Teaching Software Engineering Management – Issues and Perspectives

C. Caulfield, D. Veal, S. P. Maj

Edith Cowan University, Perth, Western Australia

Summary
The ACM/IEEE regularly proposes guidelines for software engineering education, in particular what should be part of the software engineering core body of knowledge and how this knowledge can be taught. The 2004 curriculum guidelines define seven student outcomes, two of which relate to teamwork and project control, and one Software Engineering Education Knowledge (SEEK) area on software management. The software management knowledge area is concerned with the entire software development life cycle and hence the control of people and processes. Significantly, the majority of topics within this area are classified with the Bloom taxonomy level of Application i.e. ability to use learned material in new and concrete situations. However the laboratory and assignment exemplars fail to demonstrate the dynamic, human centered complexity of project management. Simsoft, a serious game, has been designed to potentially address this pedagogical gap.

Key words:
software engineering curriculum, serious games, project management, problem-based learning

1. Software Engineering Curriculum 2004

The Joint Task Force on Computing Curricula (IEEE Computer Society and Association of Computing Machinery) suggests curriculum guidelines for undergraduate degree programs in software engineering. The SE2004 [1] volume defines a core body of knowledge called Software Engineering Education Knowledge (SEEK) which was the basis of curriculum recommendations. SE2004 also defined seven student outcomes that include:

- Work as an individual and as part of a team to develop and deliver quality software artifacts.
- Reconcile conflicting project objectives, finding acceptable compromises within limitations of cost, time, knowledge, existing systems and organizations.

There are ten SEEK knowledge areas— sub-disciplines of the field that undergraduates should know— which are broken down into smaller knowledge units— thematic modules— and finally into topics. Within the Software Management knowledge area, there are five knowledge units (Table 1).

Table 1: Software management knowledge units

KA/KU	Software Management	Hours Required
MGT.con	Management concepts	2
MGT.pp	Project planning	6
MGT.per	Project personnel and organization	2
MGT.ctl	Project control	4
MGT.cm	Software configuration management	5

Drilling down further, the Project Planning knowledge unit consists of six topics (Table 2), three of which are classified as the Bloom [2] taxonomy level of Application.

Table 2: Project planning topics

Project Planning	Bloom's Taxonomy
Evaluation and planning	Comprehension
Work breakdown structure	Application
Task scheduling	Application
Effort estimation	Application
Resource allocation	Comprehension
Risk management	Application

The Bloom taxonomy is a classification of learning objectives (learning outcomes) consisting of three domains: cognitive, affective and psychomotor. The cognitive domain defines six levels of taxonomy from the lowest to the highest:

1. Knowledge: remember previously-learned materials by recalling specific facts, terminology, theories and answers
2. Comprehension: demonstrate an understanding of information by being able to compare, contrast, organize, interpret, describe, and extrapolate.
3. Application: use previously-learned material in new situations.
4. Analysis: decompose previously-learned material into parts in order find patterns and to make inferences and generalizations.
5. Synthesis: use existing ideas in different ways to create new ideas or to propose alternative solutions.

Manuscript received July 5, 2011
Manuscript revised July 20, 2011

6. Evaluation: judge the validity of ideas or information with a certain context.

Meanwhile, the Project Personnel and Organization knowledge unit consists of seven topics, three of which are classified as the Bloom taxonomy level of Application (Table 3).

Table 3: Project personnel and organization topics

Project Personnel and Organization	**Bloom's Taxonomy**
Organizational structures, positions, responsibilities and authority	Knowledge
Formal/informal communication	Knowledge
Project staffing	Knowledge
Personnel training, career development, and evaluation	Knowledge
Meeting management	Application
Building and motivating teams	Application
Conflict resolution	Application

2. SE2004 Courses

The SE2004 curriculum guidelines define topic implementation as a series of courses. Within the context of software engineering management there are three associated courses:

- SE322 Software Requirements Analysis
- SE323 Software Project Management
- SE324 Software Process and Management

The Software Requirements Analysis course is primarily concerned with requirements analysis and modeling. The sample laboratories and assignments require students to use different analysis and modeling tools.

