Information Systems and Technology Education:
From the University to the Workplace

Glenn R. Lowry
United Arab Emirates University, UAE

Rodney L. Turner
Victoria University, Australia

Information Science REFERENCE

INFORMATION SCIENCE REFERENCE

Hershey · New York

Acquisitions Editor: Kristin Klinger
Development Editor: Kristin Roth
Senior Managing Editor: Jennifer Neidig
Managing Editor: Sara Reed
Assistant Managing Editor: Sharon Berger
Copy Editor: Holly Powell
Typesetter: Jennifer Neidig and Jeff Ash
Cover Design: Lisa Tosheff
Printed at: Yurchak Printing Inc.

Published in the United States of America by
Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue, Suite 200
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-pub.com
Web site: http://www.igi-pub.com/reference

and in the United Kingdom by
Information Science Reference (an imprint of IGI Global)
3 Henrietta Street
Covent Garden
London WC2E 8LU
Tel: 44 20 7240 0856
Fax: 44 20 7379 0609
Web site: http://www.eurospanonline.com

Library of Congress Cataloging-in-Publication Data

Information systems and technology education : from the university to the workplace / Glenn R. Lowry and Rodney L. Turner, editors.

p. cm.

Summary: "This book presents a multifaceted, global view of the human dynamics of education, supply, demand, and career development in the information systems and technology industry. It provides a tool to meet the challenges of providing improved education and employing an optimal supply of information systems and technology graduates in the decades to come"--Provided by publisher.

Includes bibliographical references and index.

ISBN 978-1-59904-114-8 (hardcover) -- ISBN 978-1-59904-116-2 (ebook)

1. Information technology--Vocational guidance. 2. Information technology--Educational aspects. I. Lowry, Glenn R. II. Turner, Rodney L.

T58.5.I528 2007

004.023--dc22

2007007278

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

Chapter VIII
Aligning Learning with Industry Requirements

Jocelyn Armarego
Murdoch University, Australia

ABSTRACT

A review of studies of practitioners of software development reveals a depth of mismatch between their needs and formal education. The conclusion to be drawn is that industry has made a long-term shift in its requirements of graduates from technical subjects, laying emphasis on personal and affective attributes. Concern has been expressed that the underlying "socialisation" requirement for a graduate to achieve "working professional" status is very poorly addressed in formal education. After establishing a framework for comparison between information technology (IT) formal education and industry requirements, this chapter discusses an action research study based on applying nontraditional and innovative learning models to address mismatches identified. Results suggest that models which focus on independent learning and soft skills prepare students to enter industry with the ability to engage in the career-long, professional learning required for success in professional practice.

INTRODUCTION

Software development has been described as a "craft." The negative connotations of this label include an inability to consistently guarantee a quality product, fit for the purpose for which it was developed, produced on time, and within budget. As an example, a mid-1990s study of over 8,000 projects (Standish, 1995) indicates only 16.2% of software was successful. These rates do not significantly differ from those reported in the 1970s and 1980s (Mann, 1996). The issues that underlie this state-of-affairs (namely, intrinsic difficulty, uniqueness of each system, multidisciplinary skills necessary, and a requirement for life-long learning in practitioners) are described later on.

A review of major model curricula for software development (e.g., information systems [IS], computer science [CS], and software engineering [SE]) shows that, in general terms, a graduate within the broad IT discipline should emerge from formal education with knowledge of the basic software development processes (and therefore, in theory, be able to produce successful software). While practitioner studies indicate that the base case of content knowledge is covered in models used in university programmes, a closer look reveals the depth of the mismatch between practitioner needs and formal education in software development in general.

Engineering Software

Those involved in the development of software agree that one mechanism for dealing with the intrinsic difficulties (e.g., complexity, visibility, and changeability [Brooks, 1986]) of developing successful software was to embed its production within an applied science environment. Royce (1970) was the first to note explicitly that an engineering approach was required, in the expectation that adhering to a defined, repeatable process would enhance software quality.

This interest in engineering is mirrored in the education of software developers, with an exponential growth in offerings of undergraduate software degrees within an engineering environment. Increasingly, this education focuses on process and repeatability, modelling scientific and engineering methodologies. The underlying assumption of this approach is that "good" software development is achieved by applying scientific investigative techniques (Pfleeger, 1999).

Creating Software

There are positive implications as well for the label "craft." Each system is considered a unique synergy between the hardware, software, and organisational context in which it will be utilised. This approach suggests that the development process cannot be repeatable, as the forces at play will differ for each context; continually changing as understanding of the characteristics of the developing system grows in all stakeholders.

From this perspective software is a collaborative invention. Its development is an exploratory and self-correcting dialogue (Bach, 1999), based on insight-driven knowledge discovery (Guindon, 1989) facilitated by opportunistic behaviour (Guindon, 1990; Visser, 1992).

The risk is that strict adherence to engineering and science methodologies hampers the quintessential *creativity* of this process (Lubars, Potts, & Richer, 1993; Maiden & Gizikis, 2001; Maiden & Sutcliffe, 1992; Thomas, Lee, & Danis, 2002). These, potentially:

- Restrict essential characteristics such as opportunism (Guindon, 1989)
- Assist in adding accidental complexity through their attempts to control professional practice (by restricting natural problem solving, Sutcliffe & Maiden, 1992)
- Impose a plan at odds to inherent cognitive planning mechanisms and hence interfering with the management of knowledge (Visser & Hoc (1990) suggest that, in practice, a plan is followed only as long as it is cognitively cost-effective)

Practicing Software

The skills and knowledge required to be active as competent professionals are multidisciplinary. For software development, Zucconi (1995) suggested the underlying disciplines of central importance are psychology, CS, and discrete mathematics, and suggests an IT professional needs to be well organised, able to work as a member of a multidisciplinary team, and able to work within the scope of the employer's policies and procedures and society's tenets.

This equates well with the stated needs of practitioners. Practitioner-based studies (Lee, 2004; Lethbridge, 2000; Trauth, Farwell, & Lee, 1993) and in the Australian context (Scott & Yates, 2002; Snoke & Underwood, 1999; Turner & Lowry, 2003) assist us in building a profile of a practicing IT professional.

Teaching Software

Freed (1992) coined the term "relentless innovation" to describe the capacity to invent and implement new ideas that will impact every facet of life. Oliver (2000) suggests the rate of innovation is so prolific that most of the knowledge that will be used by the end of the first decade of the 21st century has yet to be invented. The speed with which technology evolves, the multiplicity of its impact on society, and the ramifications of that impact mean that metacognitive and knowledge construction skills as well as adaptability become vital. This relates to a fourth issue that needs to be addressed: the need to engage in life-long learning. Professional practitioners with such skills become *agents of change* (Garlan, Gluch, & Tomayko, 1997).

However, Patel, Kinshuk, and Russell (2000) argue that learners in a traditional setting predominantly constitute students who focus on skills that will yield higher grades as an immediate objective. Cognitive skills related to "exam techniques" acquire importance, though they do not model real-life situations. The learning, in many cases, is reduced to assignment hopping with "just-in-time" and "just-enough" learning to fulfill the assessment tasks. This defeats the (academic) objective of providing a well-balanced learning experience.

RESEARCH/EXPERIENCE QUESTIONS

The aim of this chapter is to provide:

- An overview of the dominant pedagogy for formal education in IT disciplines
- An overview of practitioner studies undertaken over the last 15 years, in order to establish a base for comparison between IT formal education and industry requirements, described in relation to the dominant curriculum models in IT

The chapter continues by discussing the potential of nontraditional and innovative learning models to address any mismatch identified. The need to engage with complexity, the holistic nature of the domain, and the focus on higher learning outcomes imply a commensurate need in teachers to apply to the learning environment the principles they are advocating in their students, namely, flexibility, adaptability, and creativity.

The question:

How useful is the knowledge generally included in tertiary institution curricula for the practicalities of being an IT professional?

and based on the results of this:

Do alternate learning models address any mismatch identified?

Can these be applied successfully within a formal (tertiary) education environment?

are tackled from the perspective af an action research project conducted over several years within engineering at Murdoch University.

Curriculum Expectations

A comparison of the major model curricula undertaken as part of Minor's (2004) study of requirements engineering (RE) practitioners shows that, in general terms, the base case of discipline knowledge identified in practitioner studies is covered in models used in universities. Table 1 provides a summary of this comparison. A look at generic IT (as opposed to specific RE) suggests Minor's conclusion can be extrapolated—most bodies of knowledge (BOK) and model curricula address discipline content comprehensively.

Table 1. Minor: Curricula match to perceived industry needs

Topics	**CC-CS**	**CC-IS**	**CC-SE**
Discipline Content			
RE process	o	-	o
Feasibility study	-	o	-
Elicitation	+	+	+
Analysis	+	+	+
Documentation	+	+	+
Verification	+	-	+
Requirements management	-	o	o
Other Software Topics			
Process standards	+	+	+
Project management	+	+	+
Programming languages	+	+	+
Generic Skills			
Communication skills	+	+	+
Team skills	+	+	+

Note. RE = requirements engineering. Legend: + = extensive coverage; o = partial coverage; - = minimal or no coverage.

However, nontechnical skills are usually addressed at a more abstract level and often in association with ethics, management, or social concerns. For example, while the Australian Computer Society's core BOK for IT professionals (Underwood, 1997) indicates coverage of group 1 is mandatory (group 1 material relates to interpersonal communications; ethics/social implications/professional practice; and project management and quality assurance), little assistance in addressing these within a programme of study is provided.

Some insight into the skills and knowledge required for software development activities is provided by the studies described next, albeit from many different perspectives. This closer look at practitioners reveals the depth of the mismatch between industry needs and formal education in software development in general.

Practitioner Perspective

Practitioners of Information Systems

Summarising his work of the previous 8 years on the knowledge requirements and professional development of young IS workers Lee (1999) found that:

- Significant gaps exist between what industry expects IS workers to know and what universities teach IS students.
- The knowledge and skills required change, so that that the ability to learn quickly on the job was critical to IS workers. A wide range of nontechnical skills were identified as important to IS professionals.
- IS workers need not have a technology-relevant degree.
- IS workers draw heavily from a "bi-polar" knowledge structure—most current techni-

cal knowledge and localised team-centric project work, but are unable to exploit tacit organisational knowledge outside their specific project.

In a later work Lee suggests there is an underlying socialisation requirement for a graduate to achieve working professional status. Lee found that one of the "reality shocks" involved in the socialisation of new graduates to work was the onus of teaching themselves what they needed to know in order to perform the task successfully.

He concluded:

... educators should also help students to develop their initiatives and abilities to deal with ill-structured problems. This would require approaches which emphasize independent learning and collaborative teamwork. (p.135)

Other studies of IS confirm a change in emphasis to both generic attributes and managerial knowledge—a long-term shift from programming and other technical subjects to business analysis and people-oriented skills.

Practitioners of CS and Engineering

Fewer studies address the skills and knowledge needed in CS and SE. Lethbridge (2000) examined industry perception in a comprehensive study: His aim was to gain a practitioner ranking of the usefulness of specific topics compiled from the curricula of (emerging) software and computer engineering and CS programmes, the influence of these on respondents' career, and how much they had learned formally compared to what was required as a professional.

Although he found few surprises, an indication of differing educational focus is provided by pronounced bi-polar distribution in his data: *Leadership* and *Negotiation* ranked 3rd and 4th for industrial knowledge, while *Technical Writing* and *Analysis & Design Methods* rank as having the 5th and 6th most pronounced bi-polar distribution in education (Lethbridge, 1999).

