SOFTWARE PROJECT MANAGEMENT FOR DEVELOPING COUNTRIES

Christopher Peterson
Zenon Chaczko
Craig Scott
David Davis

University of Technology, Sydney
ICT Group, Faculty of Engineering
PO Box 123, Broadway NSW 2007, Australia

Abstract- Software is developed and implemented by enterprises that wish to increase their efficiency and effectiveness. This process is often undertaken by persons who have little or no formal training in the field, particularly in developing countries. The results are frequently disadvantageous and often fatal to the enterprise. The University of Technology, Sydney has designed a special short postgraduate program targeted at persons in developing countries who have or wish to have such software responsibility. The response to this program has proven to be significant as it provides a fast and effective approach to increasing the software project management capability.

Keywords: software engineering, computers, education, information technology, software project management

1. INTRODUCTION

For a number of years, university software education in postgraduate programs around the world has failed to explicitly recognize the need for Software Project Management specialists in the industry. This oversight is not perceived just in terms of traditional approaches to teaching and learning in which software and management topics are taught as a loose collection of subjects such as software programming, object oriented design, database theory, project management, risk management and others; but in almost completely failing to recognize the specifics and complexities of managing projects in the software industry. Enterprise software projects are not only about solving often abstract and complex systems problems but they also require ever more sophisticated tools, the integration and configuration of disparate technologies, building of product lines, a complex hierarchy of stakeholders, IP and legal issues as well as the social dynamics of highly motivated and trained collaborating specialists. In time, when the domestic software engineering industry has evolved to become a significant part of a nation's economy, practitioners have the opportunity in their work to observe and learn the essentials of good software management. No matter how advanced a country's software industry has become there is always the opportunity for improvement through formal education, for we must do better than to allow the management of software projects to be run by managers who may have only a very general and informal training in the discipline.

The process of building, configuring, maintaining, integrating and managing software by every enterprise in an efficient and effective manner is critically important for developing nations who commonly experience rapid growth of their software industry. In these countries in particular, the management of software development is often undertaken by persons who have little or no formal training in the field, usually because there is no other person to be accountable. Generally there are limited opportunities for learning in industry due to the low level of software development being undertaken. The results are frequently disadvantageous and often fatal to the enterprise.

To address these dual issues of effective software management education for practitioners in developing countries and those practicing in more developed environments, the University of Technology, Sydney (UTS) in Australia has designed a special one year postgraduate program called Master of Software Engineering Management (MSEM). Candidate students are expected to have had some experience in software technology. The response to this program has proven to be significant as it provides a fast and effective approach to increasing the software project management capability of persons, and is particularly relevant for those from developing nations. The student mix of experiences is a remarkable catalyst that supplements the delivered program. It is noted that Australia is one of the preferred choices for students because of the quality of education, language, favourable exchange rates and quality of living.

The advent of software engineering management as a "new" profession in a rapidly expanding and changing industry with its technological, social and economic characteristics meant that a new program was needed in training professional software project managers for both the local and international market. In this new approach the traditionally individual software and management subjects are taught in a unifying and coherent framework that yields a demonstrable and measurable increase in professional performance. Various practice-based subjects taught within the program concern themselves with the software project management, risk management, technology assessment, problem definition, formal and informal requirements analysis, architecture and middleware, design and implementation, testing, integration, verification and validation, delivery and maintenance, software build and configuration management. Most subjects require the student to complete project management and architectural work in teams to develop computer based hardware and software of medium to large-scale complexity. This guided teamwork has been shown to be a highly effective teaching method (Hilborn 1991; Peterson et al 1991). The main driver for the introduction of these subjects was an industry based demand for software professionals that have "real experience" in management of medium to large industry-based projects that involve considerable numbers of people, projects that are focused, process and outcome driven, and which permit experimentation with hierarchies and the dynamics of the project stakeholders. This approach was balanced against the academic and pedagogic interests of engineering and the necessary technical depth, discipline and formalism. To achieve this balance all subjects are designed to complement each other by reference to the unifying theory of the Software Development Life-Cycle (SDLC). Outcomes are focused on the depth of research, scholarship, problem solving, analytical skills, critical and reflective thinking coupled with social and communication skills including group formation, group participation, role playing and role rotation. Most critical of all is the development of the individual as a confident, coherent, ethical software management professional.

The new program fulfils industry demands by applying the theory, practice and personal development engendered in all subjects and focusing on effective management and tools, processes, delivery, schedules, budgets and risk management. This results in projects that produce high quality maintainable, reliable and usable products through appropriate management.