The Software Project Management course is designed to teach project planning. The laboratories and assignments include:

- Use a commercial project management tool to assist with all aspects of software project management
- Make cost estimates for a small system using a variety of techniques
- Developing a project plan for a significant system
- Writing a configuration management plan
- Using change control and configuration management tools
- Evaluating a software contract or license

Furthermore, this unit recommends case studies of real industrial projects.

The Software Process and Management course teaches standards, implementation and assurance of software processes. No sample laboratories and assignments are provided.

SE2004 curriculum guideline encourages a variety of teaching and learning approaches that include: problem-based learning; just-in-time learning; learning by failure and self-study materials (see for example [3-5]). However in a commercial environment software project management is a human-centered activity that attempts to address the dynamic interactions of factors such as cost, time, staffing, performance, feature set, and quality. A relatively small change in one factor, such as the resignation of a single software engineer, is likely to have a significant impact on the entire project. Whilst learning-to-fail is instructive [6], in a commercial context there are obvious economic implications.

In order to address these concerns, SE2004 includes a capstone project. The course SE400 Software Engineering Capstone Project recommends the development of a significant software system along with all the appropriate artifacts such as project plan, requirements, design documents, test plans etc. Additional teaching considerations include:

- It is suggested that students be required to have a 'customer' for whom they are developing their software
- It is strongly suggested that students work in groups of at least two, and preferably three or four, on their capstone project. Strategies must be developed to handle situations where the contribution of team members is unequal.

3. Meeting a Pedagogical Gap

The authors submit that there is a pedagogical gap between teaching software project management by means of the listed laboratories and assignments and the final capstone project. What is needed is for students to experience and experiment with a dynamic, interactive system that can be deployed by means of, for example, a game. As used here, to play a game:

…is to engage in activity directed towards bringing about a specific state of affairs, using only means permitted by specific rules, where the means permitted by the rules are more limited in scope than they would be in the absence of the rules and where the sole reason for accepting such limitation is to make possible such activity.[7]

The type of game that this paper is concerned with uses an adjective— serious— to show they want for more than simple amusement and that they are designed to educate, train, or inform their players [8-10].

A suitably designed game can not only can mimic real world complexity but can also provide immediate feedback regarding system performance and the effect of decisions made [11].

For example, in an idealized learning process (Figure 1), we receive information in its many forms from the real world in which we live, yet this information can incomplete, biased, delayed, or in other ways distorted. Still, based on this information, we make decisions that are in turn filtered through our existing mental models, in the process changing or confirming the structure of our real-world systems and creating new decision rules and new strategies or reinforcing the existing. The process then repeats against this new baseline. Games act as an alternative to applying our decisions to the real-world, a way of quickly, inexpensively, and consistently experimenting with different ideas and thereby increasing our store of contexts.

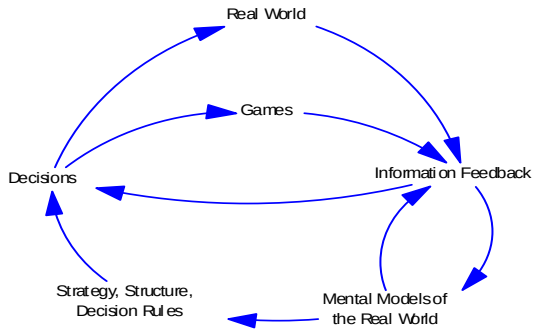

Fig.1: Idealised learning process.

For example, at its simplest, a game, such as the Beer Game, a four-point distribution game developed originally at MIT, can be used to show the cascading effects of a single compensating decision [12-14]. When using the beer game to teach planning, Caulfield [15] found that:

The participants reported a sense of having little control over their ordering decisions and tended to see the root cause of their inventory problems as being caused by other points in the supply chain.

Results such as this can potentially improve learning outcomes because the players can *see* the results of their actions and have to react accordingly.

To achieve this effect, games do not need high fidelity and need not be overly complex In fact, it has been demonstrated that whilst:

…the most complex game offered the richest leaning experience available, the game's very formidable appearance probably intimidated a number of players or faced them into a learning situation they were unprepared or unwilling to negotiate [16]

That is, rich and complex games can be daunting for players and they may not be willing to devote the time and effort to play it in depth. The next most effective game in Wolfe's study was found to be the least complex, supporting similar research that showed relatively simple games can provide essentially the same benefits as the more complex [17-19]. Game design is therefore of paramount importance.