Of the long list of topics that managers consider more important than developers at large, the high ranking of both RE or analysis-related topics and more generic skills is significant (see Table 2). Unfortunately, many of these appear to have been learned on the job (see Table 3). At least in this case it can be seen that teaching does not reflect the needs of the practice.

Lee (1986, 1992) also looked at the long-term professional development of young engineers as technologists, in studies reported in the late 1980s and early 1990s. What was found to have significance was:

- Challenging work

Table 2. Lethbridge rankings: Most important for managers

Rank	Topic
1	Project Management
2	*Requirements Gathering & Analysis*
3	Giving Presentations to an Audience
4	Management
5	Ethics and Professionalism
6	Analysis & Design Methods
7	Software Architecture
8	Leadership
9	Testing, Verification & Quality Assurance
10	Technical Writing

Table 3. Lethbridge rankings: Difference between formal learning and importance

Rank	Topic	% difference
1	Negotiation	84
2	Configuration & Release Management	83
3	Leadership	73
4	Maintenance, Re-engineering & Reverse Engineering	72
5	HCI/ User Interfaces	67
6	Software Reliability & Fault Tolerance	64
7	Ethics and Professionalism	63
8	Project Management	63
9	Management	61
10	*Requirements Gathering & Analysis*	60

- Approach to information seeking in order to keep up with the relevant changes in knowledge and information requirements
- Success of the transition from an academic environment and the formation of social ties with veteran colleagues

These results indicate that the effective preparation of young technology workers involves far more than just a fixed set of academic subjects.

The Australian Perspective

Other research looks at the situation in an Australian context. Snoke and Underwood's (1999) study sampled a wide cross section of the IS academics in Australia, including representatives from all universities offering an undergraduate degree in IS or with a major in IS as of July 1998. It showed that personal and group attributes are consistently more highly valued than technical knowledge competencies.

The aim of the Turner and Lowry (1999) study was to achieve a better fit between university study and the professional practice of IS. Their survey found that employers lay heavy emphasis on personal attributes, though this may be because technical skills are generally addressed comprehensively within formal education. A follow-up survey to explore the "other skills" aspect of IS curriculum (Turner & Lowry, 2003) shows that, in general, respondents rate soft skills higher than "hard" academic skills.

Scott and Yates (2002) report on the experience with engineering graduates as one of the parallel series being undertaken in various professions across Australia and New Zealand. The study sought to identify:

- Capabilities that are seen to be most important for successful professional practice in engineering during the first few years after graduation
- Extent to which the universities at which the participating graduates had studied focused on these capabilities

Respondents noted that learning profession-specific content provides the "scaffold" for the important task of career-long professional learning: The skills to undertake this are of great importance, with the ability to know when and when not to deploy technical expertise, and how to continuously update it, the keys to successful professional practice. The supervisors in the study acknowledged that a high level of technical expertise is *necessary but not sufficient* for successful practice, giving emphasis to the individual's ability to diagnose what is really causing a problem and testing solutions in action.

In summary then, industry requires personable professionals who integrate into the organisational structure, and, rather than cope specifically with today's perceived problems, have models, skills, and analytical techniques that allow them to learn, evaluate, and apply appropriate emerging technologies in a collaborative environment.

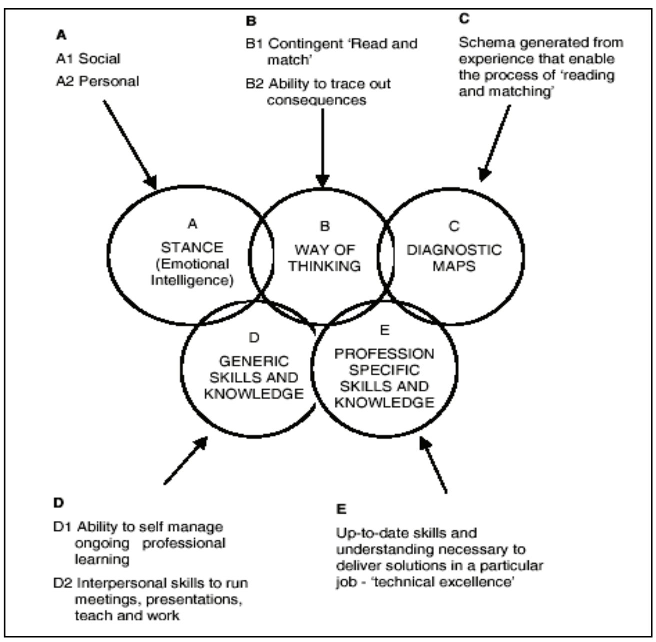
The implications of this include initiative, ability to deal with complexity, and ill-structure and organisational (self, task and information) skills. The value of these softer, more personal attributes has been explored through several studies within our target disciplines.

Exploring Affective Attributes

Bentley, Lowry, and Sandy (1999) suggest a developmental process in which personal attributes, which influence intellectual abilities and skills, are applied to the acquisition of knowledge to enable the development of higher cognitive activities. They note that at the end of the educational process, students must be able to apply knowledge to new situations and problems. This requires certain generic intellectual abilities and skills, which, although highly valued by employers, are sometimes given only "lip service" in tertiary education curricula. The personal attributes identified as important in the model proposed include attributes like curiosity, risk taking, personal discipline, and persistence. These can influence in important ways the successful application of the intellectual skills and abilities to knowledge to support the higher orders of thinking.

Scott and Wilson (2002) and Scott and Yates (2002) confirm the value of these attributes. They discuss their findings in relation to a framework that identifies professional capability as five scales consisting of:

Figure 1. Professional capability framework (Scott & Wilson, 2002)



- Emotional intelligence – Personal (EI-P)
- Emotional intelligence – Interpersonal (EI-I)
- Intellectual capability (IC)
- Profession-specific skills and knowledge (Prof)
- Generic skills and knowledge (Gen)
- Educational quality (EQ) scale

In Scott and Wilson (2002) the Professional Capability Framework is refined (see Figure 1): *Emotional Intelligence* (personal and interpersonal (now social)) becomes *Stance* and *Intellectual Capability* is now defined by two components, *Way of Thinking* (incorporating cognitive intelligence and creativity) and *Diagnostic Maps* (developed through reflection on experience).

Respondents to their studies were asked to rate items from the capability scales on their importance for successful performance in their current professional work and then to rate the extent to which the university they attended focused on them. The results of these studies show that *Emotional Intelligence* ranks highest in importance, closely followed by *Intellectual Capability*, addressing generic issues such as abstraction and contingency, while profession-specific knowledge ranks relatively low. The ability to work in teams, particularly cross-disciplinary teams that are common in the IT workplace, is also considered vital.

Issues with Traditional Learning

A review of these studies indicates practitioners emphasise attributes additional to profession-specific knowledge and skills. These latter are generally addressed by the content of a BOK and applied within model curricula, as is initial competence (i.e., cognitive attributes Bloom, 1956), though whether at an appropriate level is a moot point.

Practitioner concerns have been addressed through interventions in the learning environment and have taken several forms. In general, however, these have been attempted within the framework of traditional learning, and, according to practitioners, soft skills are still not emphasised enough. Lowry and Turner (2005) suggest that tradition and inertia act as some of the formidable barriers to substantive revisions to curricula in line with the findings of practitioner-based studies.

Learning IT

Macauley and Mylopoulos (1995) acknowledge that a standard university lecture cannot achieve what industry requires. For them, activities associated with efficient software development "*require a certain level of knowledge and maturity which can only be gained through experience in dealing with practical problems.*" Others also note the inadequacy of formal education in training competent software professionals (Lethbridge, 2000; Robillard, 1999). Bach (1997) stated that one reason software engineering is not more seriously studied is the common industry belief that most of the books and classes that teach it are "impractical": it does not address the inherent cognitive complexity (Robillard, 2005) of software development.

As exemplified by the model curricula, approaches to learning IT tend to emphasise technical knowledge. In general this education is based on traditional learning models and adheres to a normative professional education curriculum (Waks, 2001). Students first study basic science, then the relevant applied science, so that learning may be viewed as a progression to expertise through task analysis, strategy selection, trial, and repetition (Winn & Snyder, 1996).

As Waks (2001) explains, in this normative model science provides "*a* ***rational foundation*** *for practice*" [original emphasis], with practical work at the last stage, where students are expected to apply science learned earlier in the curriculum to real-life problems. The addition of either a capstone project and/or an industry-based placement, typically towards the completion of the qualification, have been seen as a means of addressing general practitioner concerns, providing opportunities for both authentic and experiential learning. Waks continues that the crisis of the professions arise because real-life problems do not present themselves neatly as cases to which scientific generalisations apply.

This poor fit between the characteristics of professional practice and those of the learning model produce an "incorrect" learning environment, where the learner is not directed to the important features of the domain, where, as Bubenko (1995) notes:

- Complexity is added to rather than reduced with increased understanding of the initial problem
- Metacognitive strategies are fundamental to the process
- Poblem-solving needs a rich background of knowledge and intuition to operate effectively
- A breadth of experience is necessary so that similarities and differences with past strategies are used to deal with new situations

Aligning Learning with Domain Characteristics

The nature of software development (complex, Nguyen & Swatman, 2000; cognitive, Robillard, 2005; opportunistic, Guindon, 1989; creative, Thomas et al., 2002; emergent, Budgen, 1995) implies a need to transcend traditional education and focus on flexibility, productive thinking, and creativity-enhancing activities. In this way, while students learn to use past experience on a general level, they are also able to deal with each new problem situation in its own terms. The implication of this is effort spent on higher (metacognitive) learning skills, including abstraction and reflection.

Schön (1987) looked to an alternative epistemology of practice when attacking the normative professional education curriculum discussed previously. For him, practitioners apply tacit knowledge-in-action, and when their messy problems do not yield to it, they "reflect-in-action," and in the languages specific to their practices. This view of professional practice as ill-structured design has implications:

- It is learnable but not didactically or discursively teachable: it can be learned only in and through practice.
- It is holistic: its parts cannot be learned in isolation. It must be learned as a whole because all components of a situation have meaning.
- It depends upon the ability to recognise desirable and undesirable qualities of the discovered world. But novice students do not possess this ability, and it cannot be conveyed to them by verbal descriptions, *only* in the operational context of the task.

For Schön (1987) the ideal site of education for reflective practice is the design studio, under the close supervision of a master practitioner serving as coach.

Others (Boud, 1985; Spiro, Feltovich, Jacobson, & Coulson, 1991) also argue against traditional learning:

- Learning based around constructivist principles is likely to be more suitable in domains involving ill-structured problems.
- Appropriate learning in ill-structured domains and/or dealing with ill-structured problems should itself be problem based.
- Problem-based learning best provides an effective environment for future professionals who need to access knowledge across a range of disciplines.

Problem-based learning (PBL) is one example of learning environments that embrace these ideas. It integrates the learning of content and skills in a collaborative environment, and emphasises "learning to learn" by placing great responsibility for learning on the learner (Wilson & Cole, 1996). As an ideology, PBL is rooted in the experiential and action learning traditions advocated by Schön (1987) and others, but with a number of different forms according to the nature of the field and goals of the learning situation: for example, Schön's design studios exemplify Savin-Baden's (2000) PBL model for professional action. This focuses on *know-how*, which will allow students to gain competence to practice within given discipline frameworks and is seen to apply within curricula that have strong links with industry and are influenced by the community of practice.

