Curriculum Development in Educating Undergraduate Software Engineers – Are students being prepared for the profession?

Jean-Guy Schneider, Lorraine Johnston, and Phil Joyce

Faculty of Information and Communication Technologies
Swinburne University of Technology
P.O. Box 218, Hawthorn, Victoria 3122, Australia
Email: {jschneider,ljohnston,pjoyce}@swin.edu.au

Abstract

The growing importance of software and its role in society places greater emphasis on developing suitably skilled software engineering graduates. In developing suitable curricula, it is vital that we understand standard models of software education, the educational processes and outcomes for the students. This is important not only for students completing a software engineering degree, but also once they have graduated and used their skills as practitioners. Here, we report on a survey of past students who have completed an undergraduate degree in software engineering and compare their perceptions with current efforts to standardise undergraduate software engineering curricula.

1. Introduction

Software is ubiquitous and has a role in almost every facet of daily life. The increase in the use of software focuses attention on both the technology itself, and on the effect of that technology on society. Indeed, software projects are seen as being central to the growth of the information economy. To support this growth, it is important that suitable methods of software development are adopted and those developing software obtain the relevant skills.

Many in the Information Technology (IT) industry carry the position title of “Software Engineer”, but professional recognition has been slow in coming. The terminology of “Software Engineering” was first used at a conference sponsored by a NATO Science committee in 1968 [1]. For many years, we talked of the “software crisis” where many software projects ran late, were over-budget and/or failed to deliver a completed working system. With growing concern, stakeholders from within the IT industry moved to provide professional guidance and to develop formal definitions of Software Engineering.

The IEEE Computer Society defines Software Engineering (SE) as [2]:

1. The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software, i.e. the application of engineering to software; and

2. The study of approaches, as in (1).

Central to these definitions are the systematic, controlled, effective and quantifiable approaches to software development. These take in the wider issues of analysis and evaluation, specification, design, and evolution of software. The issues of management and quality also must be considered, with reference to individual skills, teamwork, professional practice, and novelty in creating software solutions. Establishing a process for building these skills, approaches and attributes requires a level of training and education.

In developing a professional education curriculum, we must define the core set of attributes that a graduate should possess to meet the needs of the software profession and the wider community. We need to understand both the educational process and the educational outcomes for students. Thus, we must know about the impact of the curriculum, not only while students are completing their studies, but also once they have graduated and used their skills as practicing software engineers. We believe that students who have gained contextual experience in industry will be in the best position to analyse their perceptions of their education curriculum.

This past decade, a joint ACM/IEEE task force addressed the issue. The goal was to propose curriculum guidelines incorporating the latest developments in computing technology and matching the needs of both, educational and industrial organizations. This resulted in a final report called “Computing Curriculum – Software Engineering” (CCSE) [3]. However, as this joint task force was mainly formed by academics working in tertiary education,

1Since we started this work, this report was updated and renamed as “Software Engineering 2004.” For reasons of consistency, we will however continue to refer to this report as CCSE.
the question was raised whether the resulting efforts “were too academically biased to set the standard for Software Engineering”, and “would hinder its support from other organisations” [4]. Hence, there is a clear need to evaluate proposed Software Engineering curricula, taking the viewpoints and interests of all stakeholders into consideration.

In this paper, we report on a survey of graduates who have completed the Bachelor of Software Engineering (BSE) degree from Swinburne University of Technology in Australia, a degree course that is substantially based on the suggested standardized curriculum for Software Engineering [3]. Our goal was to determine whether the students perceived that the curriculum had provided useful education for their employment in the software industry. In particular, we wished to establish (i) whether the students perceived that the BSE degree prepared them for getting their current position, and (ii) whether the areas covered in the BSE degree were relevant to the current position they held in industry. The results of this survey not only give insight into the students’ perception of the BSE degree, but also information about where we should consider adjusting the curriculum.

The rest of this paper is organized as follows: Section 2 motivates the need for a core body of knowledge for the SE discipline. Section 3 discusses various standardization efforts, in particular the IEEE Software Engineering Body of Knowledge (SWEBOK) and the Software Engineering Education Knowledge (SEEK). In Section 4, we introduce the degree structure of the BSE at Swinburne University and outline the research methodology of our survey in Section 5. We present the results of the survey in Section 6 and compare the students’ perceptions with current efforts of standardising undergraduate software engineering curricula. Finally, we conclude this paper in Section 7 with a summary of the main observations and discussion about future work.

2. Software Engineering as a Discipline – Core Body of Knowledge

In establishing the status of a legitimate discipline and a recognised profession, the stakeholders of that community must convince the community at large. Software Engineering (SE) is no exception [2].