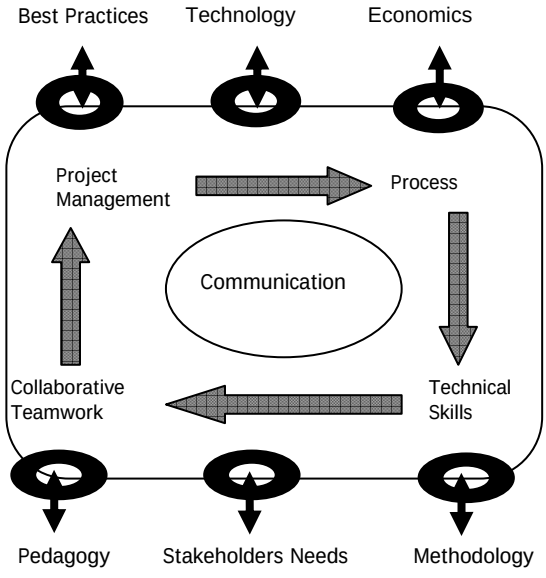

Fig 1. The domain model of teaching and learning

This report describes the experiences of teaching software project management subjects as well as evaluation of the outcomes of these experiences with the object of following a reflective process at each phase of a subject to improve the quality of learning and teaching. Also, the goal is to describe methods applied in the teaching and learning framework, which combines best practice aspects, software standards, didactics, project management, economics, the problem domain analysis, software paradigms, stakeholders concerns, technology and tools. The model can be best described as including several dimensions of an adopted reflective approach to teaching and learning, see Figure 1. Using this approach, reflections on the effectiveness, efficiency and comprehension of management and project issues involved are efficiently analyzed and resolved.

2. THE EMERGING IT INDUSTRY

Developing countries with an economy usually based on agriculture have a significant need for information technology, even if the per capita income is low. This, as always, originates from the need to use computing tools of varying degrees of sophistication, from standalone office applications to financial tools for business administration. In addition there is always the demand for large software packages, particularly from Government and public utilities where there is a concentration of resources. These packages invariably require implementation, with extensive configuration and with appropriate treatment of legacy systems, and then there is the need for ongoing enhancement. From here it is a short step to the development of novel systems to meet special needs. The demand for project managers for implementation and for development cannot always be met by consultants, particularly international consultants, so there is great pressure to appoint local staff to manage the effort. These staff would have experience in installing, configuring, interfacing and coding but frequently would not have had a management responsibility.

The educational system in virtually all developing countries supports high quality Universities with courses in Information Technology. The syllabus for technical courses is well understood and freely accessible (Bagert et al 1999) and the resulting graduates are capable in coding and system analysis. The position with software project management is quite different, for although there are many excellent resources it requires persons with advanced academic qualifications and industry experience to create and operate an effective course (Peterson et al 1991). It is the tutorials, laboratory exercises, group workshops and case studies that consolidate management issues, without which management courses do not achieve their objective of creating effective professional practitioners. It is the lack of such suitably qualified and experienced academics, coupled with the high demand for the technical graduates that are produced, that lead to the deficiency in software project management programs.

The result is that local staff are commonly appointed to manage software projects who have some technical background (not always to tertiary level), are capable and motivated, but who lack peer support or an educational infrastructure to assist them in effectively performing their task. There is a misconception that if a person is excellent at producing code then they should also be able to manage others who write code, and software is frequently seen to be ‘easy’ as a relative novice can write a short program. The basis of this misconception is that software can be so adaptable that faults are either not recognized or can be ignored for most of a project, and that the real issues are in the system specification (a skilled human undertaking) and in the project estimation and monitoring. A small project can be successful despite ineffective management; it is when projects become larger, the resources are so much greater and the loss if a project is unsuccessful that the full appreciation of good management is understood. In this situation there is an urgent demand for a relatively short and highly effective means to improve the software project management abilities of designated staff.

3. SPECIFICATIONS FOR A FORMAL IT PROJECT MANAGEMENT PROGRAM

From a survey of domestic and overseas postgraduate students studying software technology in Australia it was determined that the dominant profile of persons who would benefit from a short software project management program were those who

* hold an engineering or science degree or equivalent from a higher education institution; or
* hold a Graduate Certificate or Graduate Diploma in Engineering; and
* have some professional knowledge in the IT industry
* have adequate English skills; and
* have adequate science background (maths, physics, scientific communication at secondary school level); and
* have been involved in software development.

Some industry experience was deemed to be necessary as the students would then have some appreciation of the main issues when leading a software project.

