Games Used To Teach Software Project Management

KMQuest

KMQuest is an online collaborative simulation game to teach knowledge management aimed at managers/managing directors who are interested in improving company efficiency. The game involves voting for specific courses of action to be taken that directly affect general organisational effectiveness variables such as: customer satisfaction index, profit and market share. The game can have a maximum of three players but can be configured so that a tutor player can have the deciding vote over the other players in the game. Players begin the game with a set budget that can be used to implement plans for intervention and acquire information. The game is played over a period of three years and as it progresses to the end of each quarter the players are confronted with a situation that can affect the knowledge household of Coltec (a fictional company that primarily deals with adhesive solvents). The players must acquire the appropriate information and cast their vote. At the end of each quarter feedback is provided by the calculation of the new values for the business indicators. There is an online training session for KMQuest that consists of a limited form of the game where choices are restricted and guidance, coaching and feedback is provided. KMQuest has been written about extensively in the literature and has a lot of empirical evidence associated with it, as it is the primary focus of two doctoral dissertations and associated publications. One doctoral dissertation focuses on procedural, declarative and general knowledge acquisition and the formation of metacognitive strategies (Christoph, 2007) and one focuses on implicit and explicit knowledge acquisition and the effectiveness of advice (Leemkuil, 2005). The empirical evidence associated with KMQuest is some of the best in the literature as it is specifically looking at whether learners learn from the game, and conducts studies with an appropriate control group. In terms of the pre-test and post-tests for assessing knowledge acquisition, the questions are numerous and extensive, the data is analysed by appropriate statistical techniques and numerous business indicators are brought into account.

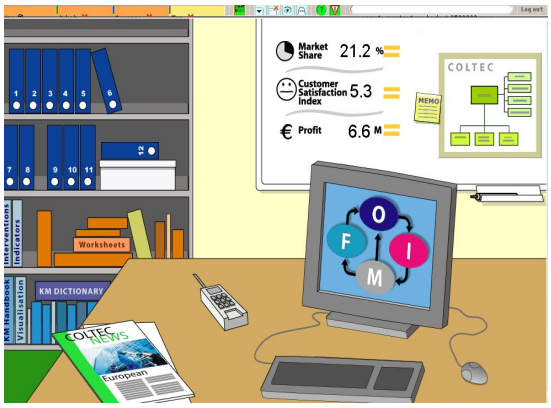

Figure 1: The virtual office and the personal page KMQuest

SimSE

SimSE (http://www.ics.uci.edu/~emilyo/SimSE/) is an educational software engineering simulation environment with the objective of bridging the gap between the large amount of conceptual software engineering knowledge provided to students in lectures and the comparatively small amount they can put into practice in an actual software engineering project. SimSE was developed by Andre van der Hoek, Emily Oh Navarro, Calvin Lee and Beverly Chan. There have been many publications about SimSE regarding its proposal (Oh & Van der Hoek, 2001), its description (Oh Navarro & Van der Hoek, 2004) and proposals for students to build new software process simulation models with it (Oh, Navarro & Van der Hoek, 2005a). SimSE is a single player game where the player assumes the role of the project manager of a team of developers and manages the employees to complete either an aspect of, or a complete software engineering project. The managerial tasks involve the purchasing of tools, monitoring project and employee progress, assigning tasks, and hiring and firing. The actual game is a fully implemented "virtual" office encompassing typical office surroundings, customers, employees and representations of software engineering artifacts including source code, design documents and requirements documents. Information regarding the status of individuals is obtained through speech bubbles that appear over the heads of the characters. This allows players to extract information such as whether a member of staff is happy with their particular salary or whether they are appropriately busy. Information about a characters skills, experience and personal working preferences are obtained by clicking on the player and the information is displayed in a box at the bottom of the screen. Due to the fact that there are different schools though in relation to software engineering, SimSE has a downloadable model builder that enables and instructor to customize the different software engineering processes to be taught. The model builder allows specification of:

- The main entities within the simulation including employees, tools, customers and projects. The main entities can be specified graphically.
- Activities or actions that the main entities participate in, including: requirements documents, design documents, coding, testing and taking appropriate breaks.
- Rules specifying the effects of activities in the overall simulation. For example, when employees are engaging in a particular action such as coding, then the amount of code increases depending on the number of participants. Having too many employees assigned to the one task means that the energy level and productivity of these employees decreases by five percent. The rules specified in the simulation can be predictive (results of causal effects) or prescriptive (allowable sequential steps).
- The start state of the simulation, meaning that the instructor can specify the main entities in the game at the beginning.