Legitimisation of professional authority is claimed by the profession if (i) the knowledge and competence of a member of the profession can be validated by the professional community, (ii) validation is based on rational, scientific grounds, and (iii) a member’s professional judgement and advice is oriented towards a set of values. Indeed, these are the hallmarks of the well-established professions of Law, Medicine, Accounting, and Engineering. Although SE is much younger, it is now reaching the status of a legitimate engineering discipline and a recognised profession. The level of maturity of a professional discipline is often predicated on the implementation of the basic components that characterise a profession [2], i.e.

- Association of support, often by a professional society;
- Commitment to norms of conduct captured in the profession’s “code of ethics”;
- Registration to practice, often voluntary certification or mandatory licensing;
- Specialisation of skill and continued professional education; and
- Development of professional education with curricula validated by the professional society.

To legitimise SE for recognition of the profession, agreement is needed on the core body of knowledge, being the key for approving levels of knowledge (awarding of degrees), approving levels of competence (licensing of individuals), and accreditation of a curriculum. It is also a prerequisite for the adoption of coherent programs for skills development and continued professional education.

A number of professional guides have been developed for the elements of the core body of knowledge which should form the basis for developing a curriculum for professionals in SE. Validation of a proposed curriculum is then needed to show that it meets the expectations of all the stakeholders, including members of the profession, regulators, students and educators. Furthermore, we must be accountable for the educational outcomes of the those who complete professional education programs in SE. We also need to validate the graduate’s preparedness for working as a professional in the discipline.

3. Educational Frameworks – Core Body of Knowledge

In 1993, the Association of Computing Machinery (ACM) and the Computer Society of the Institute of Electrical and Electronic Engineers (IEEE-CS) set up a joint steering committee to establish Software Engineering as a profession. The subsequent Software Engineering Coordinating Committee (SWEC) now coordinates the work of three efforts: the Code of Ethics and Professional Practices [5]; the Software Engineering Education Project (SWE) [6] with its set of draft accreditation criteria for undergraduate programs in SE; and a guide to the Software Engineering Body of Knowledge (SWEBOK). SWEBOK aims to support the concept of SE as a discipline by outlining the core body of knowledge. It provides coverage of required knowledge areas for software engineers after completing their initial studies, typically after four years of practice.
In 1998, ACM and IEEE-CS established a joint curriculum task force called *Computing Curriculum 2001*. Its original charge was to “develop curriculum guidelines that would best match the latest developments in computing technology in the last decade and endure into the next decade”. The scope of this task clearly showed that no single specialty group was capable of covering all areas, since its scope has now extended from computer science alone to encompassing such independent disciplines as computer engineering, software engineering, and information systems. Thus, in an effort to address these concerns, the task force recommended broadening the project by sponsoring projects in related fields.

In this section, we examine the SWEBOK and the CCSE Computing Curriculum – Software Engineering, particularly the Software Engineering Education Knowledge.

### 3.1. Software Engineering Body of Knowledge

The SWEBOK project attempted to provide a systematic, three-phased approach (Straw Man, Stone Man, and Iron Man) that captured the core knowledge areas of SE [2]. Importantly, SWEBOK does not capture non-software engineering knowledge that a software engineer should possess. The SWEBOK team established five objectives:

1. **Promotion of a consistent view of software engineering worldwide.** This objective seeks a world view of SE by enlisting over 500 reviewers worldwide (Stoneman version). Importantly, both academia and industry were represented in the development process.

2. **Clarity of place of software engineering with respect to other disciplines, such as computer science, project management, computer engineering, and mathematics**, i.e. the boundaries between SE and other disciplines. In defining the core material for SE, SWEBOK recognizes eight related disciplines in which software engineers require knowledge. The SWEBOK guide lists 10 Knowledge Areas (KA’s) which are listed in Table 1.

3. **Characterisation of the contents of the software engineering discipline.** Based on the major schools of thought, industry, software literature, and standards, this objective seeks to decompose the key knowledge areas into topics of interest, not presuming any particular application domain. Importantly, SWEBOK provided a guide to identifying the topic areas and references, and the core body of knowledge was expounded from the reference material identified in the work.

4. ** Provision of topical access to the software engineering body of knowledge.** Reference material was identified for each of the KA’s. This material comes from a range of authoritative sources and is intended to enable a student to obtain a level of mastery of SE through the completion of a corresponding undergraduate degree.

5. **Provision of a foundation for curriculum development and individual certification material.** In order to support this objective, three simple criteria for categories of knowledge have been established: *generally accepted, advanced and research, and specialised*. Generally accepted knowledge reflects established traditional practices recommended by many organisations. Importantly, this does not mean that this knowledge should be applied uniformly to all software endeavours, but instead the individual should be capable of drawing on this knowledge for potential application. Further, generally accepted knowledge forms a part of the study material for certification after four years of professional experience. Advanced and research knowledge refers to innovative practices tested and used only by some organisations, as well as concepts still being developed and tested in research organisations. Specialised knowledge refers to practices used only for certain types of software.