Given this entry level it was seen that a balance was to be made between management issues and technology matters, for management decisions are frequently dependent on the technology. A course of one year full time, conducted in English, was an appropriate compromise between oversimplification and excessive duration. This permitted eight subjects to be taught, requiring a commitment by the students of approximately 32 hours a week during each of the two semesters in a year. The eight subjects fall into several groups: basic technology, current technology, engineering processes and advanced management.

The basic technology subject is required to be studied in the first semester; it provides comprehensive coverage of software development and serves a supplementary role of bringing all students to the same level of understanding.

The current technology subjects encompass analysis, design, architecture, middleware and quality topics.

The engineering process subjects include decision making in the presence of uncertainty, judgment and entrepreneurship.

Advanced management is both general engineering management and the specifics of managing software development.

4. DESIGN OF A FORMAL IT PROJECT MANAGEMENT PROGRAM

There are several design constraints when creating new subjects, foremost is that the study program must be able to be completed within 2 semesters. This mitigates against an extensive prerequisite structure, so the decision was made to have just one requirement, that the basic technology subject must be taken in the first semester of study. It was possible to design the other subjects so they had no prerequisites. The eight subjects themselves are

Basic technology: Software Technologies

Current technology: Software Quality Processes, Software Analysis and Design, Software Architecture and Middleware

Engineering processes: Judgement and Decision Making, Entrepreneurship in Engineering

Advanced management: Managing Projects, Software Project Management

Important as individual subjects are, it is the teaching and learning environment that is equally significant for the effective transmission and retention of learning. At UTS the philosophy of all engineering programs, undergraduate as well as postgraduate, is that it be practice based, where theory is informed by industrial and commercial practice (Chaczko et al 2004). This program has been recognized as embodying world leading standards and practices, the educators all having held positions in industry as well as undertaking research programs. The students find that a year devoted to study for a higher degree permits them to focus without distraction on the deeper content of the material, to assimilate it better and to reflect on their experience. All programs that are available to overseas students must comply with Federal Government regulations and the Universities need to be accredited, thus providing an additional quality process and a level of assurance to prospective students.

5. IMPLEMENTATION OF THE FORMAL IT PROJECT MANAGEMENT PROGRAM

The individual subjects are structured so that their content is not dependent on one another but they interlock, delivering a cohesive set of principles and practices once completed. Students have noted that one of the advantages of the program, apart from the topic coverage, is the flexibility it offers in timetabling and subject sequencing. The essential characteristic of each subject derives from the combination of subject matter and practical exercises.

Software Technologies: theories of software systems, ethics, professional writing and verbal communication, research skills, software engineering lifecycles, C++ and object oriented analysis and design, agile programming, operating systems and development tools, concurrency. Students are required to create a set of project plans and to undertake a group software development project.

Software Quality Processes: software quality overview, quality planning, configuration management, change control, implementation issues, coding style, verification and validation, test planning and specification, and integration. Students are required to create model quality standards and to professionally critique case studies.

Software Analysis and Design: theories of analysis, design and test, principles of system architecture, modelling system architectures/high level design against legacy systems, requirements validation, design approaches and tools, testing designs, good design and poor design, design patterns. Students are expected to work in small to medium size teams, to produce on time and to specification.

Managing Projects: project quality, risk, time and cost, techniques and practices at each stage in the project lifecycle, the management, financial and contractual responsibilities. Assessment is through group projects and presentation, project report and project artifacts.

Software Project Management: how software is different to other engineering endeavours, types of software, the triple constraint, expectations of a software project manager, time estimates and tools, cost estimates and tools, functionality including deliverables and ownership, political and human factors, group dynamics, personalities, project completion activities and technical audits. Students are to estimate in an uncertain environment, identify actions in poorly run projects, justify make or buy decisions, propose and execute remedial actions and undertake actual software project audits.

Judgment and Decision Making: rational decision aids and when to apply them, modern theories of judgment, organizational choice and decision in the context of individual, group and strategic decision making processes. Students are required to develop their critical analysis skills and apply them in a wide variety of engineering situations.

Software Architecture and Middleware: the role of software architecture, the possible and practical alternatives, selection criteria for an architecture, open infrastructure concepts and the role of middleware. The effective use of modern middleware components and the enhancement of portability, interoperability and scalability. Students' skills are developed in the analysis of open systems via a framework for their comparison and evaluation.

Entrepreneurship in Engineering: the essential components for converting a concept to a successful company by identifying opportunities, writing a business plan, raising investment and structuring an organisation. The students' strategic thinking, selling skills, service delivery, negotiation, communication, leadership and intellectual property abilities are the focus of the practical component of the subject.