There are various different games that have been produced by the SimSE model builder that can be incorporated into the environment (Figure 2) available at the moment including: a waterfall model game, an incremental delivery model game, a rapid prototyping model game, a rational unified process model game, a code inspection game and an extreme programming (XP) model game.

Oh Navarro & Van der Hoek (2005b) performed a small qualitative evaluation of SimSE consisting of 8 questions and 29 participants. The evaluation does not use a control group and is an evaluation based on the perceptions of the participants of an introductory software engineering class. Another problem associated with the study is that the statistical analysis techniques are very limited and only the mean and frequency are present in the paper.

SimJavaSP

SimJavaSP (Shaw and Dermoudy, 2005) is a graphical interactive web-based simulation game designed to increase student's affinity for software development processes by providing experiential learning of to the software development process to enable students to gain greater experience and understanding. The simulation game allows the student to adopt the role of a project manager and supervise the development of a hypothetical software product. The simulation game is based on two software development life cycles with the idea being that the student can gain experience in both of these models in an entertaining and graphically aesthetic manner. The simulation game is implemented in Java and is fully functional. The software development life cycles supported are the standard Waterfall life cycle model and the Spiral Model (Boehm, 1988). Failure in the game means that the players have produced a software project that is unacceptable i.e. not within budget or time constraints.

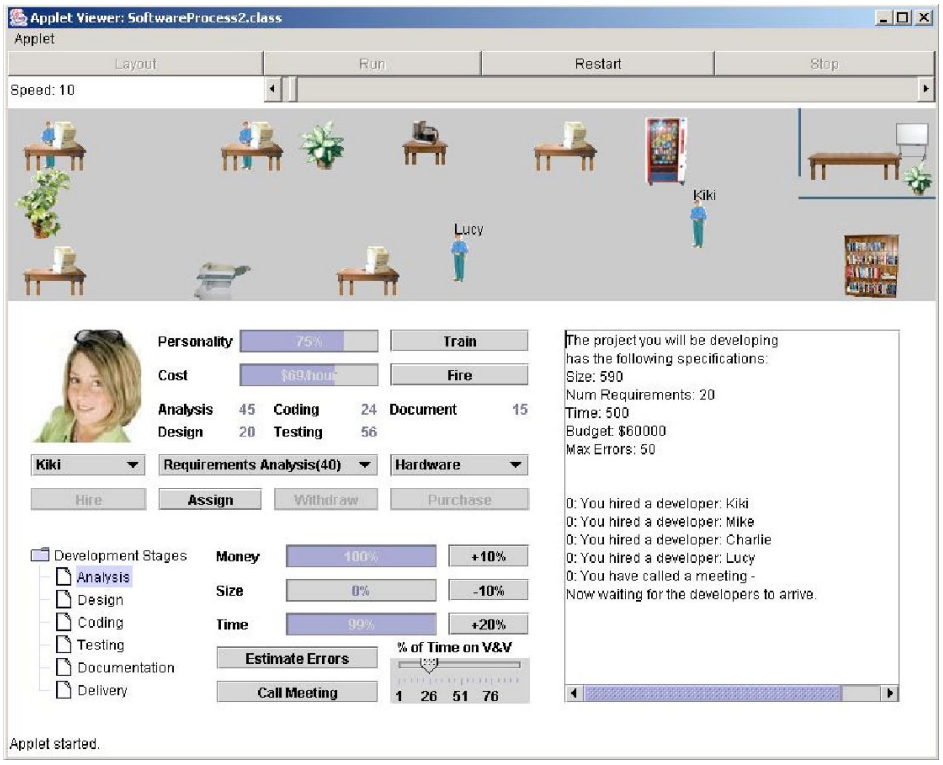

Figure 3: Screenshot of SimJavaSP

The empirical evidence associated with SimJavaSP is a simple small qualitative evaluation using 11 participants. Similarly to SimSE, the questionnaire is primary focused on the perceptions of the learners and no form of control is used to compare the game to traditional techniques of teaching software engineering.