Its supporters claim PBL results in increased motivation for learning, better integration of knowledge across disciplines, and greater commitment to continued professional learning (Boud, 1985). As well as offering the flexibility to cater to a variety of learning styles, the emphasis moves from dealing with content and information in abstract ways to using information in ways that reflect how practitioners might use it in real life (Oliver & McLoughlin, 1999).

The purpose of the research described in the next sections was to apply and evaluate alternate learning models based on PBL in order to ascertain their success in addressing the mismatch identified between practitioner needs and formal tertiary education for software development.

RESEARCH DESIGN

The contrasting philosophical and epistemological assumptions implicit in natural science and social science research approaches have been described and discussed at great length in a number of widely cited works (e.g., Bunge, 1984; Guba & Lincoln, 1994). The assumption, that cognition and understanding is not a thing located within the individual thinker, but is a process that is distributed across the knower, the environment in which knowing occurs, and the activity in which the learner participates, is fundamental to how this research is conducted.

Research for Action

Action research is proposed as a means of meeting this need for contextual research for action. It combines theory and practice through an iterative process of change and reflection and has been categorised into several types, based on the underlying assumptions and world views of the participants (Carr & Kemmis, 1986; Grundy, 1982).

Several models exist for undertaking action research in education, based on Lewin's (1946) concept of a spiral that incorporates a cycle of problem diagnosis, action intervention, and reflective learning, all leading to continuous improvement of practice and an extension of personal and professional knowledge (Zuber-Skerritt, 1995). Action research places the teacher in the dual position of producer of education theory/policy and user of that theory through their practice. Within IT research, action research is celebrated as unique in the way it associates research and practice (Avison, Lau, Myers, & Nielsen, 1999). Although a survey of the literature shows that the IT academic community almost totally ignored action research (Avison et al., 1999 report only 29 articles on action research, spanning the years 1971 to 1995), by the end of the 1990s it began growing in popularity for use in scholarly investigations of IS, spurred by the relevance of research results.

A Framework for Action Research in IT Education

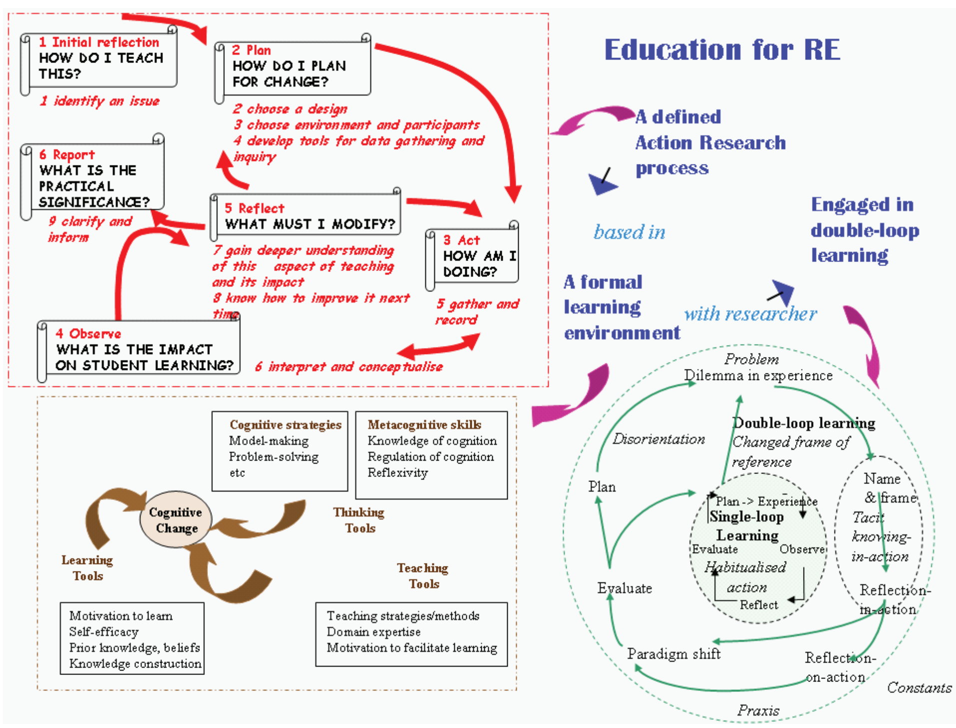
The model of action research applied to this study is adapted from the work of Borg, Gall, and Gall

(1993). However, a model for the research design is not sufficient: The context of the study suggests that the action research be placed within a conceptual framework that reflects the "culture" of the organisation in which the study is conducted. The context of this study is an institution of (formal) tertiary education, therefore requiring an acknowledgement of theories of learning as fostering cognitive change through the construction and organisation of knowledge.

The learning that takes place is not confined to the student participants in the research being undertaken. The value of action research is its ability to focus on the researcher's learning as a fundamental component of the context under investigation. The framework for double-loop learning proposed by Hatten (1997) provides a basis for consideration of the researcher's participation within each action research cycle. In this model, single-loop learning is a characteristic of a stable context in which problem solving is patterned on proven solutions and previous experience (Argyris & Schön, 1974). Double-loop learning, in contrast, is seen as transformative: required by a context where change is inevitable but its direction unpredicted. In this environment reflection becomes the basis for decision making that relies on intuitive and tacit knowledge and critical analysis. Informed, directed, and committed action (thus Praxis) requires reflective activity in order to change the frames of reference by which action is taken.

The dominant characteristics of this study suggest that a conceptual framework for action research in IT education requires each of these components to be incorporated. The process is a defined action research model, the context an environment where the aim is learning (cognitive

Figure 2. Conceptual framework for action research for cognitive change



change). The iterations explicit in the research design require double-loop learning on the part of the researcher at least, so that future action is based on varied reflection. Figure 2 illustrates this framework.

Data Acquisition and Evaluation

This research is undertaken with an acceptance of the view that not only is education a social discipline, but so is the (knowledge/discipline) domain into which the students expect to enter. The study adopts a mixed method approach as the most appropriate for the development of multiple interpretations, guided by the concept of complimentarity, reflecting the intention to use the results of one strand to elaborate, enhance, and illustrate the results from the other strand. The value of this nested concurrent approach (Creswell, 2003) is that it provides broader perspectives than by using the predominant method in isolation: Here the predominant approach is qualitative but containing smaller quantitative data collection phases. And, since change is accepted as a fundamental goal of this research, an evaluation strategy that applies a qualitative approach to the collection and analysis of data is seen to have the potential to provide the information required.

Data Acquisition

Kember and Kelly (1993) divide observation techniques common to action research in education into three categories:

- **Diagnostic:** These devices include student assessment, learning inventories, interaction schedules, diagnosis of conception (e.g., mind maps).
- **Records:** Records include such items as diary/journal and supporting documents including syllabus, documents for course development and accreditation, student assessment planned
- **Feedback:** Formal and informal questionnaires provide a mechanism for participants to address areas of interest to the researcher, while interviews allow for both general impressions to be uncovered and tight analysis to be undertaken.

These are applied in this study to provide mechanisms to validate the interpretations made through a process of triangulation (Denzin, 1970)

Data Evaluation

The action research strategy adopted utilises both formative and summative evaluation techniques but also allows for monitoring analysis to take a prominent position: The integration of data collection/data analysis allows the research to be shaped and reshaped by the participants in the research, based on the themes identified through examination of the data. This thematic analysis aims to identify important elements, with the categories induced from the data itself. According to Orona (1990), the value is in the approach's acceptance (though not reliance) on intuition and creativity, nuances and detail.

PRESENTATION OF RESEARCH RESULTS/EXPERIENCE

Since 1995, Murdoch University Engineering (MUE) has provided a suite of programmes in SE. The teaching objectives have focused on producing graduates with a special skill in software: We expect our graduates to find career opportunities in both professional engineering industries that have a strong interest in software as well as in IT disciplines where the design and implementation of quality software is considered a priority.

The investigation into characteristics of learning and of the discipline, described in previous sections, has suggested that the issues highlighted as either practitioner or domain needs of formal education could be best addressed through less traditional approaches to learning, with a focus on advanced knowledge acquisition (Spiro et al., 1991). An education framework for software development should exploit the learning models that provide an appropriate environment for practice. It should: be based on constructivist theory with a focus on strategic knowledge; be placed within a situated experiential environment where authentic context is exploited; and provide learners with exposure to activities that allow students to act opportunistically and creatively.

Pursuing these objectives has meant a gradual shift from more traditional learning, resulting in the development of a studio learning model. Based on a PBL approach, integrating Schön's (1987) ideas on design studios with creativity-enhancing activities, this learning model has been seen to provide students with a solid foundation in subject matter, while at the same time exposing them to real-world characteristics. Within the constraints of professionally accredited curriculum, studio learning addresses the issues previously described:

- An increasing focus on scientific generalisation as the education of choice for software
- Potential for misalignment with industry needs
- An acknowledged need for life-long learning

The curriculum for the Bachelor of Engineering, Software Engineering BE(SE) at Murdoch University originally integrated units[1] in three primary components:

- **Computer science:** These cover fundamental aspects and form the basis of technical knowledge and skills in software and hardware
- **Software engineering:** These focus on SE theory and practice and form the basis of core knowledge and skill in software development and evolution
- **Engineering:** These offer knowledge and skills in engineering practice and principles and are common to all our engineering students. They include natural sciences, mathematics, management, and ethics, which provides the basis for:
  - **Engineering internship/thesis:** This is also common to all engineering students, though the domain of application targets the appropriate discipline of study. The internship is wholly industry-based in that the student is an "employee" of the organisation. The thesis may also be linked to workplace experiences, but the student is not employed during the duration of the project.

As can be appreciated from this brief description, the learning environment adheres very closely to the traditional model described by Waks (2001).

The reduced opportunity for group-based projects due to the introduction of the semester-long internship/thesis was one trigger for the restructure of some of the core SE units. Other triggers included a need to provide students with a taste of the types of "messy" problems they would encounter during their internship. Exposure to the uncertainties, inconsistencies, and idiosyncrasies associated with real problems would enhance graduates' potential to deal in their own turn with ill-structured problems within an organisational context.

Cycle 1: Engaging with Authentic Practice

In this cycle, the focus is split between two units—a final semester final year (Sem 2, Year 4) unit that treats students as novice professionals (G477[2]), and the effects on students placed in this environment of the prerequisite unit (G260), which treats students as apprentices.

Students as Novice Professionals (G477)

A design studio model was applied, anchored by a PBL process based on Koschman, Myers, Barrows, and Feltovich (1994) and in the context of the phased development of a software product. As described elsewhere (Armarego, 2002), student evaluation undertaken in weeks 4, 7, 11, and 13 highlighted student concerns:

- The need to learn new content as well as adapt previous knowledge
- Dependence on other members of the team (10+ students) both for achieving the tasks and for critical assessment components through peer and self assessment
- Lack of stability in teams and task (students were rotated into and out of teams, roles, and problem component) requiring a need to "come up to speed" very quickly at each change

However, benefits were identified:

James	*we learn so much "practical" stuff from this project, it would be good to get another chance to actually do it right*
Chad	*learnt a lot about design skills and approaches for problems*
Sam	*interesting group experience*
Brad	*you need more practical application of the theory you teach ([this unit's] style)*

The restructured unit was seen to provide students with a number of opportunities (Armarego, 2002):

- To identify, analyse, and solve a number of issues, repetitively. This acts as preparation for professional employment.
- To practise the art as well as science of SE in a controlled setting.
- To test the understanding of theory, its connection with application, and develop theoretical insight.
- To deal with incompleteness and ambiguity.
- To think independently and work cooperatively, fostering insight into individual strengths and weaknesses.

