Providing students with 'real-world' experience through university group projects

Lynette Johns-Boast
The Australian National University, Canberra, Australia
lynette.johns-boast@anu.edu.au

Shayne Flint
The Australian National University, Canberra, Australia
shayne.flint@anu.edu.au

***Abstract**: Many tertiary institutions use project-based courses as a capstone for final year engineering, software engineering, computing and informatics students. Such courses give students an opportunity to draw together the apparently disparate learning they have undertaken during the preceding years. Students often work as members of a team, gaining team skills, as well as interacting with industry rather than academics only. In this paper we describe how we have restructured our project courses by forming teams comprising 4th year students as team leaders and 3rd year students. We discuss our experiences as well as evaluating the new course structure and its outcomes. We believe we have improved the outcomes for all stakeholders – students, clients and academics – and we have reduced significantly the work load of the academics involved, while gaining an improved ability to differentiate between students' contribution.*

Introduction

In response to demand from employers that universities produce graduates with appropriate 'employability skills' such as communication, team work, problem-solving, initiative and enterprise, planning and organising, and self-management (DEST, 2002), universities and other tertiary institutions are increasingly turning to group project courses and group work more generally to help students develop these important skills.

In the early years of an engineering degree, students learn in a largely theoretical fashion about the elements of engineering. Although some institutions use approaches such as team work and problem based learning (PBL) from early in the degree program, practice of this theory is frequently limited to artificial assignments or small-scale group projects and problems which fail to demonstrate adequately the true value of important engineering activities. Students frequently work individually and do not have the opportunity to develop the necessary team working skills along with their professional skills.

Group project courses enable students to work in teams to undertake medium-sized projects, frequently for real-world clients from outside the university environment. This enables students to take part in all aspects of the engineering process. Working in teams exposes students to the issues and problems associated with communication and team productivity.

Both practice and the literature (e.g. Beasley, 2003; Dutson et al, 1997; Oakley et al, 2004; Reichlmayr, 2006; Todd et al, 2005) suggest that project courses are highly effective in preparing students to face the real-world, as well as for consolidating learning. However, ensuring satisfactory outcomes for all stakeholders - academics (student learning, confidence in assessment), students (learning, grades, fun) and clients (product, intellectual assets) - can be a very difficult task as all three have such widely divergent requirements for success. Success for one, may not necessarily mean success for another. Additionally, assessing individual student contribution can be an even more difficult task.

Background

At the ANU all students undertaking a Bachelor of Software Engineering (BSEng) degree program must complete two, year-long project courses in their 3rd and 4th years. Students undertaking a Bachelor of Information Technology (BIT) must undertake a year-long project course in their final (3rd) year. Each course constitutes one-quarter of standard, full-time student load over the year.

In 2004 we completely restructured and combined these courses in an effort to improve further the outcomes for all involved. This paper discusses our experiences with the new course structure and evaluates the outcomes.

Course structure and assessment prior to 2004

Three separate courses existed: COMP3100 for non-BSEng students; and since 1999 and the inception of the BSEng, COMP3500 for third year BSEng students and COMP4500 for BSEng fourth year students.

Both COMP3100 and COMP3500 students undertook a *canned* project with a highly contrived set of deliverables and set schedule, while COMP4500 students carried out small projects for local business. COMP4500 deliverables were less specified and all were due at the end of the course. Students were supported via lectures covering team work and communication skills; meetings – agenda, proper meeting conduct, minutes; software as part of a larger system and the software development lifecycle (SDLC); requirements – elicitation and specification; design – architecture and program; implementation; validation and verification; post implementation review, project management – planning, estimation, metrics; and configuration management.

COMP3100 and COMP3500 students were organised into teams of up to eight students consisting of high performing, average and not so well performing students. Students self selected rigidly defined roles from: team leader, analyst, designer, lead programmer, documentation specialist, validation and verification (V&V) specialist and configuration manager. Each role had deliverables for which they were individually responsible, and from which they derived a large proportion of their marks.

COMP4500 students organised themselves into small teams of up to three students and chose their projects from details put forward by potential clients. Teams developed software for a different client.

The assessment scheme for all courses centred around the quality (as assessed by the academics) of various deliverables. These included, but were not limited to, a project plan (SPMP), work break-down structure (WBS), operational concept description (OCD), requirements specification (SRS), design documentation (SDD), test documentation – test plan (STP), test descriptions or test cases and test reports – source code, user documentation and installation instructions. COMP3100 and COMP3500 students had individual deliverables assessed according to a supplied schedule. COMP4500 students were assessed only at the end of the course.