### 3.2. Software Engineering Education Knowledge

In 2004, the joint task force on Computing Curricula published a final report “Computing Curriculum – Software Engineering” [3] in order to “provide guidance to academic institutions and accreditation agencies about what should constitute an undergraduate software engineering education.” Central to this document is provision and description of the body of knowledge recommended for students graduating from a SE degree (SEEK). Importantly, SEEK recognizes that non-SE knowledge is an additional body of knowledge that an undergraduate should possess. The CCSE also defines a core curriculum, i.e. ways that the knowledge outlined in SEEK and the skills fundamental to SE can be taught in various contexts.

In developing SEEK and an undergraduate software curriculum guide, it was important to define the core material considered essential for obtaining an undergraduate SE degree. By insisting on a broad consensus, the core was kept...
as small as possible to allow institutions the freedom to tailor elective components of the curriculum to meet the individual needs. It should be noted that the core is not a complete curriculum and core units are not necessarily limited to a set of introductory courses in the undergraduate curriculum. Some core material clearly must be covered only after students have significant grounding in the field.

SEEK is hierarchically organized into three distinct levels. The highest level is the education knowledge area, representing a particular sub-discipline of SE that is generally recognized as a significant part of the body of SE knowledge that an undergraduate should know. These high-level structural elements are useful in organizing and describing SE knowledge, and each area is broken down into units which represent an individual thematic module within an area. Each unit is further subdivided into a set of topics, which are the lowest level of the hierarchy. SEEK consists of the following ten knowledge areas:

- **Computing essentials:** the theory and practice of computer programming (control and data, recursion, typing, etc.) and the conceptual foundations of computer science (data structures, algorithms, etc.). It also covers construction, technologies, tools, and formal construction methods.

- **Mathematical and engineering fundamentals:** the mathematical and engineering fundamentals of SE that provide the theoretical and scientific underpinnings for the construction of well-engineered software systems.

- **Professional practice:** the knowledge and skills necessary as a foundation for practicing SE in a professional, responsible, and ethical manner, including technical communication, group dynamics and psychology, and professional responsibilities.

- **Software modeling and analysis:** not only a core concept of SE, but also of engineering. Modeling helps develop a clear understanding of the requirements and objectives upon which an engineering solution is based, and is essential in documenting and evaluating design decisions and alternatives. Requirements analysis includes an analysis of the feasibility of the desired system, elicitation and analysis of the needs of stakeholders; the creation of a precise description of what the system should or should not do; constraints on its operation and implementation; and the validation of this description or specification by the stakeholders.

- **Software design:** techniques, strategies, representations, and patterns used to determine how to implement components or systems, i.e. specification of internal interfaces, architectural design, data design, etc.

- **Software verification and validation:** use of both static and dynamic techniques for system checking to ensure that the resultant program satisfies its specification and meets the expectations of the stakeholders.

- **Software evolution:** a broad concept expanding on traditional notions of software maintenance. It covers numerous activities and issues undertaken both before and after upgrades or releases of an evolving system.

- **Software process:** the various software life-cycle process models. It focuses upon the importance of clearly defined processes and the organisation producing that product.

- **Software quality:** a pervasive concept that affects and is affected by all aspects of software development, support, revision and maintenance. It covers both the quality of the software product and the process and organisation that produces that product.

- **Software management:** the management of knowledge about the planning, organisation, and monitoring of all software life-cycle phases, issues such as costing, planning, resource allocation, risk mitigation, monitoring and control.

SEEK further assigns each knowledge area a recommended number of contact hours. For consistency with earlier curriculum attempts, a contact hour is defined as the actual in-class time required to present the material in a traditional lecture-oriented format. Each unit is also assigned a Bloom taxonomy level (Which capability should a graduate possess?) and the unit’s relevance (Is the unit essential, desirable, or optional?). Table 2 lists the knowledge areas as well as the respective core contact hours (denoted as “Hrs”).

Table 2 also lists the relative importance of each knowledge area, e.g. a percentage of 8.5% for “Validation and Verification” indicates that 8.5% of the time allocated for the core knowledge areas is to be dedicated for validation and verification of software systems. Based on the relative importance, SEEK further assigns each knowledge area a recommended number of contact hours.
importance, we have ranked the SEEK knowledge areas and added the respective ranking as the last column to Table 2.

SE education must be focused on student outcomes. The core skills from SEEK [3] require graduates of an undergraduate SE program to:

- Show mastery of SE knowledge and skills, and professional issues necessary to begin practice as a software engineer;
- Work as an individual and as part of a team to develop and deliver quality software artefacts;
- Reconcile conflicting project objectives, find acceptable compromise within limitations of cost, time, knowledge, existing systems, and organisations;
- Design appropriate solutions in one or more application domains using approaches integrating ethical, social, legal, and economic concerns;
- Demonstrate an understanding of and apply current theories, models, and techniques that provide a basis for problem identification and analysis, software design, development, implementation, verification, and documentation;
- Learn new models, techniques, and technologies as they emerge, and appreciate the necessity of such continuing professional development; and
- Demonstrate an understanding and appreciation of the importance of negotiation, effective work habits, leadership and good communication with stakeholders in typical development environments.