However, an unexpected problem was encountered early in the semester. While students were accepting of the idea of directing their own learning in a capstone project and thesis environment, they felt (very strongly, at times) that within a formal unit they should be *taught*: They were comfortable with the concept of a "master" there to oversee their every action. This perception could be traced back to a reasonably high level of teacher direction in prerequisite SE units, confirmed through analysis of teaching style and a review of the introductory unit (G260) based on Reeves (1997). The instrument developed by Reeves provides 14 dimensions for the evaluation of technology-assisted learning. This review indicated a *transitional* approach to teaching, which did not challenge students' expectations of traditional learning. The initial student resistance to the environment provided in the final unit showed that these expectations were still evident 2 years later in their studies.

Students as Apprentices (G260)

The learning environment for the introductory SE unit (on RE) at this time was based on a *cognitive apprenticeship* model (Collins et al., 1989). In cognitive apprenticeship settings, the teacher models effective practices within professionally relevant contexts: the students are presented with tasks they would undertake as practicing professionals, requiring proficiency with notations and tools, but also an appreciation of the context in which these must be applied. This requires an understanding of the underlying conceptual frameworks used in the domain. Because these skills are all new to students, the teacher closely coaches them, to apply a process for modelling each task as they reason about the issues being raised. Whenever the students reach an impasse and are unable to continue or complete the task independently or with assistance from group peers, the teacher can "take over" by once again modelling the appropriate approach, often in a protocol analysis environment, for all students. Gradually, students are required to complete tasks more independently, with the final class assessment item requiring the development of a complete model of a problem, with critique and justification of the approach taken, with minimal support from the teacher.

The curriculum for this unit was addressed as a two-cycle spiral: The first part of the semester (8-9 weeks) focused on learning the use of the tools, gaining an understanding of the conceptual framework (in this case object-orientation principles), and an appreciation for the context in which professionals practice (e.g., historical overview; issues in RE theory and practice; organisational involvement; group dynamics). The second part of the semester focused on issues of group work and knowledge transfer—students are involved in a group project that requires them to apply the tools to model a complex problem. In broad terms, the phases (see Table 4) of the cognitive apprenticeship model are traversed throughout the semester, though without a clean break—the focus of the class sessions changes, but the ability to revisit any phase as required exists.

Evaluation of the model was based on elements of assessment (which included mind maps to provide some information on conceptual understanding, portfolios that provided information on student's willingness to explore outside the boundaries provided within the unit, and hence transcend the unit material) and student feedback, both formal university-wide and school-based, open-question surveys.

Evaluation of the cognitive apprenticeship model in relation to practitioner characteristics indicated that although this model addressed some components of industry needs, the fit between characteristics of action in the discipline

Table 4. Phases of cognitive apprenticeship model as implemented

Phase	Component	Class Sessions	Activities & Teacher Role
I	Modelling	6	• Demonstration of a task as a process • Example approaches and sample solutions provided as basis for comparison and critique • Teacher explains strategies applied and use of modelling tools (e.g., notation) explicitly
II	Coaching	10	• Critique and whole-class discussion of individual approaches applied • Focus is on exploration of multiple perspectives and the reasoning process
III	Scaffolding	4	• Teacher's role is to question, prompt, and encourage students to stay on task
IV	Fading	6	• Student collaboration and peer discussion lead to a negotiated solution for submission

and those of the learning model exhibited elements of an "incorrect" learning environment. The environment exhibited some of the traits of surface learning—students focused on learning the tools and techniques of RE at the expense of a more expansive view of the discipline: They did not see themselves as acquiring the more generic skills valued by practitioners, with the majority of students focused on passing the unit. When the teacher "faded" early in the subsequent unit (G477), students were loathed to take ownership of their learning—they insisted they had a "right to be taught."

The outcome of this initial phase was to confirm the need to build into the curriculum a focus on generic skills as part of the outcomes of the unit, with the intentions of improving students' learning ability, developing employability skills, and preparing for lifelong learning. To maximise effectiveness, these needed to be embedded into the knowledge base constructed by the student during the unit. This has the advantage of enabling students to develop the requisite skills situated within the learning context but, of course, potentially required extensive adaptation of the existing material.

The conclusion reached was that the master/apprentice relation needed to be down played so that students took early control of their own learning.

Cycle 2: Creative Software Development

Reflection on the learning experience highlighted a need to emphasise student-centred learning earlier—the final year was too late. This led to a review of prerequisite units, with a view to making pedagogical changes early in the SE curriculum. Opportunities to focus in greater depth on issues raised in the discussion of education for software development were also identified. One additional issue could also be tackled: IT education has historically been plagued by a lack of integration—the methods, techniques, tools, and so forth acquired within a few isolated units do not permeate the students' approach to other software-related tasks within their programme of study. An attempt could now be made to present a more holistic approach within the SE curriculum.

This second cycle focused on the first SE unit encountered (G260), at that time still early (Sem 1, Year 2) in the 4-year programme. The core component—RE—provided an appropriate environment for attacking student expectations of a learning environment.

Education for REs based on traditional learning models tends to emphasise technical knowledge and is based largely on notations and prescribed processes (Nguyen & Swatman, 2000). Although Budgen (2003) suggests this is a requirement of the software domain, it is at odds with the inherent characteristics associated with real RE problems, which imply a need to:

- incorporate *creativity*-enhancing activities within the curriculum,
- foster *adaptability* in students by providing for divergent as well as convergent thinking, and
- focus on metacognitive strategies and reflection as an aid to *transfer* of the skills and knowledge learned.

The implication of this is the explicit development of metacognitive strategies and the ability to reflect *in* as well as *on* action. The value of metacognition is confirmed in the recurring findings from Scott's work on applying a professional capability framework (previously discussed). A focus on flexibility and productive thinking is also necessary, so that students learn to use past experience on a general level, while still being able to deal with each new problem situation in its own terms. Gott, Hall, Pokorny, Dibble, and Glaser (1993) posit that this adaptive/generative capability suggests the performer not only knows the procedural steps for problem solving but *understands when to deploy them and why they work*, in effect is wise in the use of them.

Glass (1995) suggests that discipline and creativity are the "odd couple" of software development—the discipline imposed by methodology, for example, forms a frame for the opportunistic creativity of design. The educational dilemma becomes one of providing an educational base that enables software developers to both create and engineer the systems they build: to be adaptable to the changing environment that is inevitable in their chosen discipline. Cropley and Cropley (1998), however, suggest that the process of creativity and innovation is poorly understood in engineering and not adequately fostered in undergraduate teaching. This deficiency results in an engineering culture that is frequently resistant to the factors that promote creativity and innovation.

The Place of Creativity

Albert (1996) notes that schooling at the age of starting formal education emphasises logical rather than divergent thinking, with the value of conventional behaviour, well-defined problems and good grades emphasised. In addition, many cultures (here we may say discipline-based as well as social) encourage respect for the past and discourage disruptive innovations. Promoting widespread creativity raises expectations that may change employment patterns, educational systems, and community norms.

Amabile's (1996) general theory of creativity suggests three components:

- **Domain relevant skills:** The more skills the better, and the ability to imagine/play out situations
- **Creativity-relevant processes:** Including breaking perceptual (the way you perceive a situation) and cognitive (the way you analyse) set and breaking out of performance "scripts," suspending judgement, knowledge of heuristics, adopting a creativity induced work style (e.g., tolerance for ambiguity, high degree of autonomy, independence of judgement)

Figure 3. The CreativePBL model

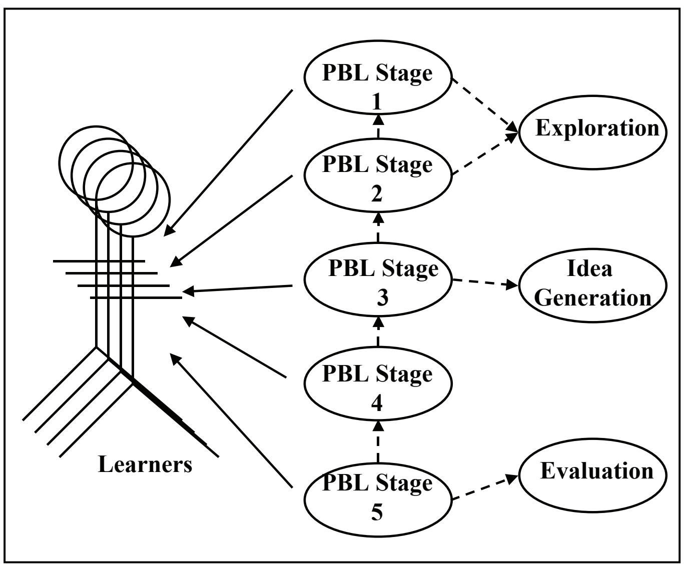


Table 5. Positive influences for enhancing creative potential

Encouraging assertion of ideas
No reliance on order and training
No fear of failure
Providing time and resources
Developing expertise
Giving positive, constructive feedback that is work or task focused
Encouraging a spirit of play and experimentation
Providing a mix of styles and backgrounds with opportunities for group interaction
Making a safe place for risk taking allowing free choice in task engagement
Offering rewards that recognise achievements or enable additional performance but maintain intrinsic motivation rather than controlling behaviour

- Intrinsic task motivation, which are necessary for the enhancement of creative potential

The PBL process applied in the design studio model in G477 provided an environment that could be adapted/enhanced for the development of creative potential. Table 5 lists some of these, based on Amabile's (1996) work.

A CreativePBL model (Figure 3) was developed to address the characteristics of software development (specifically RE in G260) as a domain and to provide a learning environment that enhances the opportunity for creative and divergent thinking. The prime motivation, therefore, in changing the learning environment was to address the issues identified previously as an "ill-fit" as early as feasible within the student's programme of study and to challenge the false expectations students had of learning through less traditional approaches.

The congruence between Edmonds and Candy's (2002) elements of creativity (see Table 6) and the PBL stages of Koschmann et al. (1994) enabled creative activities to be embedded into the PBL process. In addition, other properties of PBL were seen as relevant to the specific domain (RE) tackled in the introductory SE unit:

- Its problem solving requires the mental representation of problematic situations—the problem space (Newell & Simon, 1972) must be constructed, either individually or (of more relevance in RE) socially through negotiation.
- Active manipulation of the problem space is required for PBL problem solving and involves information gathering, model building, hypothesis generation, speculation, and solution testing, among others. This engages conscious activity, and in successful problem solvers, leads to more systematic manipulation of the problem space.