4. Simsoft

This paper reports the initial findings of a research project that developed a game called Simsoft to teach software project management. A series of game sessions were conducted with teams of post-graduate project management students (for software and general projects), and practising software project managers and developers (n=59) between May and September 2010. The data sources for the findings were the participants' performance in Simsoft, pre- and post-game surveys, interviews with the participants, and a qualitative rich analysis of the interactions that were observed during the game sessions.

Physically, Simsoft comes in two pieces. There is an A0-sized printed game board around which the players gather to discuss the current state of a project and to consider their next move. The board shows the flow of the game while plastic counters are used to represent the staff of the project. Poker chips represent the team's budget, with which they can purchase more staff, and from which certain game events may draw or reimburse amounts depending on decisions made during the course of the game.

There is also a simple Java-based dashboard, through which the players can see the current and historical state of the project through a series of simple reports, messages, and other information; and can adjust the project's settings, for example to recruit new staff, before advancing the game's time to create the state of the project.

The aim of the game was to complete the project on time and with funds (poker chips) left over.

4.1 SimpleVersus Complex Games

The players' responses to different features of the game were generally positive (Table 4). Notable in Table 4 is that a majority of players (44 out of 59) preferred playing with a game board rather than a fully computerized version. Some typical comments were:

"The board game [was] simple and I could easily see the state of the game"

"When a group plays the game on a PC, someone controls the mouse and keyboard and they tend to dominate"
"Compared to computer-based games, the design was simple and we started playing without too much wasted time"
"Sometimes technology gets in the way"
"Everyone plays board games so we all knew what to do"

Table 4: Evaluation of game features

Feature	**Average (1 = very bad, 5 = very good; or 1 = strongly disagree, 5 = strongly agree)**
Written instructions	Average = 4.44, SD = 0.771
The game was interesting	Average = 4.37, SD = 0.963
Realistic scenario	Average = 4.37, SD = 0.692
Game logic was apparent	Average = 4.18, SD = 0.730
Useful to work in teams	Average = 4.15, SD = 0.714
Prefer game-board version	Average = 3.98, SD = 0.754

Outside of this research project, seven players had played The Beer Game mentioned before. In The Beer Game all calculations are performed by hand on simple worksheets. This found favour:

"Doing the calculations by hand means we have to understand"
"The calculator half of the game hides details. Just give us a calculator and we can work it out"

Although the players' reception of the game was generally positive, clear written instructions are essential to make sure best use is made of the game session time. This comment was made by a player in the very first game session:

"Wasn't sure of what we were supposed to do"

Initially, instructions for playing the game were delivered by the researcher after the players had completed the pre-game survey and just before they started the game. For the second game session onwards, a one-page instruction sheet was emailed to each player a couple of days beforehand so they could be prepared.

The database of Simsoft game transactions showed that only three games had to be abandoned and restarted. It was observed that once teams had made the first couple of decisions, they were able to continue with too much trouble.

4.2 Working in Groups

An important component of many of the pedagogical theories behind Simsoft is the aspect of working in groups or teams, so it was important to assess how this was received by the players. A majority of players (44 out of 59) said they found it useful or very useful to work as a team and that this reflected how things often happened in the workplace:

"It was like [the agile] stand up meeting we have every morning"
"We organised our selves into roles we felt comfortable with or that fitted our day-job: someone on the calculator, someone moving the developer pieces, someone moving the units of work"

However, one student found something new in the practice:

"I thought software development was a solitary experience but it's not really"

Others liked the opportunity to share opinions and learn from more experienced peers:

"Everyone had a chance to offer an opinion"
"I have little real-world project experience so it was good to get the advice of others and see how they approached problems"

But, as in any group activity, the game facilitator needs to be aware of cultural differences that may make some less inclined to contribute and of players who are dominating their groups:

"Generally, everyone had their say in final decision but a couple of times we were overridden"

4.3 Summary

These are the initial findings discovered through a series of Simsoft game sessions conducted with teams of post-graduate project management students, and practising software project managers and developers.

The first initial finding was that the majority of the participants found working in groups was a positive experience. The participants were a diverse group of cultures, skills, and experience and many felt they were still able to work out collaborative decisions in a constructive manner. However, as with any group activity, facilitators need to be cognizant of any individuals dominating a group or others who might need a gentle prompt to contribute more.