Problems prior to 2004

The perceived problems with this model were many and varied. For 3rd year students they included:

- students carrying out their activities in parallel with little understanding that the activities were in fact dependent upon each other;
- a lack of student involvement in the complete project lifecycle because individuals received marks for the deliverables associated with their role not the overall project; .
- students more concerned with marks than the project and thus producing deliverables that they thought would gain them the highest marks rather than deliverables that had real value; they were simply looking to put *ticks in the boxes;*
- individual student work-loads that were very uneven, although they were worth equal marks;
- little incentive to do more than was required by the assessment scheme, as there was no *real* client and the developed software was not going to be used; and
- an extremely heavy workload for the team of four academics, plus a further three to six mentors.

While COMP4500 was generally more successful from all points of view, significant problems still existed. These included:

•limited exposure to all activities and responsibilities of the project development lifecycle during COMP3500 caused significant problems for some teams, eg where teams were composed of students who had the roles of documentation specialist, programmer, configuration manager.
•students who remained unconvinced of the virtue of using a plan, of estimation, requirements, and design, especially architecture. There was, however, some appreciation of the need to capture requirements, though many seemed to believe that design is wholly encapsulated within the code (having no heed to architecture) and that formal testing and reporting serves little purpose; and
•client sponsored projects were harder to source as clients did not always receive software which they could use, despite their best efforts working with their student team throughout the course.

New course organisation

From the commencement of the 2004 academic year all three project courses were integrated. At the start of each academic year clients present their projects to students, after which students indicate a preference for undertaking each of the projects. Academics then form the students into teams, based upon student preference. Overwhelmingly (over 70%) students are given one of their first three preferences with the remainder placed in a team that is within their first five preferences. COMP4500 students provide team leadership, while the COMP3100 and COMP3500 students form the team.

Goals

We believe strongly, as Dr Peter Greenwood (2003) said, that engineering is 'about using scientific knowledge, mathematical insight, analysis, creativity, practical know-how, and above all a disciplined approach to design and synthesis, to devise and construct useful things that work reliably'. To help our students understand this and develop the necessary skills we redesigned the courses with the following goals in mind. To:
•teach and to help students make and implement appropriate engineering decisions related to the development of software systems that deliver demonstrable value to clients;
•improve both student learning outcomes and client satisfaction with the quality of deliverables;
•give all students enrolled in these courses, not only the BSEng students, an opportunity to work with industry or university clients on real-world projects;
•provide an environment that is as close to industry as is possible while still within the relatively *safe* context of a university;
•provide team mentors with real-world experience;
•encourage peer learning through establishment of a community of practice;
•develop leadership capacity within students;
•increase student enjoyment of group project work;
•develop assessment that mirrors industry practice, through project reviews rather than traditional assessment of deliverable items;
•increase our ability to differentiate between individual students;
•lighten the academic load while increasing the value of the course; and last, but not least
•to improve software engineering practice through, among other things, the provision of world class software engineering education.

Structure

Teams comprise two COMP4500 students who act as team leaders with a group of between three and six COMP3100 and COMP3500 students who carry out the majority of the technical work. The size of the teams is dependent upon the ratio of third and fourth year students. Some years we have teams of six third years, while others we have had teams of only three third years. Very occasionally we have only one COMP4500 student providing leadership.

COMP4500 students manage the projects and carry primary responsibility for contract negotiation, requirements elicitation, project planning and oversight, analysis and design and the deliverables agreed with the client. All third year students are involved in various aspects of requirements elicitation, including modelling, prototyping, architecture and detailed design, testing, technical

writing, quality assurance, version control and programming. All students are encouraged to have regular contact with the client.

Academic staff and mentors fill the role of senior management in a notional consulting company with oversight of all projects. Teams meet with mentors on a fortnightly basis to report on project progress, discuss any issues or risks that have arisen and receive guidance on activities and deliverables. Frequently issues surround the management of the individuals in the team and how to improve their performance. Academic staff and mentors provide constructive feedback and general assistance on a regular, as well as, an as needed basis, to all members of the team on all aspects of the project.

Students are supported by weekly two-hour formal, semi-formal and informal, tutorial-style lectures covering topics such as leadership and team building, communication, meeting conduct, planning and tracking, scoping, requirements elicitation, requirements specification, architecture and design, risk and issues management, testing and test documentation. Guest lecturers present sessions on the 'softer' skills required for good team work, such as communication, problem resolution, team leadership and management skills.

