Chapter 6

Open Source: Solution to the Information Systems Capstone Course Dilemma?

Susan Keller & John Lamp

Introduction

Many universities offer an industry-based capstone course to final year students in Information Systems (IS) programs. The purpose of a capstone course is to provide students with an opportunity to apply the knowledge and skills acquired during their undergraduate studies to a real-world situation. Typically students work in teams to complete a genuine industry project. The project involves analyzing, designing, and implementing an information system solution to meet the industry partner's business needs. Students are responsible for all aspects of the project including project management, requirements gathering, analysis, design, implementation and post implementation.

The capstone course offers many potential benefits for both students and industry partners. For students, it provides an opportunity to improve both technical and interpersonal skills and to make a contribution to the community. These benefits often enhance students' employment prospects. For industry partners, it provides an opportunity to acquire an information system for little or no cost.

However, the changing nature of IS education presents a dilemma for educators offering such courses. Information Systems at Deakin University, Melbourne, Australia, is taught within a business department and the flexible nature of the program means that students have a great deal of choice when selecting individual courses within an IS major. The trend over the last five years is that students are electing to complete only the one mandatory programming course. So while students enrolling in the capstone course have the prerequisite skills to complete the necessary requirements gathering, analysis and design phases of a
project, they do not have sufficient programming skills to satisfactorily develop a working system, especially within the tight deadline of a single semester. Open source software offers a potential solution to this dilemma.

In this book chapter, we detail the trial of open source software in our capstone course. Rather than developing a custom application for their client, students customize an existing open source solution to match the client's requirements. The challenges, benefits, lessons learnt, and learning opportunities that Free/Open Source Software (FOSS) can provide Information Systems capstone course students, are explored later in this chapter.

Capstone Courses in Information Systems Programs

Benefits of the Capstone Course

There are many benefits of a capstone course. From the perspective of technical skill development, students are able to apply knowledge gained throughout their studies to solve real-world problems. Most instructional theories and models recognize that learning is enhanced when learners have the opportunity to solve real-world problems (Dunlap, 2005; Merrill, 2002; Mills, Hauser, & Pratt, 2008). However, more importantly the program also provides an exceptional opportunity for students to develop so called ‘soft-skills’. Both government agencies and employer groups world-wide have emphasized the importance of graduates developing these skills (The Association of Graduate Recruiters, 1995; Crebert, Bates, Bell, Patrick, & Cragnolini, 2004; Department of Education and Training, 2002; Taylor, 2005). These generic skills and attributes include communication, negotiation, conflict resolution, ability to work in a team, problem-solving, initiative, self-management skills, and learning skills.

Additionally, the capstone course also provides a mechanism for universities to contribute to the community in a direct way. The projects selected for the capstone course in the School of Information Systems, at Deakin University, are drawn largely from not-for-profit community organizations. These organizations do not have the means to acquire or develop information systems even though such systems could streamline and support their operations. The capstone course can provide useful systems for these organizations. In return, the community or-
organization has an opportunity to support students in their professional development.

Finally, the capstone course enhances students' employment prospects in several ways. Firstly, in the process of completing the project, students make industry and community contacts that may lead to employment opportunities. Secondly, at the completion of the project, students also have a folio of work to show prospective employers and are able to say they have some real-world experience. However, the greatest impact is the self-confidence students gain by completing the project. Although intangible, this inner confidence may be the single most important factor in assisting students in gaining employment.

**The Dilemma**

The landscape of Information Systems programs is changing. Since the late 1990s, the discipline world-wide has experienced a sharp drop in enrolments. Accurate figures are hard to determine because many reported statistics do not clearly identify what is being counted (Granger, Dick, Jacobson, & Van Slyke, 2007). However, estimates of the decline in enrolments in US Schools from 2002 to 2004 run between 25% to 75% (Granger, 2007). A similar phenomenon was experienced in Australia and Europe. In Australia for instance, enrolments in Information Systems courses between 2002 and 2005 fell by 30.6% (Dobson, 2007, p. 23).

This huge decline is generally attributed to three main factors: the end of massive ERP rollouts; the successful resolution of the Y2K issue; and the dot.com crash (Firth, Lawrence, & Looney, 2008; Ives et al., 2002). Together, these factors led to an acute contraction in IS job opportunities, which in turn led to a severe contraction of IS enrolments. Other factors, including a worldwide economic slowdown and off-shoring of IT jobs, also contributed to the malaise (Firth et al., 2008).