SimVBSE

SimVBSE (http://simvbse.com/) is a game designed to let students better understand the concept of value-based software engineering (VBSE) and its theory (Jain & Boehm, 2006). VBSE involves identification of the value preferences of the success-critical stakeholders of the system, analyzing the different tradeoffs and aspects of the product and project using the stakeholder value considerations and then identifying a solution that satisfies the primary value preferences of the success-critical stakeholders identifying a "win-win" equilibrium. The SimVBSE current prototype is designed using the Sierra Mountain Bikes (SMB) case study and comprises of seven distinct rooms: the CEO's briefing room, the board room, the tactical, the strategic, the back to school room, the lounge and the situation room. The game begins in the CEO's briefing room where the student is asked to assume the role of the project manager and is provided with a brief of the overall project objective and the organisational situation by means of animated videos. The game then moves onto the lounge where the student meets the set of initial system's success-critical stakeholders. The board room is where the student can monitor the stakeholder values and assess the level of satisfaction based on their decisions during the course of the game. The strategic and tactical rooms contain project management controls that the student can monitor. These include exogenous variables (with indirect or no control) or endogenous variables (with direct control). Exogenous variables include: actual number of defects, personnel changes, market changes, technology changes and stakeholder values. Endogenous changes include: investment in tools and technologies, systems level of service, hiring options, investments in process maturity and cost and schedule budgets through negotiation. The strategic room displays organisation metrics, risks, and investments etc and the tactical room displays project metric features and summary detailing such things as where the metric comes from, for example COCOMO II. The situation room presents successive scenarios to the student detailing the finite set of choices, the objective, the scenario description and a decision node. The students then have to make a decision from the choices provided. The back to school room contains tutorials on subjects that are of interest. Although the study does say that a preliminary evaluation gauging enthusiasm towards SimVBSE has taken place, the results are not discussed.

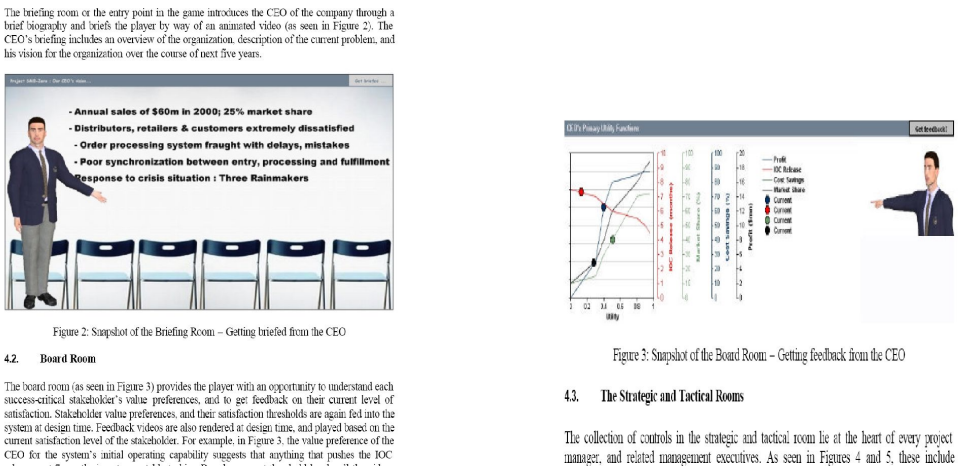

Figure 4: Screenshots of SimVBSE (The briefing room on the left and the board room on the right).

RPG-SE

Role-Playing Game for Software Engineers (RPG-SE) is a 3D multiplayer online software engineering processing game within Second Life (http://slurl.com/secondlife/OHIO%20Outreach/173/190/34). The game was developed from the idea of SimSE and aims at teaching students the principles of software engineering by simulating a software project (Zhu, Wang & Tan, 2007). The game consists of six cubicles and can only have a maximum of six players playing at anyone time. Players join the game by clicking on the chair in a cubicle, forming part of a collaborative team with the goal of releasing the software product to the customer as complete as possible and with as few errors as possible. When the product is released to the customer then the game produces a score between 0 and 100 to let players evaluate their performance. To start the game the player must press the timer which then starts at 300. The player can select a creating task meaning that the artifact progress bar will increase. When the player selects a task that is either reviewing, inspecting or testing then the amount of the artifact errors will increase indicating that there are more errors in the code. If the player then selects a correct task then the amount of the errors in the artifact will decrease indicating that the errors in the artifact have been correct. There is a rest state icon that switches between rest and work states, which the player can use to regenerate their energy. The more that a player works the less energy they have and eventually they contribute nothing to the project. The player can select an integration code task to cause the integrated code bar to increase. There is a status bar above the monitor in the cubicle (each row representing a player) allowing the player to view the progress of a player and their current work. To end the game the player selects the timer icon. The literature review did not reveal any empirical evidence associated with RPG-SE.