6. OUTCOMES OF THE PROGRAM

Students have undertaken these subjects from a wide variety of countries including India, Pakistan, Bangladesh, Hong Kong, Thailand, Spain, Sweden, Jordan, Taiwan, Vietnam and China. Variously they have furthered their careers as academics and become team and project leaders in the software industry. The combination of overseas and domestic students, with their rich variety of professional backgrounds, brings a vitality and synergy to the program that greatly accelerates the learning process.

Through program design and delivery there are of course significant benefits to the individual. In addition to this there are a range of economic spillovers that benefit the community to which they return, benefits that accrue neither to the individual nor to their organization. These spillovers include the following, all of particular advantage to a developing country.

Import Replacement: skilled locals are able to replace international consultants, who may have a low commitment to the country even though they are dedicated to the project. This provides foreign exchange savings.

Knowledge Spillovers: indigenous project managers will disseminate their skills and methods through their professional and informal contacts, thus providing a multiplier effect of the knowledge gained through completing the program. Knowledge spread throughout the organization is directly captured through the raising of the capability of employees, yet there are contacts with others outside the company that significantly benefit the community.

Strategic Community Benefits: there is a net community benefit from improved management of critical projects through applications failing less often, or perceived to fail less often. In addition, there is net community benefit through a higher level of service when online difficulties occur. It is common that the organization is not rewarded directly through higher reliability, maintainability, usability and so on, yet these are tangible economic consequences from utilizing greater skills and capabilities.

7. PROGRAM EFFECTIVENESS

At enrolment students provide a list of problems they previously encountered on software projects. Overwhelmingly (4 out of 5 problems) these are management related issues such as inadequate and/or incomplete software requirements, poor state of documentation, lack of accountability, no statement of operational need and poor configuration control. Interestingly, the issues around system integration, which is a pertinent problem in computer systems and the IT industry as a whole, never receives much attention among aspiring software engineers. Our experience with undergraduate programs (Peterson et al 1992) verified by results of final exams, surveys and feedback from industry advisers to the program leads us to believe that graduates of our programs are able to deal with integration issues with confidence and engineering rigor.

Students were generally able to resolve specific technical problems encountered, such as tools and response times, by themselves and often pointed to their need to improve their understanding of management practices that are unique to software development.

After completion of the degree students provide a reflection of their experiences and suggestions to improve the course syllabus and delivery. A recurring comment has been that graduates have "implemented some of the practices at their workplace and are impressed with the improvement". The use of industry speakers and consolidation of material with frequent small assignments was very favourably received. The students requested greater depths on system implementation, integration and testing; and improved means of judging the size of a project before development commenced so as to provide more confidence in estimates. Requests for a greater emphasis on Entity–Relationship (ER) analysis and database issues that encapsulate aspects of business processes and less on real-time and data flow diagrams indicate areas where the bulk of system development activity is undertaken in industry and commerce. The feedback from graduates has been evaluated and wherever feasible, incorporated through changes on emphasis, modification of practical exercises and new representative case studies.

8. CONCLUSIONS

The Master of Software Engineering Management program, through its design and implementation, is achieving its objectives of providing accessible, effective software management education to professionals at an early stage in their career. At this point there is the maximum leverage for individual, organizational and national benefits. From the number and variety of student nationalities it is clear that one of the prime objectives, to cater for the needs of developing countries, is being achieved is a practical and effective manner.

REFERENCES

Bagert, D., T. Hilburn, G. Hislop, M. Lutz, M. McCracken, S. Mengel (1999) Guidelines for Software Engineering Education Version 1.0, *Carnegie Mellon University Software Engineering Institute CMU/SEI-99-TR-032, ESC-TR-99-002* http://faculty.db.erau.edu/hilburn/se-educ/99tr032.pdf

Chaczko Z., D. Davis, V. Mahadevan (2004) New Perspectives on Teaching and Learning Software Systems Development in Large Groups. In *5th Internat Conf on IT Based Higher Ed and Training ITHET '04* p278

Hilborn, S. (1994) Team Learning for Engineering Students. In *IEEE Trans Ed* **37** (2) pp207-211

Peterson C., C. Drane, J. Leaney (1991) Systems Engineering for Large Groups. In *6th Annual Conference of Australasian Association for Engineering Education*, (Parr and Johnston (Eds)) pp 541-547

Peterson C., C. Drane, J. Leaney, N. Carmody, K. Fung, A. Ginige, P. Mallon, R. Meegoda, S. Murray (1992) Reflections Upon a Completed Computer Systems Engineering Degree. In *4th Annual Conference of Australasian Association for Engineering Education*, (Simmons, Radcliffe and Wallace (Eds)) pp 334-337