Within the CreativePBL framework the focus is firmly on examining the problem at length rather than quickly solving it. There is evidence that students who have been taught to explore different ways to define problems engage in more creative problem solving over the longer term (Baer, 1988), addressing flexibility and adaptability issues raised by practitioners. The

Table 6. Creativity activities (Edmonds & Candy, 2002)

Exploration of ideas, knowledge, and options is based on • breaking with conventional expectations, whether visual, structural, or conceptual is a key characteristic of creative thought • immersion—the complexity of the creative process is served well by total immersion in the activity • holistic view—the full scope of a design problem is only fully embraced by taking a holistic or systems view. The designer needs to be able to take an overview position at any point and, in particular, to find multiple viewpoints of the data or emerging design important • parallel channels—keeping a number of different approaches and viewpoints active at the same time is a necessary part of generating new ideas. Exploration involves accessing source data that may be examined, assessed, and interpreted in terms of the goals. This is an open process, possibly without observable directions, but the thoroughness and selectivity of the activity is critical. Having a comprehensive set of knowledge sources readily available is extremely advantageous. Knowing where to look and how to select the knowledge is even more important.
Idea generation—problem formulation, as distinct from problem solving, is critical to the effectiveness of the solution space that is defined. It draws upon a wide range of analogous cases often outside the immediate domain. This has been characterised as an ability to make remote associations. Creativity is demonstrated by the generation of many potential solutions instead of gravitating quickly toward a single and (usually) familiar solution that is not necessarily the optimal one. The ability to consider parallel lines of thought and to select and transform the results to meet the demands of a different situation is a critically important aspect of solution generation.
Evaluation involves taking the results of the generative activity and testing the candidate solutions against a set of constraints. This leads to modifying, reformulating, or discarding solutions depending on the feedback. Selection of the optimal solution may involve a number of trade-offs against the constraints that are applied especially where, as is usually the case, the product is a complex one. The application of tight constraints may be considered conducive to creative solution finding and thus evaluation is a vital part of the creative process. Evaluation may be viewed as a pervasive activity that takes place from the exploration phase onward. The use of expert knowledge in evaluation has been identified as an important aspect of successful solution finding.

model was developed to focus on creativity and divergent thinking, so that, instead of students aimed at finding the single, best, "correct" answer to a standard problem in the shortest time (convergent thinking), they aimed at redefining or discovering problems and solving them by means of branching out, making unexpected associations, applying the known in unusual ways, or seeing unexpected implications.

Evidence from qualitative and quantitative evaluations[3] of this environment (Armarego, 2005) indicates that while some deep learning is exhibited students are still "hedging their bets" by focusing some of their learning strategies on learning for reproduction. This suggests that further work is required in building an appropriate learning environment that provides students with the ability to transcend imposed frameworks, whether those of disciplinary boundaries or of personal stance.

Table 7. Savin-Badin – Model IV: PBL for transdisciplinary learning

knowledge	the examining and testing out of given knowledge and frameworks
learning	critical thought and decentring oneself from disciplines in order to understand them
problem scenario	characterised by resolving and managing dilemmas
students	independent thinkers who take up a critical stance towards learning
facilitator	an orchestrator of opportunities for learning (in its widest sense)
assessment	opportunity to demonstrate an integrated understanding of skills and personal and propositional knowledge across disciplines

Within the context of education for software development, Savin-Badin's (2000) model IV may provide an appropriate framework for learning. As Table 7 summarises, in this model students are encouraged to develop an autonomous position as individuals within the group, and as a group, and implies an evaluation of one's own stance and openness towards the stance of others. Students take a critical position towards knowledge, themselves, and their peers and elect to use the group to resolve dilemmas. A learning environment based on this model enables students to deal with problems within a metacognitive-rich framework that makes complexity apparent and lets students deal with it explicitly. The challenge for the teacher is to focus on quality of product and provide feedback to the group, as well as facilitate the process.

Although generally considered beyond undergraduate learning, this model appears to reflect more closely the skills required to undertake software development and therefore provides both a challenge and a goal in the context of undergraduate education for IT.

While much has been written regarding the value of PBL in learning, (e.g., Boud, 1985; Wilson & Cole, 1996), undertaking such a project comes at a cost:

- **Content:** Guidelines for implementing PBL indicate that success is partly based on a reduction to the content covered: assuming too much content is a pitfall in a PBL environment (Albanese & Mitchell, 1993).
- **Time to develop project:** Bridges (1992) suggests that each PBL project requires 120 to 160 hours to construct, field-test, and revise. To this figure should be added technical effort when the problem is developed in an online environment.
- **Cost:** PBL is economical for classes of less than 40 students (Albanese & Mitchell, 1993). It is considered not to scale well to large student numbers without a greater increase in staffing resources.
- **More time to teach less content:** Albanese and Mitchell (1993) also suggest 22% more time is required to teach in PBL mode, despite the reduction in content usually advocated. In their study, when academic staff consider the hours per week in preparation to teach problems in comparison to presenting lectures, instead of 8.6 hours/week primarily preparing lectures, staff spend 20.6 hours/week primarily in groups with students.
- **Diffculty in transitioning, both for staff and students:** Bridges (1992) suggests academic staff are uncomfortable withholding

Table 8. Thomas: Issues in flexibility and creativity

Issues identified by Thomas et al	Addressed in this context
Individuals or groups do not engage in effective and efficient processes of innovative design. As examples of *structuring failure*, people typically fail to spend sufficient time in the early stages of design: problem finding and problem formulation, then often bring critical judgment into play too early in the idea-generation phase of problem solving. As another example, empirical evidence shows that peoples' behaviour is path dependent and they are often unwilling to take what appears to be a step that undoes a previous action even if that step is actually necessary for a solution (Thomas et al, 1977)	Problem analysis is a critical stage: starting from the unknown and progressing to a description of the problem itself, and the knowledge needed to deal with it is fundamental to RE. Problem-solving habit is challenged by the need to generate alternate solution paths. In learning RE this problem analysis is a critical outcome.
Evidence suggests individuals have a large amount of relevant *implicit knowledge* that provides alternate perspectives to a problem. Providing appropriate strategies, knowledge sources or representations can significantly improve an individual's effectiveness in problem solving and innovation (Thomas et al., 1977)	The value of alternative perspectives is fostered through participation in a collaborative environment and the active promotion of critical friendship. Critical appraisal and self-appraisal skills are developed through the use of reflection tools such as the 4SAT (Zimitat & Alexander, 1999)
The appropriate level, type, and directionality of motivation are not brought to bear.	Although external motivation is difficult to eliminate within an undergraduate degree, PBL is seen to foster intrinsic motivation through the authenticity of the tasks undertaken (Wilson & Cole, 1996).

information as they watch students struggle with problems and need training to develop facilitator skills or they may be unsuccessful in PBL. Students may be uncomfortable with the extensive collaboration required or with the lack of teacher direction given.

However, despite these costs, the Creative-PBL approach also had the value of addressing issues identified by Thomas et al. (2002): They suggest there is a widening gap between the degree of flexibility and creativity needed to adapt to a changing world and the capacity to do so. Table 8 summarises these issues and indicates the approach taken to address them within the framework developed.

Evaluation of this model indicated that student conceptions of the characteristics of the learning environments were related to their study orientations and strategies. Meaning-oriented students were likely to see their learning environment with positive terms such as having good atmosphere and demanding deep learning, while reproduction orientation was associated with the view that the learning environment demands surface learning and requires students to be overworked.

These findings give support to the contextual view of student learning: Study approaches or orientations are formed in the interaction between individuals and their environment. Figure 4 summarises student perception of their learning in this environment and confirms results from other evaluation instruments.

Although a great deal of effort went into developing the CreativePBL environment, students needed greater preparation in order to tackle a different learning model (e.g., a better understanding of the PBL process) and support structures (examples, guidelines), so that they have a clear indication of the appropriateness of their learning.

Therefore, while the CreativePBL model provided some insights to student learning, ultimately it is a process-oriented approach, implying process is of greater importance than the product (Dahlgren, 2000), and that problem solving is a smooth process of sequential stages. To some extent, this was an inhibitor to student engagement with the learning environment—so much effort was expended in applying and monitoring the process rigidly (a novice characteristic) that students did not transcend characteristics of surface learning, nor, in particular, allow opportunism and heuristic insight the importance it was warranted.

The aim of the next cycle was to improve the proportion of students using aspects of deep learning approaches and to downplay the *process* of learning (to some extent), while still focusing on *reflection* on learning in order to balance the importance of both process and product on professional practice.

Cycle 3: Agents of Change

Studies of the design process indicate the importance of opportunistic approaches (Carroll & Swatman, 1999; Guindon, 1990; Khushalani & Smith, 1994), based on the catastrophe cycle illustrated in Figure 5 (Nguyen & Swatman, 2000) rather than a smoothly evolutionary problem-solving model. The catastrophe cycle can be compared to classical Wallas' (1926) model

Figure 4. Learning in a CreativePBL environment

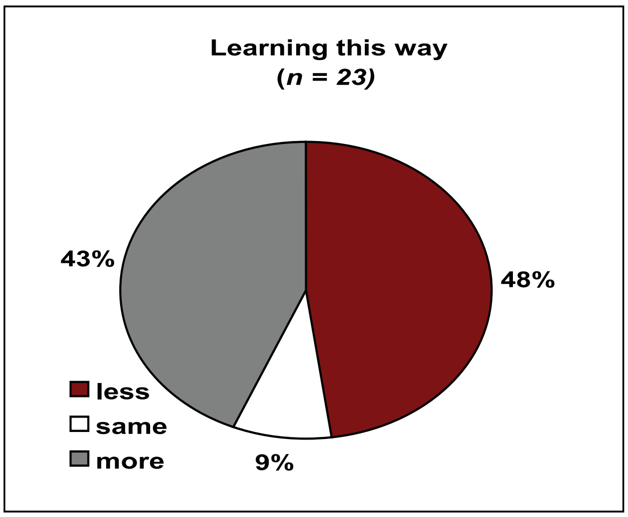
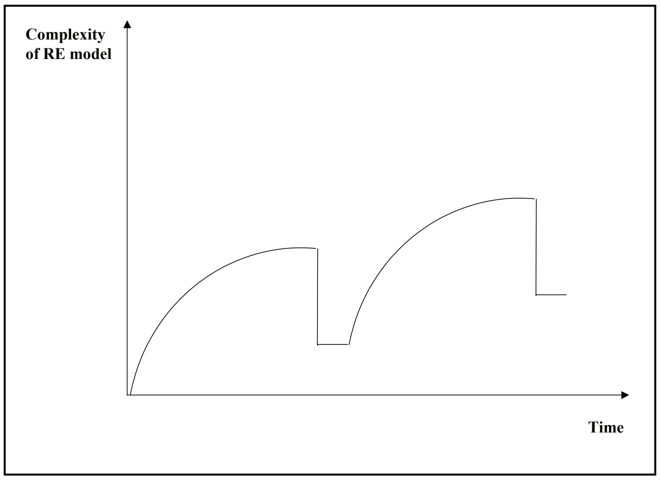


Figure 5. Catastrophe cycle for RE



of creative problem solving. He identified four stages of invention: (1) preparation, (2) incubation, (3) illumination (insight), and (4) the verification and expression of insight.

Schön (1987) notes that in the ordinary form of practical knowledge practitioners do not think about what they are doing, except when puzzled or surprised. Schön named this *reflecting-in-action* and argued that it is central to the ability to act effectively in the unique, ambiguous, or divergent situations that become central to professional practice. Conceptually, this means being able to think outside the existing boxes altogether in order to invent new ones (Table 7 provides one framework that applies these concepts), which are critical in "messy" disciplines.

This relates to a further issue that needs to be addressed: the need to engage in life-long learning. The speed with which technology evolves, the multiplicity of its impact on society, and the ramifications of that impact mean that meta-cognitive and knowledge construction skills as well as adaptability become vital. Professional practitioners with such skills become *agents of change* (Garlan et al., 1997).