4. Bachelor of Software Engineering at Swinburne University of Technology

Since 1997, the Bachelor of Software Engineering (BSE) is a degree offered by Swinburne University of Technology, Australia, and was one of the first accredited SE degrees in Australia. While the degree initially evolved from a combined Computer Science and Software Engineering degree offered by Swinburne [7], its design was heavily influenced by international efforts to standardise SE education (i.e. earlier versions of both the SEEK and SWEBOK). Over the years, it had been continuously reviewed and adjusted, based on these standardization efforts. The SEEK knowledge areas discussed in the previous section now define the core of the degree.

More precisely, the aim of the degree is to give graduates a sound education in all areas of SE, focusing on analysis, design and implementation of large-scale systems, along with a sound understanding of the traditional aspects of computer science including hardware and operating systems. There is also a focus on applications involving multimedia and web-based systems, with an emphasis on the design of effective Human-Computer interaction (HCI). As such, it can be seen as an “implementation” of the proposed SE curriculum with a special focus.

The degree has accreditation with Engineers Australia, and graduates are also eligible for the professional level of membership of the Australian Computer Society. Table 3 summarizes the structure of the degree in further detail.

<table>
<thead>
<tr>
<th>Year</th>
<th>BSE Development</th>
<th>Database I</th>
<th>Computer Systems</th>
<th>Engineering</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Software</td>
<td>Introduction to Software Engineering</td>
<td>Electronic Systems</td>
<td>Math. 1</td>
</tr>
<tr>
<td></td>
<td>Development 2</td>
<td>Programming</td>
<td>Operating Systems</td>
<td>Math. 2</td>
</tr>
<tr>
<td></td>
<td>Data Structures &amp; Algorithms</td>
<td>Management</td>
<td>Systems</td>
<td>Math. 3</td>
</tr>
<tr>
<td></td>
<td>C++ for Java Programmers</td>
<td>Advanced Web Development</td>
<td></td>
<td>Math. 4</td>
</tr>
<tr>
<td>2</td>
<td>Software</td>
<td>Introduction to HCI</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Maintenance Project</td>
<td>Intelligent Agents</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Optional Industry-Based Learning Year</td>
<td>Software Testing and Reliability</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Software</td>
<td>Data Comms. and Security</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Maintenance Project</td>
<td>Engineering Minor Stream</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Final Year Capstone SE</td>
<td>Large-Scale System Design</td>
<td>Elective</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Team Project (2 Semesters)</td>
<td>Issues in IT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Engineering Minor Stream</td>
</tr>
</tbody>
</table>

As is standard for many universities in Australia, all subjects at Swinburne follow a 12 week semester, and it is expected that, on average, students spend a minimum of 10 hours per week on each subject. However, there are a number of attributes of the BSE degree that make it different from many other undergraduate SE degrees offered at Australian universities, and we briefly discuss these attributes.

In the two final years of their degree, students are required to enhance their engineering knowledge by taking a number of subjects in another engineering discipline. At present, engineering minor streams are offered in the areas of Telecommunication, Digital Electronics, and Robotics.

After three years, students have the option to undertake a supervised industry-based learning year. During this time, students work full-time as a paid employee for an organization that provides a suitable environment to gain hands-on experience in software development projects. The

---

2As a result of the re-accreditation process by Engineers Australia in 2003, year 3 of the degree was slightly modified. This is, however, not of importance for our study as all surveyed students had a curriculum very similar to the one indicated in Table 3.
organizations participating in industry-based learning vary from small software companies (fewer than 10 employees) to large multi-national firms such as major financial or telecommunication organizations.

Final year students must undertake a two-semester “capstone” SE team project. In this project, they work in a team of 12 to 15 students and develop a software system for an external client, applying the knowledge and concepts acquired throughout the earlier years of their study. The main philosophy behind the project is that students own their project, and supervisors intervene only when necessary.

5. Research Methodology

Central to any fieldwork is the development of a suitable hypothesis and the proper implementation of a suitable research methodology. This work is no exception.

This study used a survey tool to gain an insight into the perceptions of the students who have completed the BSE degree at Swinburne University of Technology, Australia. The survey is divided into four parts with an appendix. Part I addresses the background of the students, covering issues of their starting and finishing year, their project task, and the completion of industry-based learning. Part II addresses the respondents’ current work position and employer. The respondents perception of how well the BSE course prepared them for work as a software engineer is addressed in Part III. Part IV attempts to draw on the respondents perception of the course, considered against the internationally agreed body of knowledge relevant in educating undergraduate software engineers. The appendix provides an extract from the CCSE Final Report of (i) the student outcomes and (ii) the SEEK Knowledge Areas and Knowledge Units in order to help the respondents’ understanding what these core knowledge areas are. An electronic version of the research instrument can be found online at www.it.swin.edu.au/personal/jschneider/BSE.pdf.