The second initial finding was a majority of participants preferred to play around a game board rather than a fully computerized game because this was a familiar and simple activity and less time was lost to overcoming technological problems and to making simple ergonomic arrangements such as fitting all the team around a single computer. Even so, facilitators need to prepare the participants for the game

sessions by giving them clear instructions and sufficient lead time to absorb the information.

These findings were reviewed by four participants chosen at random and all concurred without comment.

5. Conclusions

Preparing students for employment is of paramount importance for universities. Not only can this help improve employment prospects but it can also better meet employer expectations. A capstone project is designed to assist with this transition to employment. However, prior to undertaking a capstone project there are potentially significant pedagogical benefits to teaching project management using a game such as Simsoft. Importantly, the interim results presented in this paper demonstrate that even simple games can help students experience the team work, negotiation, and consensus-building skills they will need in the workforce.

References

[1] Joint Task Force on Computing Curriculum, Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering, IEEE Computer Society/Association for Computing Machinery, 2004.
[2] B.S. Bloom, B.B. Masia, D.R. Krathwohl, Taxonomy of Educational Objectives: The Classification of Educational Goals, Handbook I: Cognitive Domain ed., Longman, London, 1956.
[3] J.P. Gee, Situated Language and Learning: A Critique of Traditional Schooling, Routledge, London, 2004.
[4] M. Savin-Baden, C.H. Major, Foundations of Problem-Based Learning, The Society for Research into Higher Learning & Open University Press, Maidenhead, 2004.
[5] C. Aldrich, Learning by Doing: A Comprehensive Guide to Simulations, Computer Games, and Pedagogy in e-Learning and Other Educational Experiences Pfeiffer, San Francisco, 2005.
[6] R.F. Baumeister, C. Finkenauer, Review of General Psychology, 5 (2001) 323 – 370.
[7] B. Suits, Ethics, 77 (1967) 209 – 213.
[8] C.C. Abt, Serious Games, The Viking Press, New York, 1970.
[9] M. Schrage, T. Peters, Serious Play : How the World's Best Companies Simulate to Innovate, Harvard Business School Press, 1999.
[10] D. Michael, S. Chen, Serious Games: Games That Educate, Train, and Inform, Thomson Course Technology PTR, Boston, 2005.
[11] J.D. Sterman, Business Dynamics: Systems Thinking and Modelling for a Complex World, Irwin McGraw-Hill, New York, 2000.
[12] J.S. Goodwin, S.G. Franklin, Journal of Management Development, 13 (1994) 7 – 15.
[13] E. Mosekilde, E.R. Larsen, System Dynamics Review, 4 (1988) 131 - 147.
[14] J.D. Sterman, Management Science, 35 (1989) 321 – 339.
[15] C.W. Caulfield, S.P. Maj, in: M. Iskander (Ed.) Innovative Techniques in Instruction Technology, E-learning, E-assessment, and Education, Springer, 2007, pp. 86 – 91.
[16] J. Wolfe, Decision Sciences, 9 (1978) 143 – 155.
[17] A.P. Raia, The Journal of Business, 39 (1966) 339 – 352.
[18] K.E.F. Watt, Simulation, 28 (1977) 1 – 3.
[19] R.J. Butler, T.F. Pray, D.R. Strang, Decision Sciences, 10 (1979) 480 – 486.

Craig Caulfield is a senior software engineer for a technology consulting company and PhD candidate at Edith Cowan University. His research areas include problem-based learning and the application of serious games to software engineering education and project planning.

Dr. David Veal is a Senior Lecturer at Edith Cowan University. He is the manager of Cisco Network Academy Program at Edith Cowan University and be a unit coordinator of all Cisco network technology units. His research interests are in Graphical User Interface for the visually handicapped and also computer network modeling.

A/Prof S. P. Maj has been highly successful in linking applied research with curriculum development. In 2000 he was nominated ECU University Research Leader of the Year award He was awarded an ECU Vice-Chancellor's Excellence in Teaching Award in 2002, and again in 2009. He received a National Carrick Citation in 2006 for "*the development of world class curriculum and the design and implementation of associated world-class network teaching laboratories*".