Bowden and Marton (1998) explore several significant ways of engaging with the question of preparing others for situations that are highly variable and novel and that do not neatly match up with the (artificial) boundaries between discipline or knowledge areas. The CreativePBL environment students experienced has gone some way to addressing these:

- Shifting the focus from teaching to learning. The environment is student-centred and minimises "teaching."
- Concentrating on developing (generic) capabilities and on student learning outcomes. It may be considered a creative environment that enhances divergent thinking and the creative potential of students.
- Moving from highly differentiated and fragmented curricula to integrated learning programmes. The approach is somewhat holistic.

Therefore, while the progressive development of design studios and CreativePBL models have had some measure of success, the learning diagnostics (e.g., Approaches to Study, Entwistle & Ramsden, 1983) results indicated at least as strong a bias to surface learning as there is to deep learning. The literature suggests this is an outcome of the (different) learning environments students are exposed to in (different) units. Ultimately, innovation introduced a few units may be undermined if traditional approaches are maintained elsewhere in the students' programme—so that benefits may only be apparent or are enhanced if the innovation is introduced across the entire curriculum.

The next cycle therefore took a two-pronged approach:

- Applying a studio learning model across all units and all programmes within engineering, albeit for the final 2 years of study only. This approach addressed the issue of undermining the learning "philosophy" being effected in the SE programme.[4]
- Building into the curriculum an even greater focus on generic skills as part of the outcomes of the learning environment. Further refinement to the model was also required in order to achieve a greater degree of "fit" or constructive alignment between the components of the learning environment.

Design Week

In 2005 the school instituted a shift to studio learning in the final years of all undergraduate engineering programmes. Based on the model developed through the research described in this chapter, studio learning is a group-based learning approach that requires academics working as facilitators to provide guidance in a richer, holistic learning environment. The aims of this move were:

- Improved learning outcomes for students in areas such as project management; problems solving; group and co-operative work skills; and communication skills
- Increased focus on design content within each discipline area
- A closer match to professional requirements and the potential to integrate into employment positions on graduations

By applying studio learning throughout engineering the aim was to enable all graduates to meet the dramatic changes of a transforming industry.

In order to effect the "cultural change" towards learning required by this move, students coming into third year (and for 2005, 4th year students) were involved in an orientation programme (Armarego & Fowler, 2005). The objectives of this week-long activity included:

- Modelling studio learning
- Establishing the roles and responsibilities of students and academics within this model
- Providing an introduction to the necessary support services made available with the learning environment

These were achieved through a small-scale design task as a means of identifying and exposing the studio learning approach and an introduction to generic tools, techniques, methods, and processes that might otherwise have to be duplicated in each studio.

A key component of the orientation was reflection on the process and outcomes by way of a journal/diary indicating tasks, outcomes, and times spent. This incorporated student feedback on the value of the experience. As additional feedback, students were asked in Week 6 (fourth years) and a few weeks later (3rd year students) during the semester to comment on the *Design Week* in the light of subsequent experiences with studio learning. In addition, students were required to complete a set of learning styles diagnostics prior to commencement. These act as benchmarks and will be one of the bases for ongoing evaluation of the learning approach.

All students completed the programme successfully—success being measured in terms of both the product (task adequately designed) and the process (group process established, PBL process applied). Students demonstrated their engagement with this learning model: The quality of the final presentations and diversity of solutions emphasised their ability to be self motivated independent learners. Significantly, initial observation indicated that students who were in the department prior to the *Design Week* were better able to make the shift to studio learning—understandable since it is pre-empted in several units already running. However, articulation students and (international) students joining the school on exchange programmes initially found the learning model disorienting and confronting.

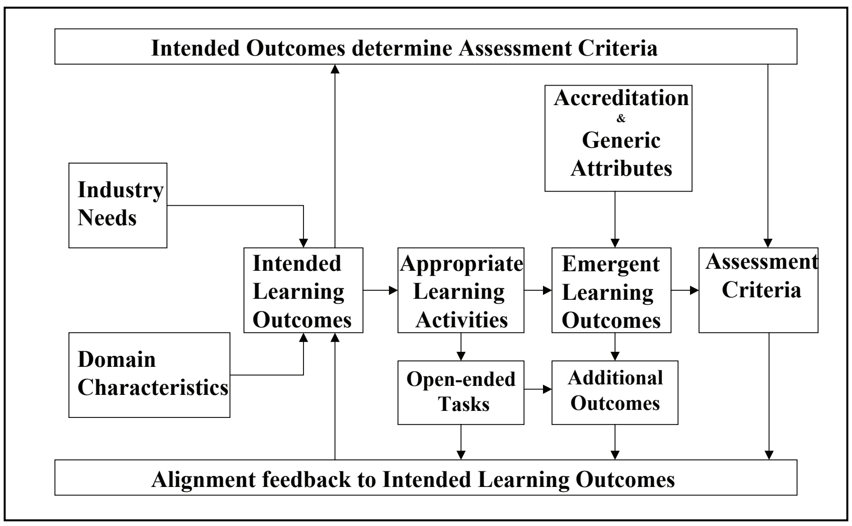
Curriculum Mapping

As well as other adaptations, applying the studio learning model required changes in class structure. All discipline-specific units were moved to the final 2 years. This meant G260 (or its equivalent design studio) moved into Sem 1, Year 3. Also, rather than time set aside for lectures, tutorials, and labs, studios worked in a block-teaching framework—each studio was allocated 10 hours of class contact (teacher present)—generally on a single day and 10 hours of additional class time. Therefore, students were expected to spend a minimum of 2 days a week on each studio plus any added time required by individual study habits.[5]

To tackle the issue of a surface learning focus in student (or, in reality, a lesser swing to deep learning than could be expected) a curriculum map was developed to examine the alignment between outcomes and assessment. English (1978) advocated the use of mapping to ensure that the declared aims of a curriculum match those which are taught and learned, while Biggs (1999) suggested mapping of assessment in order to achieve the alignment of declaration, delivery, learning, and assessment of individual skills. Based on a model developed by the Engineering Subject Centre of the Learning and Teaching Support Network (LTSN, 2002), all topics in the unit were categorised, firstly by the broad area of syllabus and then by the learning outcome to be addressed.

The map based on this model (see Figure 6) indicates that the learning objectives noted in the unit documentation are modified through student

Figure 6. Alignment between outcomes and assessment (adapted from LTSN, 2002)



engagement with the tasks and activities. The teacher identifies additional outcomes drawn from this engagement that address generic attributes. If alignment exists, the assessment is based on demonstration of the combined outcomes. The feedback loop ensures adaptation is facilitated for closer alignment in the next offering of the unit.

Students as Studio Learners

The student cohort undertook the unit successfully—although exams are not a totally appropriate summative assessment component in this environment, they do indicate "individual" performance as opposed to group achievement. A statistically significant increase in marks across all components of the exam was noted, with the exam modelling previous offerings intentionally.[6] The average exam marks are as indicated in Table 9.

However, more telling are (sample) comments made in response to the question regarding individual student perception of their learning in this environment (Figure 4 also refers to results for the same query from a previous year).

Table 9. Comparative exam marks

Year	Average exam mark
2001	48.08
2002	56.53
2003	55.53
2005	67.45

Simon	*In my opinion I learn more communication skills and in organizing and less in technical skills in this unit. So in my opinion I learned neither more nor less in this unit, but different things which I haven't learned before.*
Vaughn	*Seriously I feel I have learnt a lot more useful things in this unit compared to most of the other units I have taken at this University … I am learning more, much more for reasons that include:* • *I have been working in a very good team and feel that some of the knowledge I have learnt has resulted from the interaction with my team members i.e. I don't believe the level of understanding I now have, would have been achieved by working on the assignments by myself* • *The assignments being based around a problem gives a more realistic context as opposed to some abstract exercise to test understanding of theory* • *I have found that the assignments have been an extension of the previous one other and clearly a process that is being built upon at every stage i.e. each additional stage in the process has enlightened me to the relevance of the previous stage. This method of teaching has provided me with a framework that I can use to identify future problems and develop solutions.* *I have noticed that the design studios require a lot more work from me than if I was working alone. For example I have to spend more time working on problems because of the extra overhead of working in a team (meetings and social interaction). There is also the need to do extra research to gain information that is normally just handed out in a lecture. However I don't mind putting in the extra effort because I feel the extra effort is worth it because I feel more confident that I do know the material (not an impostor) and can apply it to future situations.*

Alaina	*I felt I have not learnt adequately because I could not manage my time effectively. However, the context of this unit was very interesting and the amount of workload was not heavy, I believe I could have learnt much more if I could organise the study more successfully*
Dermot	*I personally feel I learn less, I guess this is not my style of learning. It is as good as me taking a unit externally and just staying at home and teaching my self, and if I have problems asking a friend, or researching further. I guess however teaching your self things you do tend to understand concepts better. However I feel that I am an audio visual learner, thus listening to someone explaining the concepts, PowerPoint's and teaching it to us makes life easier for me. I believe I gain a better understanding in this way*

From the university-wide student survey (undertaken anonymously at the end of semester) the following comments are noteworthy:

Student A	*This unit teaches a process that is built on knowledge but more importantly that knowledge is converted to a skill via practice on the problem. I don't believe this is achieved by the other style of teaching e.g. lectures and exercise type assignments*
Student B	*These design studios are a formalisation of what is occurring naturally i.e. we learn from and work with each other already*
Student C	*This method of teaching has provided me with a frame work that I can use to identify future problems and develop solutions.*

Comments such as these samples may be aligned with the professional capability framework described earlier. Students are demonstrating awareness of their capabilities in several of the scales:

- Alaina is concerned about her organisational skills.
- Simon acknowledges his learning of generic skills.
- Vaughn and student B acknowledge the importance of social stance in learning.
- Student A and Vaughn comment on the educational quality of the environment.
- Student C demonstrates diagnostic mapping.
- Dermot indicates awareness of his personal stance and how it affects his learning.

Interestingly, Dermot's comments support the findings of Entwistle and Tait (1990, 1995). They found that students who reported themselves as adopting surface approaches to learning teaching and assessment procedures that supported that approach, whereas students reporting deep approaches preferred courses that were intellectually challenging and assessment procedures that allowed them to demonstrate their understanding.

Students also noted that with all their studies undertaken within a studio learning environment, they felt a reduction in the need to justify their efforts. Probing of this concept within a focus-group environment indicated the following:

- Students felt academic staff were more tolerant of the needs of other studios.
- With a full-time load of only two studios, student time was not as fragmented across different areas.
- Except for the (negotiated) compulsory attendance, students could vary the time they spent on each studio in response to their total learning context. It was the project manager's role to ensure tasks were on schedule.

They concluded that this flexibility reduced stress and allowed them to focus on the learning they needed to achieve.

Vaughn's comments in response to the question:

Do you feel there are any good things about a unit structured in this way?

act as a summary of student perception:

Vaughn	Yes I think this design studio is very very very (Did I mention very) well run. The problem (Development of the game environment by Mursoft for TerColl) covers all the learning outcomes. We have to apply learning's to a realistic problem which means it moves as out in the real world e.g. the lecturer (TerColl) pointed out errors in thinking and this resulted in us having to revise what we had completed previously in order to move to the next step. I found this gave me a greater depth of knowledge than the usual do an assignment get some of it wrong and move on to the next usually non-related assignment. The lecturer spending the agreed allocated time in the class room has been very useful i.e. we have been able to learn at a faster rate because we have been able to consult with the lecturer when we where unsure i.e. the lecturer became a mentor/consultant who suggested and guided rather than just giving being a lecturer/guru.