The surveyed population came from Swinburne’s BSE graduates since 2001, a total of 42 students having completed their degrees according to our records. After gaining ethics approval for the study in July 2004, the survey tool was sent out by e-mail in August 2004 to 35 students whose contact details were still valid at the time, and 21 students responded to the survey by e-mail (i.e. a response rate of 60%). This also represents an overall participation of 50% of the study (i.e. 21 of the 42 students in the overall population). All responses were returned anonymously and the results do not identify any respondents. Only one respondent was not employed at the time the survey was completed. In October 2004, the results of the survey were tabulated and analysed to address the outlined research questions.

With a small population of this nature, it is difficult and misleading to attempt to generalise the results of this survey.

However, within the narrow confines of this subject group, it does provide an insight into the perceptions of students who have completed the BSE degree at Swinburne University. We have information on their preparation for obtaining their current position and the relevance of the areas covered in the BSE degree to their current position.

6. Results of Survey

In this section, we summarize the survey and discuss the corresponding results. As mentioned in the previous section, the first two parts contained some background information about the students (Part I), followed by information about their employment history and current position (Part II). These parts are not further discussed in this work as they are outside the scope of the research questions outlined earlier. In the following, we discuss the results of the remaining two parts in greater detail.

6.1. Readiness for employment

In Part III of the survey, we first asked the students about their perception of readiness for employment on completion of their degree. More precisely, graduates responded to:

- “I felt confident in starting work.”
- “After commencing work, my employer quickly gave me added responsibilities, realising that I was capable of working at a higher level.”

For both questions, we asked for an answer in the range of 1 (“strongly disagree”) to 5 (“strongly agree”). The corresponding responses are summarized in Table 4. It should be noted that throughout this section, we list percentages in the respective tables, e.g. a percentage of 33% agree for the first column in Table 4 indicates that 33% of the students agreed with the respective statement. Furthermore, responses in the range of 4 to 5 are considered as being positive whereas responses in the range of 1 to 2 as negative.

The results indicate that the students were very confident starting work (only 10% disagree against 33% agree and 57% strongly agree) and that their employer recognized

<table>
<thead>
<tr>
<th>Table 4. Readiness for Employment; value of Engineering knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confident in starting work</td>
</tr>
<tr>
<td>---------------------------</td>
</tr>
<tr>
<td>Employer recognized as</td>
</tr>
<tr>
<td>being capable</td>
</tr>
<tr>
<td>Knowledge gained in</td>
</tr>
<tr>
<td>engineering minor useful</td>
</tr>
<tr>
<td>Employer valued skills</td>
</tr>
<tr>
<td>from engineering minor</td>
</tr>
<tr>
<td>stream</td>
</tr>
</tbody>
</table>

In Part III, we first asked the students about

With a small population of this nature, it is difficult and misleading to attempt to generalise the results of this survey.

5. Research Methodology

Central to any fieldwork is the development of a suitable hypothesis and the proper implementation of a suitable research methodology. This work is no exception.

This study used a survey tool to gain an insight into the perceptions of the students who have completed the BSE degree at Swinburne University of Technology, Australia. The survey is divided into four parts with an appendix. Part I addresses the background of the students, covering issues of their starting and finishing year, their project task, and the completion of industry-based learning. Part II addresses the respondents’ current work position and employer. The respondents perception of how well the BSE course prepared them for work as a software engineer is addressed in Part III. Part IV attempts to draw on the respondents perception of the course, considered against the internationally agreed body of knowledge relevant in educating undergraduate software engineers. The appendix provides an extract from the CCSE Final Report of (i) the student outcomes and (ii) the SEEK Knowledge Areas and Knowledge Units in order to help the respondents’ understanding what these core knowledge areas are. An electronic version of the research instrument can be found online at www.it.swin.edu.au/personal/jschneider/BSE.pdf.

The surveyed population came from Swinburne’s BSE graduates since 2001, a total of 42 students having completed their degrees according to our records. After gaining ethics approval for the study in July 2004, the survey tool was sent out by e-mail in August 2004 to 35 students whose contact details were still valid at the time, and 21 students responded to the survey by e-mail (i.e. a response rate of 60%). This also represents an overall participation of 50% of the study (i.e. 21 of the 42 students in the overall population). All responses were returned anonymously and the results do not identify any respondents. Only one respondent was not employed at the time the survey was completed. In October 2004, the results of the survey were tabulated and analysed to address the outlined research questions.

With a small population of this nature, it is difficult and misleading to attempt to generalise the results of this survey.

However, within the narrow confines of this subject group, it does provide an insight into the perceptions of students who have completed the BSE degree at Swinburne University. We have information on their preparation for obtaining their current position and the relevance of the areas covered in the BSE degree to their current position.