The landscape of IS itself is also changing. There has been a shift in the IS function from delivering technology-based solutions to managing the process of delivering solutions (Abraham et al., 2006). Where once an entry point into IS was application programming, this is no longer the case. A combination of outsourcing, ERP systems, and modern development tools has vastly reduced the need for application programmers.
The recent release of the IS 2009 model curriculum (Topi et al., 2009) for undergraduate degrees in Information Systems acknowledges the changing professional context in which IS graduates do their work. For the first time, since IS model curricula were developed, application development is not included in the prescribed core.

If the landscape of IS and IS programs is changing, the same could be said of IS students. The trend over the last five years is that students are electing to complete only the one mandatory programming course. For instance, in 2005, 76% of our IS students at Deakin elected to complete a second programming course. However, by 2008 this percentage had fallen to 38%. Discussions with students suggest the key factors in this trend are the perception that off-shoring will make such skills redundant and the general perception that programming courses are too time-consuming.

The lack of programming skills of students entering the capstone course presents a difficult dilemma, since the production of a working application cannot be achieved without considerable programming expertise. The inability of students to produce a working system leads to two difficulties: attracting industry partners and providing students with the experience of completing a working system.

Industry partners invest considerable time in a student project. Firstly, they must set aside time for requirements gathering. Because the students are novices this time investment is greater than if they were working with professional consultants. Secondly, they must make themselves available to review the analysis and design documents produced by students and to attend various student presentations. Without the possibility of a working system, few organizations would be prepared to invest this time if the end product was 'just a report', no matter how well fashioned.

From the students’ perspective, the knowledge that the completed system will make a significant difference to the organization is a powerful motivating force. Students regularly work long hours and extend themselves in order to complete the project. From the perspective of technical skills acquisition, the complexity of a realistic implementation provides opportunities to exercise and extend knowledge across a diverse range of issues including IT governance, ethics, analysis, design, database implementation, security, networks, change management, cutover, and training. For these reasons, the provision of a working system is
important. However, the lack of programming skills now threatens our ability to offer the capstone course in the traditional way.

**The Dilemma: Possible Solutions**

Possible solutions to the dilemma are to replace the industry based component of the capstone course with a case study scenario. This would allow us to control the size of the projects. However, the industry contact provides a rich educational experience and is consistently mentioned by students as being the key advantage of the course. Another alternative is to keep the industry component and to ask students to deliver a report outlining the improvement opportunities that an information system could provide the organization. However, this would make attracting industry partners difficult. The third option is to use customizable off-the-shelf software. While this option overcomes the problem of lack of programming skills it is problematic from a funding perspective. The University could not afford to fund it, and most industry partners would be reluctant to purchase an expensive system on the recommendation of students especially since a successful outcome could not be guaranteed. The final option, and the one we are exploring, is the use of open source.

The open source movement is large and growing. For instance Sourceforge.net, the largest repository of open source software, currently has over 230,000 software projects. This provides a rich source of potential systems for students completing the capstone course. Instead of needing to write a system from scratch, students could tailor an open source solution to suit an organization’s business needs. Many open source software solutions allow creation of a customized database and significant interface configuration. This means students will have the opportunity to project manage and deliver a useful system that includes requirements gathering, systems analysis and design, selection of a solution that will meet the client’s needs, customizing the solution, installation, cutover, and training. While programming skills are not directly exercised, students will develop expertise in identifying FOSS opportunities and evaluating the fit between the business requirements and available solutions, in essence, the make or buy decision. For these reasons, we decided on trialing the use of open source in our capstone course.
Current Understanding: Open Source and Capstone Courses

A search of the literature shows there is also growing awareness of the potential for using open source software to enhance the educational experience of information technology students (see for example Bennett & Watson, 2006; Kamthan 2007; Sowe & Stamelos, 2007). However, to date there is limited discussion of the use of open source software in capstone courses. Two exceptions, in the field of software engineering (SE), are Ras, Carbon, Decker, and Rech (2007) and Stamelos (2009). Ras et al. report the results of using an open-source Wiki-based tool in their software engineering capstone course. The Wiki was used to capture and manage the students’ observations and experiences, supporting an important pedagogical concept called ‘reflective practice’ where students learn by reflecting on their actions, decisions, and their consequences. Stamelos reports on the use of open source software to provide software engineering students with practical real-world experience. Students become directly involved in various aspects of open source development, including testing, requirements assessment, documentation, and software construction. Stamelos also suggests that a worthwhile capstone project would be for a group of students to initiate their own FOSS project.