Each year we run, depending on student numbers, from between 10 to 15 different projects. Projects are proposed by local business, government and university clients. They cover the full range of software development from basic information and web-based systems, to heavily rules based systems right through to largely experimental, proof of concept software and/or research projects. Some projects are more technical in nature than others. Projects are vetted by academic staff and not all are put forward to students to choose from. They need to include 1500-2000 hours and should not be of a critical nature to the organisation proposing them. Interest in putting forward projects has been growing each year since 2004 and we now receive more proposals than we are able to complete.

Assessment

The most significant change since 2004 is that there is now a balanced emphasis on process and output rather than output alone. The assessment scheme has slowly evolved since 2004, but in general, COMP4500 students are expected to demonstrate a high level of professional judgement and application of software engineering best practice. They need to demonstrate their ability to successfully manage the identification, development, use and evaluation of appropriate processes and artefacts required to provide real value to the client. COMP3100/3500 students are expected to demonstrate technical competence in all aspects of the software development life cycle.

At weeks 8, 14 and 23 teams undergo a project review, where students present their work for the preceding period to academics, mentors and clients. This is effectively an oral exam where students are expected to be able to produce and demonstrate work that supports their claims. Project reviews have proved very useful, sometimes with significant decisions being made by the client during the reviews. Students have coped well with them and anecdotally feel that this is both a much fairer means of assessment and is something that they actually quite enjoy.

Prior to each project review, students are required to submit a peer assessment based on work presented by Oakley et al (2004). Rankings achieved through this process are used to moderate the team mark which is awarded after each project review so that we can give individual marks that reflect each student's level of contribution to the team's work in the preceding period.

In weeks 24 and 25 all teams are required to present their working software to the class in a formal 15-20 minute presentation. During week 26, the final week of the course, we hold a *Project Showcase* to which members of local industry and government, potential clients, current clients and students are invited. Teams set up stands from which to talk about and demonstrate their software. The best two to three teams – chosen from the class presentations – make formal presentations at the showcase.

In week 15, all students are required to submit a personal report, reflecting on team processes and progress so far. They are required to consider what has worked and what has not worked so well. They are also required to consider ways in which they plan to improve their performance during the latter part of the course.

Reflection on the new course structure

Apart from student marks and the number of hours academics spend assessing we have no empirical evidence to support our claims. We can, however, make the following observations.

Overall we believe that the new course structure and assessment scheme has enabled us to meet our primary goals of ensuring satisfactory outcomes for all stakeholders − academics (student learning, confidence in assessment), students (learning, grades, fun) and clients (product, intellectual assets). Additionally, we are now able to give individual marks which are defensible and which directly reflect the contribution made to the overall team product.

Benefits

- **Assessment scheme handles any project lifecycle model** – As neither project lifecycle nor deliverables are specified, students manage their projects in a manner that is appropriate for their project and client. This is especially important for research and proof-of-concept type projects.
- **Lessened the focus on student perceived major deliverables such as SRS and SDD, and increased focus on delivering *value* to the client** – Increasing the visibility of the teams' process and assessing that process rather than the *perceived value* of the *output* of the process, means students now aim to provide value to the client rather than simply producing documents that they assume will achieve high marks.
- **Improved client satisfaction** – Increasingly, teams are delivering to clients well developed and tested software that not only meets requirements but in some cases exceeds them. In a number of instances clients have purchased IP from the students so that they can incorporate it into their corporate offerings. Clients are more forthcoming in their praise of the students. For example, an unsolicited letter received from a client after the hand-over of the team's software, said 'I consider that this project was very successful ... this system will replace the existing names application and it will also be available to the general public through our web mapping application ... Throughout the project ... the students ... were highly professional, enthusiastic, knowledgeable, courteous, friendly ... The Authority is extremely happy to continue its involvement with this scheme and as a consequence we intend to submit proposals for other projects next year.'
- **Improved ability to differentiate between students and therefore to award defensible individual marks** – Using peer assessment to moderate team marks has made allocation of individual marks and differentiation between students a relatively easy and straightforward process. In conjunction with mentor observations, it has also helped us detect and award appropriate marks to 'freeloaders' and 'couch potatoes' (Oakley et al, 2004).
- **Ability for teams to more easily handle the loss of a team member** – Occasionally a student will withdraw from the course once it has begun. Because the new structure ensures that everyone in the team is more involved with all aspects of the project, means that while the loss of a team member has an impact and causes some re-scoping, progress does not come to a sudden halt.
- **Improved student satisfaction with the course** – At the end of each course the university collects anonymous student evaluation of courses. Comments from students in all courses are fairly universally positive and have included comments such as 'Open ended – not too stuffy in terms of requiring us to do certain things'; 'Excellent exposure to problems faced in team work'; 'Most notable strength was the assessment structure ... freedom of scheduling'
- **Increased emphasis on reflection by students improves real learning**
- **Project showcase enables us to *show off* the quality of our students to the community at large** – Feedback from clients and guests at the showcase indicates that they are impressed with both the quality of the students themselves and their ability to answer questions about their project as well as the quality of the software developed. For example, an unsolicited letter of thanks from a guest said 'I would like to commend you ... on this very successful initiative and also on the high quality of students you have in this area. The showcase last week demonstrated that all the students (and projects) are of a high standard.'
- **Students are keen to do the project course –** it helps them get jobs on graduation, it looks good on their CV and also is fun.