6. Results of Survey

In this section, we summarize the survey and discuss the corresponding results. As mentioned in the previous section, the first two parts contained some background information about the students (Part I), followed by information about their employment history and current position (Part II). These parts are not further discussed in this work as they are outside the scope of the research questions outlined earlier. In the following, we discuss the results of the remaining two parts in greater detail.

6.1. Readiness for employment

In Part III of the survey, we first asked the students about their perception of readiness for employment on completion of their degree. More precisely, graduates responded to:

- “I felt confident in starting work.”
- “After commencing work, my employer quickly gave me added responsibilities, realising that I was capable of working at a higher level.”

For both questions, we asked for an answer in the range of 1 (“strongly disagree”) to 5 (“strongly agree”). The corresponding responses are summarized in Table 4. It should be noted that throughout this section, we list percentages in the respective tables, e.g. a percentage of 33% agree for the first column in Table 4 indicates that 33% of the students agreed with the respective statement. Furthermore, responses in the range of 4 to 5 are considered as being positive whereas responses in the range of 1 to 2 as negative.

The results indicate that the students were very confident starting work (only 10% disagree against 33% agree and 57% strongly agree) and that their employer recognized

<table>
<thead>
<tr>
<th>Table 4. Readiness for Employment; value of Engineering knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confident in starting work</td>
</tr>
<tr>
<td>---------------------------</td>
</tr>
<tr>
<td>Employer recognized as</td>
</tr>
<tr>
<td>being capable</td>
</tr>
<tr>
<td>Knowledge gained in</td>
</tr>
<tr>
<td>engineering minor useful</td>
</tr>
<tr>
<td>Employer valued skills</td>
</tr>
<tr>
<td>from engineering minor</td>
</tr>
<tr>
<td>stream</td>
</tr>
</tbody>
</table>
their skills as being of value (10% disagree against 29% agree and 52% strongly agree). Hence, it seems that the BSE degree meets the needs of both industry and the students themselves.

6.2. Value of core Engineering knowledge

One of the widely discussed questions in Information Technology is how much engineering knowledge should be taught to undergraduate software engineers. As illustrated in Section 4, the BSE degree at Swinburne contains four Mathematics subjects, followed by four subjects of one of the three Engineering minor streams. Hence, the students are exposed to a broad body of knowledge in at least another engineering discipline. It should be noted that it is a requirement from Engineers Australia for any accredited SE degree to contain such an engineering component. The motivation behind this requirement is to facilitate communication amongst engineers from various disciplines. But is this engineering knowledge really of any benefit to BSE graduates once they have started working in industry?

To gain more insight, we included in the survey the following two statements directly related to this topic:

- “I have found that the knowledge in my Engineering minor stream to be very useful.”
- “In employing me, I felt that my employer valued highly the knowledge I gained in my studies of my Engineering minor.”

As is summarized in Table 4, the responses were not in favour of the engineering knowledge. Only 29% of the students perceive the knowledge gained in the Engineering minor streams as being useful (19% agree, 10% strongly agree), but 57% perceive this knowledge as not being useful (33% disagree, 24% strongly disagree). Even worse, only 15% think that the skills from the Engineering minor stream are valued by their current employer (10% agree, 5% strongly agree), contrasting with 48% disagree and 24% strongly disagree. Although the results could be biased by the current employment of these students, it is, nevertheless, an indication that we might need to reconsider the amount of core engineering knowledge SE students are exposed to.

6.3. Skills developed/enhanced during studies

In the next section of the survey, we asked the students to comment on the degree to which they believed they developed/enhanced a number of skills during their studies. For this question, we used a scale from 1 (“very little”) to 5 (“a great deal”). Table 5 summarizes the corresponding results.

As is shown in Table 5, Coding is rated as an area where students skills were most learned/enhanced (33% much, 57% a great deal, no negative responses). This does not come as a major surprise as the degree contains four dedicated programming subjects plus a number of subjects where programming plays a major role (e.g. Real-time Programming plus the two project subjects; c.f. Table 3). Similarly, Requirements Elicitation and Documentation is rated highly (38% much, 38% a great deal), an area that is covered in considerable depth in two of the core SE subjects (Introduction to Software Engineering; Large-Scale System Design).

Teammwork is considered a skill that is of major importance for any software engineer and, therefore, highly emphasized in the two project subjects, in particular in the final year capstone project where the students learn to work in a large team (up to 15 team members). It seems that the majority of the students also perceive teamwork as an area where they considerably learned/enhanced their skills (33% much, 43% a great deal). Quality Assurance is another area that received a thumbs up from the students (57% much, 19% a great deal) as, like Teamwork, it is strongly emphasized in the final year project subjects.

More surprisingly, the students rate Design as a skill they learned/enhanced during their studies (52% positive vs. 24% negative responses). This contrasts with the perception of both the students themselves and the supervisors during project subjects (c.f. Table 3) where a deficit in design knowledge was apparent. Hence, it seems that despite the need for further enhancement, the students think that they did gain considerable design skills during their studies.