Although there is some overlap between the IS and the software engineering disciplines, Information Systems students focus not just on technology, but the people and processes around the technology. As shown in Figure 1, the approach we are taking is different from the previously reported use of FOSS within IT capstone courses. Since IS is an applied field, the needs of IS students are best met by applying open source software to an organizational context, whereas the educational needs of software engineering students may be best supported by students adding directly to open source projects as reported by Stamelos (2009).

Figure 1: Use of open source for IS students vs. SE students
Piloting Open Source

The Capstone Course at Deakin University

The fundamental principle behind the third year IS capstone course is to try to expose students to an experience that is as close as possible to life in the industry for which they have been studying. In addition, the course aims to teach the principles of project management. In some respects it is similar to most project management courses; there are lectures covering the project management body of knowledge, and there are practical requirements as well. The point of departure is that the practical components are not there to illustrate material covered in lectures, rather the course is based around students undertaking a project, and the lectures provide the necessary project management knowledge.

Students are placed into project teams of six to nine students and the composition of those teams, including the nomination of the team’s project manager, is determined by the teaching staff. The teaching staff also allocate projects to teams. Each teams’ project manager is given a one line description of their project and the name and contact details of their client. There is a framework of deliverables (Table 1) over the twelve week semester, but all other decisions – approach, tools, techniques, and implementation strategies are the responsibility of the team.

<table>
<thead>
<tr>
<th>Week</th>
<th>Deliverable</th>
<th>Assessment contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Project proposal and presentation</td>
<td>5%</td>
</tr>
<tr>
<td>4</td>
<td>System development plan and presentation</td>
<td>5%</td>
</tr>
<tr>
<td>6</td>
<td>System design walkthrough</td>
<td>5%</td>
</tr>
<tr>
<td>7</td>
<td>System design specification report</td>
<td>25%</td>
</tr>
<tr>
<td>8</td>
<td>System design specification presentation</td>
<td>10%</td>
</tr>
<tr>
<td>11</td>
<td>Class system demonstration</td>
<td>-</td>
</tr>
<tr>
<td>12</td>
<td>Programmer/system manual</td>
<td>12.5%</td>
</tr>
<tr>
<td>12</td>
<td>User manual</td>
<td>12.5%</td>
</tr>
<tr>
<td>12</td>
<td>Final presentation to the client</td>
<td>5%</td>
</tr>
</tbody>
</table>

Table 1. Course Deliverables
The size of the team and the directive nature of the team selection, team management, and project selection is also seen as reflecting industry practice.

During the semester, practical workshops are also conducted. These experiential workshops cover a broad range of issues including information gathering, negotiation, conflict management, cross-cultural communication, interpersonal communication, negotiation, team building, and decision making.

Students are provided with a comprehensive document containing guidelines on the expectations of each deliverable. The final 20% of the mark for the course comes from each individual student writing a reflective essay on a set project management topic, which must be illustrated from their course experience.

**Pilot Project**

Last year, we piloted the use of open source software in our capstone course. The target project was for a local community sexual assault centre which we will call AASA (Agency Against Sexual Assault). AASA provides counseling to community members who have experienced sexual assault and/or family violence. AASA is a small organization, but it is decentralized over a large rural and regional urban area with up to 100km between offices. Out-posted officers are mostly part-time employees and the organization of meetings was not always straightforward due to competing local demands and limitation of resources, including motor vehicles, office space and specialized equipment. AASA was using a simple spreadsheet, held by the head office receptionist, to manage all of these items, and frequent phone calls were needed to co-ordinate meetings and the use of resources. Recently, AASA had arranged broadband connections to all of its offices.

AASA’s basic requirements were for a system which could manage and provide a booking system for all its employees and resources. Ideally such a system would allow the identification of common availability of the people and resources and send advice to the people concerned when a booking was entered. Bookings need to be made up to one month in advance. Improvement in the booking system was also seen as reducing the stress on clients visiting staff at AASA. This would result in better use of resources and staff time, which is a significant
issue for a community organization. The ability to easily generate reports on resource usage for management was also seen as important.