Conclusion and future work

In conclusion, we believe that our project courses provide an environment which mirrors the real-world – we have team leaders who do not have significant experience and are virtually peers of the team members, and we have teams reporting to different levels of management. We have created an environment which encourages students to investigate, to experiment and in which to make mistakes (from which they can learn) without affecting their employment prospects. Having come from industry ourselves, we recognise the need to send our graduates out with more than just technical knowledge. All our students are graduating with experience of those very necessary skills that make them employable. We have enabled them to begin to place in their tool-box some useful approaches and activities upon which they can draw once employed in industry.

Although we feel that we have now reached a fairly stable course structure and assessment scheme we know there will always be improvements we can make. Some ideas we have are:

- costing of projects as part of scoping, *invoicing* of clients, *paying* teams on production of an invoice
- requiring COMP4500 students to write job descriptions for the positions required for their project, against which the 3rd year students need to frame their applications for those positions. Both job descriptions and applications would be assessed by the Graduate Recruitment Office who would provide students with appropriate feedback

We plan also to create a framework against which the course and future enhancements can be evaluated.

References

Bealsey, R.E. (2003). Conducting a Successful Senior Capstone Course in Computing. *Journal of Computing Sciences in Colleges, October 2003,* Volume 19, Issue 1, 122-131

Chamillard, A.T. & Braun, K.A. (2002). The Software Engineering Capstone: Structure and Tradeoffs. *ACM SIGCSE Bulletin, March 2002,* Volume 34 , Issue 1, 227-231

Dawson, R (2000). Twenty dirty tricks to train software engineers. In *Proceedings of the 22nd international conference on software engineering* (pp. 209-321). Limerick, Ireland. SIGSOFT.

DEST (2002). *Employability skills for the future: project final report*. Accessed at: http://www.dest.gov.au/archive/ty/publications/employability_skills/final_report.pdf on 30 July 2009.

Dutson, A.J., Todd, R.H., Magleby, S.P. & Sorensen, C.D. (1997). A Review of Literature on Teaching Engineering Design Through Project Oriented Capstone Courses. *Journal of Engineering Education, January 1997,* 17-28

Greenwood, P. (2003). Editorial. *Engineers Australia. April 2003*

Isomöttönen, V. & Kärkkäinen, T. (2008). The Value of a Real Customer in a Capstone Project. In *Proceedings of the 21st Conference on Software Engineering Education & Training* (pp.85-92). Charleston, USA. CSEET

Miller, R.L. & Olds, B.M. (1994). A Model Curriculum for a Capstone Course in Multidisciplinary Engineering Design. *Journal of Engineering Education, October 1994*, 1-6.

Oakley, B., Felder, R.M., Brent, R. & Elhajj, I. (2004). Turning Student Groups into Effective Teams. *Journal of Student Centered Learning, Volume 2, No.1*, 9-34

Reichlmayr, T.J. (2006). Collaborating With Industry – Strategies for an Undergraduate Software Engineering Program. In *Proceedings of the 2006 international workshop on Summit on software engineering education, International Conference on Software Engineering (pp.13-16).* Shanghai, China. SSEE'06

Todd, R.H. & Magleby, S.P. (2005). Elements of a successful capstone course considering the needs of stakeholders. *European Journal of Engineering Education, May 2005,* Vol. 30, No 2, 203-214