Students as Advanced Learners

This student cohort was further observed in a subsequent studio—specifically the unit where previous cohorts had "insisted on being taught." On this occasion, students exhibited a willingness to work independently and to vary their interactions (e.g., teacher explaining, students discussing together, students working individually, etc.) depending on the needs of the learning situation, calling on the teacher only as required.

Extensive data provide significant insight to the students' perceptions of themselves as learners. As one example, students have the choice, in this studio, of taking a final exam or presenting a working demonstration of the problem/system. Although the latter requires much more effort and is group-based, the cohort decided to dedicate the (extra) time required in producing a working system. The implication of this decision is their ability to gauge the level of proficiency of their attempts to master the problem and complete the task: They appear to be drawn actively into the problem and learning environment, suggesting "real learning" is occurring.

Student engagement with subsequent design studios has been exemplary: In effect, the teacher was consulted only as required. More interestingly, as students rotated into the role of project manager, they (individually) applied what they had previously learned with regards to learning strategies and approaches to study in order to motivate their group members.

Markus has the final word, in an e-mail 6 months after the end of the studio:

> Sent: Wednesday, 10 May 2006 12:08 AM
>
> Sorry I have not got back to you sooner, I have been waiting on definitive answers regarding internship possibilities[…] The particular project is a large one and most likely I will only get to the simulation phase. I will be redesigning a complete operating system. I am confident of doing the task with both my background […], and also using the methodology of Software Design you have taught me.
> I still stand by that the Software Design Studio you taught last year really has given much confidence in the process and the importance of Software Design.

He exhibits many of the attributes this research was attempting to target. He expresses confidence in his own ability to learn and apply new knowledge as well as adapt what he has learned. This confidence is based on knowledge and metacognitive skills that have been encouraged and developed throughout his formal education.

Reviews of Studio Learning

A department-wide review of the studio learning model was undertaken at the end of the first-semester offerings. Feedback was sought from all academic staff involved and student representatives across all design studios. The conclusion reached was that while refinement was needed, (for many staff this was the first implementation of nontraditional learning environments), and a need for student feedback aligned to the learning environment identified, design studios had been successful. Students commented positively on the provision of an introduction and rationale to PBL and design studios (46% of comments received, $n = 33$) and to the value of working in interdisciplinary groups (30%). Staff noted that, despite concerns regarding "lost" content, learning objectives were achieved.

External review of the proposal (as opposed to post-implementation) for design studios by the professional accreditation panel suggested this could become the leading programme in this country, while a school review undertaken by the university, at the end of one academic year of design studios, acknowledged staff and student satisfaction with studio learning and recommended that the model be applied throughout the school (i.e., not just to engineering). Longitudinal monitoring with data collection will go some way towards confirming these initial findings and perceptions.

DISCUSSION OF RESEARCH RESULTS/EXPERIENCE

Although in its infancy within this university and in the discipline of IT, studio learning has been seen to address issues raised in studies of discipline practitioners and the education literature. The need to:

- Provide students with authentic experiences that address competencies additional to specific discipline knowledge
    - Students were exposed to learning both as a "generic" metacognitive activity and as a skill to be continually adapted and utilised within a discipline context
    - Flexibility in thinking—addressing creativity, opportunism, and divergency/convergency—was made explicit and strategies to exploit it developed
- Provide learners with a deep understanding of self and others in complex human activity systems:
    - In a collaborative environment, students became aware of and learned to utilise each others' strengths and weaknesses in achieving the unit outcomes. They learned how to "jell," what to do if they did not, and to be empathetic to others' contexts.
    - They learned to value and exploit alternate perspectives brought to a problem by different stakeholders (client, teacher/consultant, other team members) to enrich their learning context.
    - They became aware of the need to be self-motivated and learn independently.
    - Students were confident in questioning their own and others' assumptions within the learning environment.
- Allow time to explore new ideas and to reflect on possible processes and outcomes:
    - Students were open to discussion and feedback and willing to retrace their steps/redo the work in order to advance to a solution.
    - They were willing to "trust" each other's knowledge (implicit or not, technical or not), accepting the multidisciplinary nature of the skills and

Table 9. Questions and answers addressed

Question 1	*How useful is the knowledge generally included in tertiary institution curricula for the practicalities of being an IT professional?*
Answer	Practitioner studies indicate a mismatch: profession-specific knowledge is generally addressed adequately within model curricula. However, practitioners emphasise affective qualities and generic intellectual abilities and skills. "Lip service" is more likely to be paid to these within formal education.
Question 2	*Do alternate learning models address any mismatch identified?*
Answer	Learning models based on construction of knowledge within a collaborative active learning environment appear to address issues raised by practitioners, especially if problems tackled are complex, ill-structured, and authentic.
Question 3	*Can these be applied successfully within a formal (tertiary) education environment?*
Answer	This research shows it can, at least in the context in which it was applied. In addition, the model developed—studio learning—has been successfully applied to all disciplines of engineering at 3rd and 4th Year within this university. Long-term success, however, is based on employer reaction and graduate career prospects. These require further research.

knowledge required to achieve the learning objectives

- Be challenged
  - Students were motivated by the (increasing) complexity of the task and were able to focus on cognitive and interpersonal skills to adapt to the changes required.

Within the context of IT learning within Murdoch Engineering, this research goes some way to answering the questions posed at the commencement of this chapter (see Table 9).

As noted previously, within the SE programme, additional research has been undertaken to evaluate student ability to transfer the skills and competencies gained to subsequent units and to a workplace-learning environment (in the context of an internship). While that work is discussed only briefly here, preliminary results, and in particular, employer reaction within the IT discipline are encouraging, to say the very least.

As an example of employer reaction, a global software development organisation with a workforce of over 60,000 accepted a lone SE intern in 2003. In 2004 this was doubled to two students—in 2005 the request was for 10 students who had participated in studio learning in SE. More revealing, this demand was not matched at other universities in the state offering engineering programs for software.

A further indication of employer satisfaction is provided by graduate career prospects. While empirical evidence is in the process of being accumulated, (there are still too few SE graduates to provide statistically significant results), the anectodal evidence is also encouraging. Where one (20%) 2004 graduate SE was employed by the same global software development organisation noted in the previous paragraph, of the 2005 cohort 50% (six graduates) are now employed there. Both 2006 graduates (100%) are also with the same organisation.

CONCLUSION

Industry requires professionals who integrate into the organisational structure, and rather than cope specifically with today's perceived problems, have models, skills, and analytical techniques that allow them to evaluate and apply appropriate emerging technologies. More broadly, software technology is seen as a rapidly shifting landscape: new methods, tools, platforms, user expectations, and software markets underscore the need for education that provides professionals with the ability to *adapt* quickly (Garlan et al., 1997).

As we learn more about *how* students learn and *what* they need to learn in order to practice as competent professionals in their chosen discipline, we move further from traditional teaching and closer to the concept of learning as a reflection on professional practice undertaken by both teachers and learners.

This view of professional education has implications for the design of teaching (Laurillard, 1993):

- Academic *learning* must be situated in the domain of the objective—the activities must match that domain.
- Academic *teaching* must address both the direct experience of the world and the reflection on that experience that will produce the intended way of representing it.

The progression to studio learning has been a journey undertaken by academics of this university. In empowering graduates to be industry-ready IT professionals, staff benefit from a double-loop approach as the espoused theory of teaching becomes aligned with the theory in practice. It provides learning situations to examine and experiment with our theories of action (Argyris & Schön, 1974). For the student, the collaborative nature of the learning environment that has evolved transcends the classroom, fostering self-directed learning and reflective practice that integrates class and work experience.

Although results are positive and promising, their future will test the long-term wisdom of our approach.

REFERENCES

Albert, R. S. (1996). Some reasons why childhood creativity often fails to make it past puberty into the real world. In M. A. Runco (Ed.), *Creativity from childhood through adulthood: New directions for child development* (pp. 43-56). San Fransisco: Jossey-Bass.

Argyris, C., & Schön, D. A. (1974). *Theory in practice: Increasing professional effectiveness.* San Francisco: Jossey Bass.

Armarego, J. (2002). *Advanced software design: A case in problem-based learning.* Paper presented at the CSEET2002 15th Conference on Software Engineering Education and Training, Covington, KY.

Armarego, J. (2005). *Educating agents of change.* Paper presented at the CSEE&T2005 18th Conference on Software Engineering Education and Training, Ottawa, Ontario, Canada.

Armarego, J., & Fowler, L. (2005). *Orienting students to studio learning.* Paper presented at the Proceedings of the 2005 ASEE/AaeE 4th Global Colloquium on Engineering Education, Sydney, Australia.

Avison, D. E., Lau, F., Myers, M., & Nielsen, P. A. (1999). Action research. *Communications of the ACM, 42*(1), 94-97.

Bach, J. (1997). SE education: We're on our own. *IEEE Software, 14*(6), 26, 28.

Bach, J. (1999). Reframing requirements analysis. *IEEE Computer, 32*(2), 120-122.

Baer, J. M. (1988). Long term effects of creativity training with middle-school students. *Journal of Adolescence, 8,* 183-193.

Bentley, J. F., Lowry, G. R., & Sandy, G. A. (1999). *Towards the complete information systems graduate: A problem based learning approach.* Paper presented at the Proceedings of the 10th Australasian Conference on Information Systems.

Biggs, J. (1999). *Teaching for quality learning at university: What the student does*. Buckingham, UK: Open University Press.

Bloom, B. S. (1956). *Taxonomy of educational objectives: The classification of educational goals handbook 1: Cognitive domain*. New York: David Mackay.

Borg, W., Gall, J. P., & Gall, M. D. (1993). *Applying educational research: A practical guide* (3rd ed.). White Plains, NY: Longman.

Boud, D. (1985). Problem-based learning in perspective. In D. Boud (Ed.), *Problem-based learning in education for the professions* (pp. 13-18). Sydney: Higher Education Research Society of Australasia.

Bowden, J., & Marton, F. (1998). *The university of learning: Beyond quality and competence in higher education*. London: Kogan Page.

Brooks, F. P. (1986). *No silver bullet—Essence and accidents of software engineering.* Paper presented at the Proceedings of Information Processing 86: the IFIP 10th World Conference, Amsterdam.

Bubenko, J. (1995). *Challenges in requirements engineering: Keynote address.* Paper presented at the RE'95: Second IEEE International Symposium on Requirements Engineering, York, UK.

Budgen, D. (1995). *Is teaching software design a "wicked" problem too?* Paper presented at the Software Engineering Education: Proceedings of the 8th SEI Conference on Software Engineering Education, New Orleans, LA.

Budgen, D. (2003). *Software design*. Harlow, Essex, UK: Pearson Education Ltd.

Bunge, M. (1984). Philosophical inputs and outputs of technology. In G. Bugliarello & D. Donner (Eds.), *History and philosophy of technology* (pp. 263-281). Urbana: University of Illinois Press.

Carr, W., & Kemmis, S. (1986). *Becoming critical: Education, knowledge and action research*. Lewes, UK: Falmer.

Carroll, J. M., & Swatman, P. A. (1999). *Opportunism in the requirements engineering process* Working Paper No. 1999/02. Australia: Deakin University, School of Management Information Systems.

Cropley, D. H., & Cropley, A. J. (1998). *Teaching engineering students to be creative—Program and outcomes.* Paper presented at the Australasian Association of Engineering Education: 10th Annual Conference.

Dahlgren, M. A. (2000). Portraits of PBL: Course objectives and students' study strategies in computer engineering. *Instructional Science, 28*(4), 309-329.