Initially the project team evaluated a number of systems, both proprietary and open source. As AASA relies on grants for their ongoing funding, cost was a significant factor in the selection process. While AASA had a server, it was a modest machine, and most proprietary software would have required upgrades which were out of reach of the organization. A booking system with a calendar style front end was considered too sophisticated to build from scratch in the time allocated to the project. The project team identified a FOSS package on SourceForge, eGroupWare (http://sourceforge.net/projects/egroupware/) that they felt could meet the requirements of AASA. In addition to being a FOSS package itself, it was designed to run on the LAMP infrastructure (Linux, Apache, My-SQL, and PHP) which is itself a FOSS package. Specifically, the XAMPP (http://sourceforge.net/projects/xampp/) implementation of the LAMP infrastructure was used as the AASA server was Microsoft Windows based.

As with all projects using customizable software, modifications were needed to the data structure and some procedures within the eGroupWare in order to meet AASA’s requirements. Changes to the work systems within AASA were also required, probably to a larger degree than would be the case with a total build approach, as there are limits to the extent to which any package can be customized. This shifted the balance of work and nature of work required of the project team from predominantly technical to predominantly socio-technical.

The customized system was installed at AASA, training was provided by the project team as part of their expected activities, and the system has had a high degree of acceptance by the users and management of AASA. The management of AASA has now suggested a further project for development by a future student team.

**Discussion: Three Perspectives**

The trial was a success on a number of levels. The system was well accepted by the client, the students were provided with a rich educational experience, and faculty confirmed that the amount of work required of students was comparable to building from scratch, though the focus of that work was different.
We now discuss benefits, challenges and implications of the use of open source software in the capstone course from three perspectives:

- Students
- Clients
- Educators

**Student Perspective**

From student evaluations conducted at the end of the semester, students involved in piloting the use of open source were very satisfied with their experience. This may partly be explained by the fact that the project was a success. A review of all student projects undertaken over the past two years shows that only half are actually completed. Projects that fail tend to fail at the implementation stage possibly because of the decline in programming expertise reported earlier. It is often obvious to students well before the end of semester that they are on a ‘death march’ project – that is, one that is destined to fail (Yourdon, 2003). Students involved in failed projects are not as satisfied with their project experience as students on successful projects. Further, when a project fails, students are denied the opportunity to experience important aspects of the systems development lifecycle including installation, cutover, change management, and user training. Further, we know that students use the project as a means of impressing potential employees that they do have some real-world experience. A successful project is a further selling point, and indeed the client may be prepared to provide students with a testimonial following a successful system delivery. The move to FOSS based projects may increase the success rate for projects.

Using FOSS as a repository of customizable solutions, as described in this chapter, also provides students with direct experience of open source itself. This is important because the use of open source in both small and large organizations is growing fast. In 2002, an extensive survey financed by the European Commission (Wichmann, 2002) found that 43.7% of organizations surveyed in Germany were using open source, 31.5% in the UK, and 17.7% in Sweden. In 2006, an IDC survey of over 5,000 developers in 116 countries world-wide found that 71% of survey respondents use open source software, and that open source software is in production in 54% of their organizations. The IDC report concluded that open source software use is increasing so
rapidly that it ‘represents the most significant all-encompassing and long-term trend that the software industry has seen since the early 1980s’ (Gardner, 2009). From this perspective, our current Information Systems students and recent graduates will be required to add open source software into the ‘make or buy’ equation. Using open source in the capstone project provides students with this valuable hands-on experience of open source software.

However, there were some challenges for students involved in the pilot project. Since students had not studied basic FOSS principles in their courses, they were required to quickly obtain this knowledge before they began working with FOSS. This placed an additional burden on students and took up time that could have been better used on the actual project. Actually locating suitable candidate FOSS packages was also time consuming since the supplied documentation is often inadequate and students were forced to download and install software to determine its capabilities. There was also the anxiety of not knowing if a suitable candidate system would be found.

Client Perspective

The use of FOSS in the capstone course offers several potential advantages for clients. Firstly, there may be a greater likelihood of a delivered project. Secondly, there is the possibility of a more comprehensive system than would be possible for students to implement in a twelve-week semester. However, a disadvantage is that there is a limit to the degree of customization that can be accommodated in a FOSS package. This may mean clients feel pressured to change their business processes to fit the system rather than the system fitting the business processes.