Both, Project Management and Risk Management, are areas where we do not see a clear (positive or negative) pattern in the students perceptions. Whereas Project Management has a slightly positive rating (43% positive vs. 19% negative answers), Risk Management is slightly on the negative side (25% positive vs. 43% negative), but both areas have a considerable number of neutral answers (38% for Project Management, 33% for Risk Management). It should be noted that the subject on project management was updated since the surveyed students took this subject. Nevertheless, we need to consider how both project and risk management can be incorporated in a more effective way in our SE curriculum.
It is important that a development team can react appropriately if any serious misunderstandings (or even conflicts) occur. Unfortunately, conflict resolution is a skill that is often taken for granted in tertiary education and not explicitly addressed. In our project subjects, we focus more on strategies to avoid conflicts rather than resolving them. Interestingly, in the SEEK, very little time is dedicated to group dynamics altogether (a total of 5 hours under Professional Practice). Hence, we wanted to find out how the students perceived the gained or enhanced skills in conflict resolution during their studies. As Table 5 indicates, only 20% of the students gave a positive response in regards to learning/enhancing skills in this area. Therefore, this is definitely an area that needs to be further addressed in curriculum reviews and we must investigate how conflict resolution can be best addressed within the given curriculum framework.

6.4. Value of the BSE degree

Completing Part III, we asked the students whether in hindsight, they would do the BSE degree again. The results of this question are summarized in Table 6. None of the students gave a negative response (0% no way, 0% probably not) and 24% had a neutral view, whereas 76% think that they would do the degree again (38% probably, 38% of course). Thus, we have another indication that the degree meets the students expectations on completion of their degree.

6.5. Relevance of CCSE graduate outcomes

In Part IV, we asked the students a number of questions relating to the CCSE, in particular their perception of the relevance of the corresponding graduate outcomes, and the SEEK knowledge areas. A copy of the description of the graduate outcomes (Section 3.2 of [3]) as well as the SEEK knowledge areas (Table 1 of Section 4.7 of [3]) was included as an appendix in the questionnaire, and a link to the full on-line version was also provided.

First, we asked the students to comment on their perception of the relevance of the CCSE graduate outcomes. As is summarized in Table 7, the majority of the students rated all outcomes as either being relevant or extremely relevant, with positive responses rating from 100% (working as an individual or part of a team) to 67% (designing appropriate solutions using SE principles). In particular, working as an individual or part of a team (19% relevant, 81% extremely relevant) and showing mastery of SE knowledge (24% relevant, 62% extremely relevant) were considered as being of major relevance.

6.6. Relevance of SEEK knowledge areas

As discussed earlier, we have ranked the SEEK knowledge areas based on their relative importance (i.e. based on the assigned number of core hours) and listed these (c.f. the last column of Table 2). But does relative importance correspond to industry perception?

To get some further insight into this question, we asked for perceptions of how relevant these knowledge areas are in industry, specifically in regards to the students current employment. Unfortunately, the questionnaire left some room for misinterpretation and some of the students rated these areas based on the five-point rating scheme given for the CCSE graduate outcomes, and other students ranked the knowledge areas based on their level importance (i.e. 1 being the most important, 10 the least important knowledge area). As it is not easy to translate one set of responses into the other, we discuss both, the ranking as well as the rating separately, and compare the responses with the relative importances defined by the SEEK.

Table 8 summarizes the answers of those students who rated the SEEK knowledge areas. There are three areas with 100% positive responses: Software Design (31% relevant, 69% extremely relevant), Software Modeling and Analysis (46% relevant, 54% extremely relevant), and Software Quality (54% relevant, 46% extremely relevant). A slightly lower relevance is given to Computing Essentials (46% extremely relevant, but 8% neutral and 8% not very relevant) and Software Management (38% extremely relevant, 8% neutral, 8% not relevant at all). At the bottom end of the rating are Software Evolution (8% extremely relevant, 8% not very relevant, 8% not relevant at all) and Mathematical and Engineering Fundamentals (8% extremely relevant, 15% not very relevant and 15% not relevant at all).
Comparing these results with the relevant importance given by SEEK, we can see that there are quite noticeable differences: Software Quality, for example, is rated much higher in our survey (in SEEK, it has only a ranking of 8), which to a lesser extent also applies to Software Design (SEEK ranking of 4). Computing Essentials is not rated as the most important knowledge area (but with 85% positive answers it is still rated very high). On the other hand, Software Evolution as rated quite low, in accordance with its ranking in the SEEK (i.e. rank of 10).

Most prominently, however, Mathematical and Engineering Fundamentals has a SEEK ranking of 2, but is clearly at the bottom in our survey. This is in line with our students’ perceptions that the core engineering knowledge was not of much help to them in industry (see Section 6.2). However, over a lifetime with different work experience, they may develop different views.