Denzin, N. (1970). Strategies of multiple triangulation. In N. Denzin (Ed.), *The research act in sociology: A theoretical introduction to sociological method* (pp. 297-313). New York: McGraw-Hill.

Edmonds, E., & Candy, L. (2002). Creativity, art practice and knowledge. *Communications of the ACM, 45*(10), 91-95.

English, F. (1978). *Quality control in curriculum development*. Arlington, VA: American Association of School Administrators.

Entwistle, N. J., & Ramsden, P. (1983). *Understanding student learning.* London: Croom Helm.

Garlan, D., Gluch, D. P., & Tomayko, J. E. (1997). Agents of change: Educating future leaders in software engineering. *IEEE Computer, 30*(11), 59-65.

Glass, R. L. (1995). *Software creativity.* Englewood Cliffs, NJ: Prentice Hall.

Gott, S. P., Hall, E. P., Pokorny, R. A., Dibble, E., & Glaser, R. (1993). A naturalistic study of transfer:

Adaptive expertise in technical domains. In D. K. Detterman & R. J. Sternberg (Eds.), *Transfer on trial: Intelligence, cognition and instruction* (pp. 258-288). Norwood, NJ: Ablex.

Grundy, S. (1982). Three modes of action research. *Curriculum perspectives, 2*(3), 23-34.

Guba, E., & Lincoln, Y. (1994). Competing paradigms in qualitative research. In N. Denzin & Y. Lincoln (Eds.), *The handbook of qualitative research* (pp. 105-117). Thousand Oaks, CA: Sage.

Guindon, R. (1989). The process of knowledge discovery in system design. In G. Salvendy & M. J. Smith (Eds.), *Designing and using human-computer interfaces and knowledge based systems* (pp. 727-734). Amsterdam: Elsevier.

Guindon, R. (1990). Knowledge exploited by experts during software systems design. *International Journal of Man-Machine Studies, 33,* 279-304.

Hatten, R. A. (1997). *Guidelines for effectiveness in continuing professional education.* Unpublished master's thesis, University of Sydney, Australia.

Kember, D., & Kelly, M. (1993). *Improving teaching through action research.* Campbelltown, New South Wales, Australia: Higher Education Research and Development Society of Australasia Inc.

Khushalani, A., & Smith, R. (1994). What happens when designers don't play by the rules: Towards a model of opportunistic behaviour in design. *Australian Journal of Information Systems, 1*(2), 13-31.

Koschmann, T. D., Myers, A. C., Barrows, H. S., & Feltovich, P. J. (1994). Using technology to assist in realising effective learning and instruction: A principled approach to the use of computers in collaborative learning. *The Journal of the Learning Sciences, 3*(3), 227-264.

Laurillard, D. (1993). *Rethinking university teaching: A framework for the effective use of educational technology*. London: Routledge.

Lee, D. M. S. (1986). Academic achievement, task characteristics, and first job performance of young engineers. *IEEE Transactions on Engineering Management, EM-33*(3), 127-133.

Lee, D. M. S. (1992). Job challenge, work effort, and job performance of young engineers: A causal analysis. *IEEE Transactions on Engineering Management, 39*(3), 214-226.

Lee, D. M. S. (1999). *Information seeking and knowledge acquisition behaviors of young information systems workers: Preliminary analysis.* Paper presented at the 1999 Americas Conference on Information Systems.

Lee, D. M. S. (2004). Organizational entry and transition from academic study: Examining a critical step in the professional development of young IS workers. In M. Igbaria & C. Shayo (Eds.), *Strategies for managing IS/IT personnel* (pp. 113-141). Hershey, PA: Idea Group.

Lethbridge, T. C. (1999). *The relevance of education to software practitioners: Data from the 1998 survey* (Tech. Rep. No. TR-99-05). Canada: University of Ottawa, School of Information Technology and Engineering.

Lethbridge, T. C. (2000). What knowledge is important to a software professional? *IEEE Computer, 33*(5), 44-50.

Lewin, K. (1946). Action research and minority problems. *Journal of Social Issues, 2,* 34-46.

Lowry, G., & Turner, R. (2005). Information systems education for the 21st century: Aligning curriculum content & delivery with the professional workplace. In D. Carbonara (Ed.), *Technology literacy applications in learning environments* (pp. 171-202). Hershey, PA: IRM Press.

LTSN. (2002). *Constructive alignment and why it is important to the learner.* Retrieved April 30, 2007, from http://www.engsc.ac

Lubars, M., Potts, C., & Richer, C. (1993). *A review of the state of the practice in requirements modeling.* Paper presented at the International Symposium on Requirements Engineering, San Diego, CA.

Macauley, L., & Mylopoulos, J. (1995). Requirements engineering: An educational dilemma. *Automated Software Engineering, 2*(4), 343-351.

Maiden, N., & Gizikis, A. (2001). Where do requirements come from? *IEEE Software, 18*(5), 10-12.

Maiden, N. A. M., & Sutcliffe, A. G. (1992). Exploiting reusable specifications through analogy. *Communications of the ACM, 34*(5), 55-64.

Mann, J. (1996). *The role of project escalation in explaining runaway information systems development projects: A field study.* Atlanta: Georgia State University.

Minor, O. (2004). *Theory and practice in requirements engineering: An investigation of curricula and industry needs.* Unpublished masters thesis, University of Koblenz-Landau, Koblenz, Germany.

Nguyen, L., & Swatman, P. A. (2000). *Essential and incidental complexity in requirements models.* Paper presented at the Fourth International Conference on Requirements Engineering Education, Schaumburg, IL.

Oliver, R., & McLoughlin, C. (1999). *Using Web and problem-based learning environments to support the development of key skills.* Paper presented at the Responding to Diversity: Proceedings of ASCILITE '99, Brisbane, Australia.

Orona, C. (1990). Temporality and identity loss due to Alzheimer's disease. *Social Science and Medicine, 30*(11), 1247-1256.

Patel, A., Kinshuk, & Russell, D. (2000). Intelligent tutoring tools for cognitive skill acquisition in life long learning. *Educational Technology & Society, 3*(1), 32-40.

Pfleeger, S. L. (1999). Albert Einstein and empirical software engineering. *IEEE Computer, 32*(10), 32-37.

Reeves, T. C. (1997). *A model of the effective dimensions of interactive learning on the World Wide Web.* Paper presented at the Proceedings of Interaktiivinen Teknologia Koulutuksessa (ITK '97), Hameenlinna, Finland.

Richardson, J. T. E. (1990). Reliability and replicability of the approaches to studying questionnaire. *Studies in Higher Education, 15,* 155-168.

Robillard, P. N. (1999). The role of knowledge in software development. *Communications of the ACM, 42*(1), 87-92.

Robillard, P. N. (2005). Opportunistic problem solving in software engineering. *IEEE Software, 22*(6), 60-67.

Royce, W. W. (1970). *Managing the development of large software systems: Concepts and techniques.* Paper presented at the IEEE WESCON.

Savin-Baden, M. (2000). *Problem-based learning in higher education: Untold stories*. Buckingham, UK: Society for Research into Higher Education and Open University Press.

Schön, D. A. (1987). *Educating the reflective practitioner: Towards a new design for teaching in the professions*. San Fransisco: Jossey-Bass.

Scott, G., & Wilson, D. (2002). *Tracking and profiling successful IT graduates: An exploratory study.* Paper presented at the Proceedings of the 13th Australasian Conference on Information Systems.

Scott, G., & Yates, W. (2002). Using successful graduates to improve the quality of undergraduate engineering programs. *European Journal of Engineering Education, 27*(4), 60-67.

Snoke, R., & Underwood, A. (1999). *Generic attributes of IS graduates—A Queensland study.* Paper presented at the Proceedings of the 10th Australasian Conference on Information Systems, Wellington, New Zealand.

Spiro, R. J., Feltovich, P. J., Jacobson, M., & Coulson, R. (1991). Cognitive flexibility, constructivism and hypertext: Random access instruction for advanced knowledge acquisition in ill-structured domains. *Educational Technology, 31,* 24-33.

Standish. (1995). *Most programming projects are late*. West Yarmouth, MA: Author.

Sutcliffe, A. G., & Maiden, N. A. M. (1992). Analysing the novice analyst: Cognitive models in software engineering. *International Journal of Man-Machine Studies, 36,* 719-740.

Thomas, J. C., Lee, A., & Danis, C. (2002). Enhancing creative design via software tools. *Communications of the ACM, 45*(10), 112-115.

Trauth, E. M., Farwell, D., & Lee, D. M. S. (1993). The IS expectation gap: Industry expectation versus academic preparation. *MIS Quarterly, 17,* 293-307.

Turner, R., & Lowry, G. (1999). *Educating information systems professionals: Towards a rapprochement between new graduates and employers.* Paper presented at the Proceedings of the 10th Australasian Conference on Information Systems, Wellington, New Zealand.

Turner, R., & Lowry, G. (2003). Education for a technology-based profession: Softening the information systems curriculum. In T. McGill (Ed.), *Current issues in IT education* (pp. 153-172). Hershey, PA: IRM Press.

Underwood, A. (1997). *The ACS core body of knowledge for information technology professionals*. Sydney: Australian Computer Society.

Visser, W. (1992). Designers' activities examined at three levels: Organisation strategies and problem-solving processes. *Knowledge-Based Systems, 5*(1), 92-104.

Visser, W., & Hoc, J. (1990). Expert software design strategies. In J. M. Hoc, T. R. G. Green, S. R & G. D. J (Eds.), *Psychology of programming* (pp. 235-247). San Diego, CA: Academic Press.

Waks, L. J. (2001). Donald Schon's philosophy of design and design education. *International Journal of Technology and Design Education, 11,* 37-51.

Wallas, G. (1926). *The art of thought*. London: Jonathan Cape.

Wilson, B. G., & Cole, P. (1996). Cognitive teaching models. In D. H. Jonassen (Ed.), *Handbook of research for educational communications and technology* (pp. 601-621). New York: Simon & Schuster Macmillan.

Winn, W., & Snyder, D. (1996). Cognitive perspectives in psychology. In D. H. Jonassen (Ed.), *Handbook of research for educational communications and technology* (pp. 112-142). New York: Simon & Schuster Macmillan.

Zuber-Skerritt, O. (1995). Models for action research. In S. Pinchen & R. Passfield (Eds.), *Moving on: Creative applications of action learning and action research* (pp. 3-29). Mountt Gravatt, Queensland, Australia: Action Learning, Action Research and Process Management Assn. Inc.

Zucconi, L. (1995). *Essential knowledge for the practicing software engineer and the responsibilities of university and industry in her education.* Paper presented at the Software Engineering Education: 8th SEI Conference on Software Engineering Education, New Orleans, LA.

ENDNOTES

1 Within the environment of this university a unit is equivalent to a course. Within a defined programme of study (e.g., BE(SE)), a prescribed set of units must be completed successfully.

2 These codes are inserted purely to assist in identifying the units being discussed.

3 Quantitative—based on assessment components and the reduced *Approaches to Study Inventory* confirmed by Richardson's (1990) work to possess adequate internal consistency and test-retest reliability; qualitative—based on surveys, interviews, and personal journals.

4 Since SE students must also complete generic engineering units.

5 A full-time load was defined as enrolment in two design studios, requiring 40 hours.

6 This has been the case throughout the study. While questions differ, the outcomes being assessed and the form of the question did not. In addition, students always had access to previous exams.