The possibility of support for long-term maintenance of the system may be an advantage of FOSS for clients. There is no doubt that the systems that our students have successfully built from scratch over the years have been appreciated by the organizations for whom they were built. But, a few years later, if modifications are required, then the issue of maintenance becomes a problem. While project teams are required to provide a systems manual as well as a user manual, any future work requires significant effort to understand and work on the system. The students who worked on the project will probably not be able to be contacted. However, FOSS systems often have an active user base that can provide advice or possibly a contact who can assist.
Educator Perspective

From the educator's perspective, the initial challenges the student team faced of understanding the client's business requirements and the environment in which they operated remain very much unchanged, as did the later ones of delivery and change management. The greatest difference was identifying and evaluating packages, and the changed nature of uncertainties and risks in selecting and customizing the package.

Faculty needed to mentor students through this difficult and unfamiliar process. The SourceForge website (http://sourceforge.net/) was used to assist in identifying potential systems. The role of the site is to provide a resource to find and develop FOSS packages. As such there are many packages hosted which are a long way from being implementable: some do not have any downloadable code, some are at proof of concept, and all of these must be navigated through as part of the identification and selection process. This creates a degree of risk and uncertainty for all parties involved in the project, which is different in nature to the risks and uncertainties involved in developing a package from scratch. The success of the project hinges on whether a suitable FOSS system exists and the degree to which it can be customized to meet the needs of the client.

The client must also be carefully managed. A major issue is that the non-customized system can be demonstrated to the client, who may not understand the degree of customizing necessary to bring the FOSS package into alignment with their requirements nor the limitations inherent in this process. Therefore, helping students manage client expectations becomes crucial to the process.

Possibly the greatest benefit of offering this approach is that it provides a mechanism for us to retain the industry-placement component of the capstone course despite the decline in programming skills. As Brooks, Nocks, Farris, & Cunningham (2002, p.191) point out, "it is not possible to capture the full scope of the messy political, procedural aspects of actual practice in the confines of an academic setting."

Another benefit to being able to offer this approach as an alternative to building a system from scratch is that it provides a realistic experience of customizable off-the-shelf software. Within the Information Systems industry, there are many situations in which making an off-the-shelf software decision is desirable. The success of enterprise packages, such as SAP and JDE, is testament to the increasing use of such approaches.
Using FOSS packages provides a cost effective way of introducing this approach into the curriculum and allows faculty to demonstrate in practice many of the issues of dealing with customizing software.

From the broader educational perspective, evaluating the use of FOSS in the capstone course requires consideration of our educational aims. Universities aim to produce IS graduates with the skills and knowledge that industry requires of them, but also graduates with a variety of generic skills which can be applied to a broad range of tasks and contexts beyond the university setting. Most universities have a curriculum renewal process which is informed by local industry representatives and also by model curriculum produced by peak IS bodies in consultation with industry representatives. The latest, produced by the Association for Computing Machinery (ACM) and the Association for Information Systems (AIS), is the IS 2009 Model Curriculum (Topi et al., 2009) which can provide a compass for considering the desirable outcomes for an IS graduate.

**Educational Aims: Model Curriculum IS 2009**

IS 2009 (Topi et al., 2009) suggests that IS graduates need knowledge and skills in three related areas as illustrated in Figure 2.

![Diagram](image)

*Figure 2: Areas of high-level capabilities suggested for IS graduate from IS2009 (Adapted from IS 2009 (Topi et al., 2009))*

The three areas are:

A. Information Systems Specific Knowledge and Skills. This area is the direct focus of all IS programs, and IS 2009 (Topi et al.,
2009) articulates four high-level capabilities of an IS graduate in this category.

B. Foundational Knowledge and Skills. This area focuses on generic skills, such as communication, leadership, and critical thinking.

C. Domain Fundamentals. This area focuses on the context into which information systems are embedded.

These skills and attributes represent the destination point for IS graduates. Since the capstone course is undertaken by students in their final semester, we examine the extent to which using FOSS in the capstone course provides learning opportunities that can assist students reach these desirable graduate skills and attributes.