Table 9 summarizes the answers of the students who ranked the SEEK knowledge areas. In addition to the percentage of each ranking, we have also listed the corresponding median for each knowledge area. Clearly, Computing Essentials is perceived as the most important knowledge area (Median of 1, 75% of all responses ranked it at position 1), followed by Software Process (Median of 3, 50% of all responses at rank 2). Both Software Design and Software Quality have the same median (i.e. 4), but taking all rankings into consideration, Software Design is ranked slightly higher than Software Quality. Software Evolution (Median of 8.5, highest ranking of 7) and Mathematical and Engineering Fundamentals (Median of 9, 50% of the responses ranked it at a rank of 10) are at the bottom end of the ranking.

Both, Software Process and (again) Software Quality, were perceived by students as being quite important, whereas SEEK rated them much lower. Furthermore, Software Modeling and Analysis is not rated very highly (Median of only 6), and Software Evolution has quite a low ranking again. Most notably, however, Mathematical and Engineering Fundamentals is again rated at the bottom.

Summarizing the most prominent findings of the two comparisons, we can conclude that:

- Mathematical and Engineering Fundamentals are perceived as the least important knowledge area, in contrast to SEEK which considers it to be very important;
- Software Quality is perceived as being an important knowledge area, in contrast with SEEK which considers it to be of lesser importance; and
- Both SEEK and the results of our survey rate Computing Essentials as being a very important, if not the most important knowledge area.

### 6.7. Student Responses to Open Questions

For a number of survey questions, students also had the opportunity of adding additional comments besides a rating/ranking. To contrast with the “hard” numbers we collected during the survey, here are two of the student comments relating to the SEEK knowledge areas:

- “You cannot take away any skills in any of the knowledge areas without impacting the persons ability to perform well in that area once he/she is finally faced with it. Broad knowledge of everything at least means you are not shell shocked when faced with that issue and [it] gives you the confidence to solve problems/learn further in that area.”
- “None of these knowledge areas are unimportant, but some are more instinctual than others. For example, even though I listed Mathematical and Engineering Fundamentals as least relevant, they still require to be taught (i.e. there is no way to deduce this knowledge from common sense), whereas something like Professional Practice I feel is mostly common sense and decency. I believe the BSE degree successfully balanced the course content to reflect this.”

### 7. Conclusions and Future Work

With the growing importance of software and its role in society, it is increasingly important that suitable curricula are developed to educate young software engineers. As discussed in this paper, there are a number of international efforts to standardise undergraduate Software Engineering degrees. But does such a standardised degree meet the needs of universities, industry and students?

To address this question, we surveyed students who completed an undergraduate degree in SE at Swinburne University of Technology and have been working in industry for up to three years. This degree program is substantially based
on the results of these international curriculum efforts, and as such, offers the necessary setting to evaluate a standardised undergraduate SE degree. In our study, we particularly focused on (i) the students perception of readiness for employment after completing their degree and (ii) the industry relevance of the topic areas covered in the degree.

The results of our survey indicate that the students perceive their studies being of value, that they were confident in starting work, and that their employer(s) valued their knowledge and skills. However, there is also an indication that the knowledge they gained during their studies from other core engineering disciplines was not perceived as being valuable, from both the student and the employer perspectives. There was also a perception that skills in areas such as teamwork and conflict resolution should be addressed in more depth. Furthermore, we can also conclude that the international efforts in standardizing SE curricula are perceived as focusing on the right areas, the main exception being that Mathematical and Engineering Fundamentals are rated considerably lower by the participants of our study. Hence, our results indicate that a standardized SE degree meets students expectations, but that we should reconsider the exposure of undergraduate SE students in two areas: (i) Mathematical and Engineering Fundamentals and (ii) non-technical areas, such as teamwork and inter-personal communication, that are not treated prominently in the CCSE.

However, with the small number of students having responded to our survey, it is difficult and could be misleading to generalise our results. We need to get more data from students having done similar degrees and worked in industry for a number of years in order to get more conclusive insights. Such an extended study could not only be done within Australia, but also in North America and Europe, and we are currently investigating possible avenues to do so. Also, it is worth thinking about surveying the employers of SE graduates to get some insight into their perceptions.

Furthermore, there are other disciplines in the field of Information Technology where international efforts have occurred to define standardized curricula, and there is a clear need to evaluate the proposed curricula from the viewpoints of all stakeholders. From our perspective, we are interested in evaluating the proposed curricula in the Information Systems discipline. “Information Systems 2002” [8] is a document similar to the CCSE and forms the basis of a number of Information Systems degrees in Australia.

To conclude, it would be worth repeating this survey in two or three years time to see how the perceptions of the students surveyed for this work are changing over time.

Acknowledgements

We would like to thank all students participating in this survey as well as the Ethics Committee of the School of Information Technology for valuable feedback on earlier drafts of the survey.

References