

Part A course guides for all the courses can be found at:

<http://www.rmit.edu.au/programs/courses>

Capability Matrix

Capability Development in Computer Science Component:

Capability	Programming 1	Programming 2	Database Concepts	Web Programming	S Eng Fundamentals	Prog Techniques	Algorithms & Analysis	SE: Process & Tools	Computing Theory	Object-oriented Prog	Operating Systems	Database Systems	Artificial Intelligence	Approved Work Experience	SE: Principles and Practice	SE Project
Enabling Knowledge	P	S	P	P	P	S	S	S	S	S	S	S	S	S	S	S
Critical Analysis	P	S	P	P	P	S	S	S	S		S	S	S	S	S	S
Problem Solving	P	S	P	P	P	S	S	S	S	S	S	S	S	S	S	S
Communication	P	S		P	S	S	S	S	S					S	S	S
Team Work		P			P	S		S	S					S		S
Responsibility	P	P		P	S	S	S	S	S	S		S		S	S	S

P – Primary level capabilities

S – Secondary level capabilities