A. Information Systems Specific Knowledge and Skills

1. Identifying and designing opportunities for IT-enabled organizational improvement.

To identify and design opportunities for IT-enabled organizational improvement requires an understanding of the organization's strategy and goals, current systems, and business needs, as well as the ability to match the organizational needs with a feasible IT solution. Creating learning opportunities for students to gain these high-order skills is challenging because they require an amalgam of skills and abilities across a broad range of the IS body of knowledge.

Using FOSS in the capstone course may provide an especially rich opportunity for students to identify IT-enabled organizational improvement because students do not have to limit their investigations to opportunities for IT systems that could be produced from scratch in a twelve week period. Additionally, students have the opportunity to experience the complexity of a real organization and interact with a real client as they attempt to understand the opportunities that an IT enabled solution can provide for that organization.

2. Analyzing trade-offs

This attribute focuses on graduates' ability to analyze trade-offs especially in regard to evaluating sourcing alternatives in a way that considers a wide range of organizational issues as well as technology characteristics. Analyzing trade-offs is in a sense encapsulated in the previous
attribute since determining opportunities for IT-enabled improvements for an organization will involve analyzing trade-offs.

However, a key strength of using FOSS in the capstone course is that it exercises this particular skill very well. This aspect could be strengthened further by asking students to use a formal method to evaluate the tradeoffs of the competing FOSS products. There are several methodologies available in the open source arena for this purpose including the Qualification and Selection of Open Source (QSOS, 2006) and Open Business Readiness Rating (OpenBRR, 2005). These methodologies encourage a formal review of the fit between an organization’s needs and a particular FOSS product in terms of criteria such as functionality, quality, usability, support, and security. Since the importance of a particular criterion depends on the context, both methodologies include a weighting measure for the criteria.

3. Designing and implementing information systems solutions.

Although the students using FOSS in the capstone course do not produce any program code they do have the opportunity to practice their design and implementation skills in the following ways:

- Conducting a requirements analysis.
- Creating high level design documents for the customized FOSS product, for instance, use case diagrams, process diagrams, and Entity-Relationship diagrams.
- Customizing the user interfaces for the FOSS products. This involves an analysis of the existing interface in terms of usability and aesthetics and implementing any required changes.
- Implementing the required customizations for the FOSS product. This may require designing, creating, or modifying database tables and customizing functionality.
- Installing the system, managing cutover, and user training.
- Producing user and system manuals.

4. Managing ongoing information technology operations.

This skill includes the management, operation, and securing of the IT infrastructure. It encompasses all the high-level skills required of managers over a wide field of expertise, and also it implies a long-term per-
spective. Realistically, it may not be feasible to provide direct learning experiences for this skill. However, students participating in a FOSS project do have to consider issues such as security, privacy, and the long-term maintenance of the system they install. In addition, they also have the opportunity to develop their management skills by participating in the management of the project.

B. Foundational Knowledge and Skills

Moving away from direct information system expertise, the second category of graduate attributes articulated in IS 2009 (Topi et al., 2009) is functional knowledge and skills. These skills are highly valued by employers and are generally what universities refer to as generic skills. As discussed earlier, one of the key advantages of an industry-based capstone course is the development of these soft-skills. Although various class-room activities such as group-work and role-playing can provide some of these learning opportunities, the ‘messiness’ of interacting with a real client, in a real organization is difficult to replicate in the classroom.

The IS 2009 foundational knowledge and skills (apart from advanced numeracy skills) are detailed next, along with the educational opportunities afforded to students working with FOSS to implement a solution for a real client.

1) Leadership and collaboration

Students are placed in teams and assigned a leader. Although the leader has direct responsibility of leading the project, other members of the team may assume or be assigned other leadership roles within the team. For the project to be successful students must collaborate effectively. Barriers to this are lack of team skills, students from different cultural backgrounds, and the outside work commitments which prevent students from meeting face-to-face. To ameliorate these obstacles we discuss cultural differences in communication styles, structure workshops to foster reflection on team processes including conflict resolution, and discuss with students their options for keeping connected virtually when physical interaction is not possible.

2) Communication

Apart from inter-team communication, students have many opportunities to develop communication skills within the capstone course.
Students must conduct requirements gathering interviews at the client’s premises. This requires well developed interviewing and listening skills, which we practice during class workshops. From the initial phone call to the client, students understand the importance of building a constructive working relationship with the client.

Apart from team contact at the client’s premises, students also are required to present their work formally to the client, faculty, and other students at Deakin. There are two formal presentations: one to validate the design and a final presentation of the delivered system. In addition, students are required to present the project proposal and conduct a technical walkthrough for faculty. Each week, teams are also required to give an oral report in which they are asked to reflect on their progress and performance.

In terms of written communication, students are required to produce several professional quality reports including a requirements document, a design specification, a user manual, and a system maintenance document.

3) Negotiation

The capstone course using FOSS also provides the opportunity for students to practice negotiation skills both with the client and with other team members. During workshops students are provided with some tools and techniques for managing client expectations and for constructive ways to negotiate, especially around issues of scope. This is a key skill because students are inexperienced and are therefore inclined to promise more than is possible for them to deliver within a twelve week course.

4) Analytical and critical thinking, including creativity and ethical analysis

Rich opportunities for analysis, critical thinking and creativity accrue because students are responsible for all aspects of the project from project management, requirements gathering, analysis, and design, to implementation and post implementation. For most students, the project is the first opportunity they have to integrate and apply the knowledge and skills developed across the breadth of the IS courses they have studied.
Although these learning opportunities accrue from the industry-based nature of the capstone, FOSS may be the only realistic avenue to provide future IS students with this experience.

C. Knowledge and Skills Related to Domain Fundamentals

The final category of knowledge and skills in IS 2009 (Topi et al., 2009) is domain fundamentals. The rationale for this category is that IS students must be able to understand the context in which an information system is embedded.

Indeed, research shows that one of the most difficult issues for teachers of business students is that students often have insufficient business experience to understand the taught concepts as they relate to real life (Harmer, 2009). This issue was highlighted by one of our project clients who explained her frustration when working with students as they attempted to develop a payroll system: "It was really obvious some of them had no business experience at all. They hadn't had to look at overtime issues, and holiday pay and things like that. But that's a Catch 22 because they can't get out there and do it unless they've done the schooling."

There is no easy answer to this conundrum but the capstone project experience does provide students with exposure to a genuine organization and its business processes, although this does place an additional overhead on the client.

Lessons Learnt

Following the success of the pilot we are planning to move all our student projects to FOSS. Reflecting on the pilot project, we have compiled some lessons learnt which may be of benefit to other IS educators who wish to follow this approach.

- Client projects need to be carefully selected. Although the repository of FOSS products is large, it is not exhaustive. The success of the course depends on students being able to find a FOSS product that can be customized to meet the client's needs. Each year we put a call out for potential projects through a community umbrella group. There are always many possible projects to choose from, and part of the preparatory work for the course involves a faculty member contacting pro-
pective clients to establish the suitability of the project. For FOSS projects, a necessary additional step is for faculty to briefly explore the FOSS repository to establish the depth of possible solutions. The aim is not to locate a suitable solution, but only to ascertain that there are sufficient products that students could explore. This step does not take much time but ensures that students will be able to find some solutions to work with.

- Clients’ expectations need to be managed. As part of the initial contact by faculty it is important to impress on clients that since the project will be undertaken by students there can be no guarantee of success. Further, for FOSS projects clients also need to understand that there may be a limit to the amount of customization that is possible for a particular product especially within the twelve week time frame. Finally, the client must be made aware that the project is not close to finalization even though a working uncustomized package can be demonstrated.

- Provide students with background knowledge on FOSS. The capstone course is extremely demanding for students; they have a tight deadline and are challenged with applying skills to new areas, and, indeed, developing expertise in new areas. The additional burden of becoming familiar with all the intricacies of FOSS is probably an unfair load. If the curriculum does not include detailed knowledge of FOSS this needs to be included ideally before semester begins or at least in the first week of classes.

- Provide students with the tools and strategies to identify and evaluate packages. The most significant change in approaching a project is the identification and selection of a suitable package. Clear and complete user requirements are essential before making a selection. Further, grading user requirements as essential, desirable, or optional assists in the selection process. Failure to prioritize requirements in this way can result in the selection of an inappropriate package and result in project failure. While some degree of modification of work processes might be necessary to implement a particular package, efforts must be made to minimize this, and it must be done in consul-
tation with the client, otherwise there will be project failure at this stage.

From a practical perspective, schools can set up test servers for students which are configured with common FOSS infrastructures such as LAMP (Linux, Apache, My-SQL and PHP) so students can more easily test prospective systems. From the theoretical perspective students can be encouraged to use the formal evaluation methods for determining the fit between the organization and the FOSS system under consideration (see for example Qualification and Selection of Open Source (QSOS, 2006) and Open Business Readiness Rating (OpenBRR, 2005)).

- Evaluate FOSS projects differently. While the requirements gathering and analysis phases of a FOSS project are the same as a project developed from scratch there are differences downstream. In a FOSS project, design is more conceptual than detailed, and testing is much more about matching work processes with package features rather than checking coding. Faculty mentoring the project need to be aware of this and rather than being focused on compliance issues, such as unit testing or integration testing, have to focus at a more systems based level. Faculty must also be aware that seemingly trivial modifications to a package might result in significant effort by the project team. Data structure changes, for example, might result in significant code changes in a number of package modules.

**Summary and Conclusion**

The dilemma of delivering a capstone IS project management course where students are only undertaking a single basic programming course is only a dilemma when we look backwards rather than forwards. The field of IS is changing and so must our educational approaches. Re-phrasing the issue it becomes, “How can we design a capstone IS project management course which gives students a realistic experience of common practice within the industry?” Once the issue is rephrased this way, much of the dilemma is resolved. The move within industry towards customizable off-the-shelf software provides an alternative to building from scratch.

Moving towards sourcing IT solutions, rather than building from the ground up, allows students to concentrate on the sourcing decision,
project management, and the client facing skills that research shows are now favored by organizations (Abraham et al., 2006). This shift in emphasis is also reflected in the desirable graduate skills articulated in the IS 2009 model curriculum (Topi et al., 2009).

Once this realization has been accepted, the only remaining question is, "How can educational institutions provide access to a library of IT solutions that would be sufficiently diverse to meet the needs of the varied organizations that agree to participate in this educational process as clients?" The FOSS movement provides the answer to that question. Through the use of FOSS packages, educational institutions are able to provide an experience of managing a project where customizable off-the-shelf software is sourced in a cost effective manner.

We also assert that the use of FOSS packages provides additional benefits for both students and clients. By use of a FOSS package clients may receive a more comprehensive system than could have been developed by students in the same time. Clients are also likely to more easily and economically obtain ongoing support of a FOSS package. The need for accurate, comprehensive requirements very early on places greater emphasis on negotiation and requirements analysis. Thus, the importance of these foundation skills is brought into focus much more clearly.

Our experience of using FOSS packages also identifies a number of areas where necessary material is either not covered, or not sufficiently covered, in our existing pre-requisite courses. There is no detail in existing courses on the nature of customizable off-the-shelf software – what it provides, what must be provided by the team adopting it. Few students knew of the FOSS libraries. The need for a more detailed understanding of business requirements before locking the project into a particular FOSS package requires a different emphasis in the course. Further, strategies for the evaluation of customizable off-the-shelf software against client requirements are hardly mentioned in systems analysis and design courses. Proven methodologies for evaluation and selection are not widespread.

In this book chapter we have detailed the important pedagogical advantages that IS students gain when they have the opportunity to complete a genuine project for a real client. Further, we explained that the changing skill-set of IS students seemed to make coding a system from scratch problematic. The use of FOSS libraries as a source of customizable off-the-shelf software was identified as an alternative approach.
This has resolved the apparent dilemma and provided an alternative project management educational experience. It has also identified the evaluation of the matching of customizable off-the-shelf software to client needs as an area requiring further work.

This book chapter makes a valuable contribution to the discourse on the educational use of open source software in capstone courses since it describes the use of open source to support the educational needs of Information System students which to date is absent from the literature.

References


**Author Information**

**Susan Keller** PhD (Susan.Keller@deakin.edu.au) is a Lecturer in the School of Information Systems at Deakin University in Australia. Susan is the course coordinator of the capstone unit. Her research interests include information privacy and ethics, design science and higher education pedagogies.

**John Lamp** (John.Lamp@deakin.edu.au) is a Senior Lecturer in the School of Information Systems at Deakin University in Australia. He has been involved in teaching project management since coming to academia from industry in 1995. His research interests are in the area of information classification and he is best known for the *Index of Information Systems Journals*.