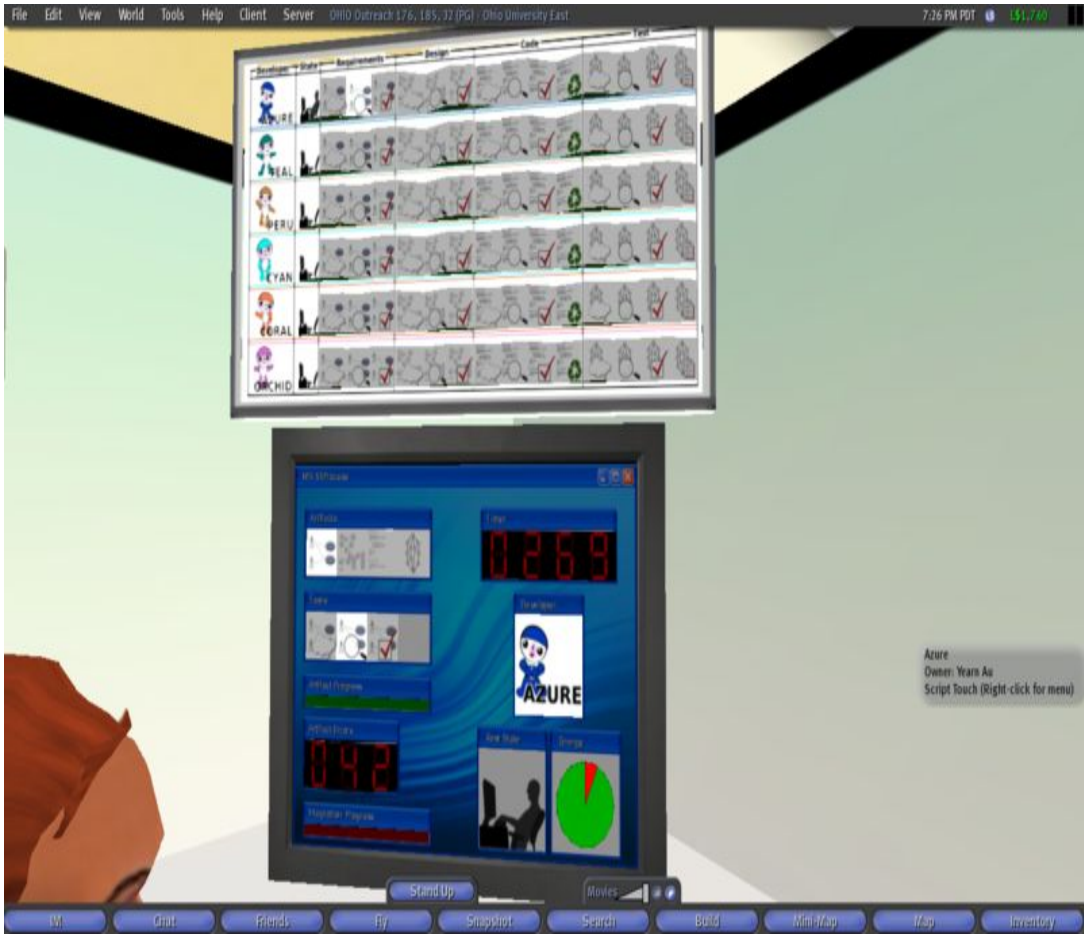

Figure 5: A screenshot of RPG-SE.

The Incredible Manager

The Incredible Manager is a simulation game designed to train software project managers designed and implemented by the Software Reuse Team (Software Reuse Team Homepage, 2008). The player must assume the role of a software project manager and develop software projects within budget, schedule and quality demands. The game consists of three components: the simulation model, the scenario model and the machine. The simulation model represents the world and aspects that will be presented to the player. It contains scenario models providing a generic library of theories and events that can be integrated into a project model by an instructor. The simulation machine controls the steps in the simulation and evaluates the behaviour of system elements based on calculated model equations. The game machine is the component that the player interacts with which runs game phases continuously. Each phase is divided into five steps: The begin phase introducing the project, scenarios that may impact development and project characteristics such as budget. The project planning phases require the player to allocate staff to tasks, decide on the effort required on tasks and effort required on quality assurance activities. The player can modify the project plan at any time. The planning acceptance stage is where the project plan is sent to the stakeholders to be approved. If

the project plan is refused then the player must revert back to the project planning phase until it is accepted. The project execution plan is where the project plan is executed in continuous turns consuming project resources. The player has to take corrective action in this phase in response to such things as exhausted developers or altering the original project plan. The end phase occurs when the project resources are done (project failure) or the project is completed successfully. There are three different characters in the game: the manager, the boss and the developers. The manager is the role assumed by the player and is responsible for all of the decision-making processes. The boss is effectively the stakeholders and is responsible for the acceptance or rejection of the project plan. The developers are the team developing the project and have different characteristics and skills such as work hours per day and hourly rate of pay.

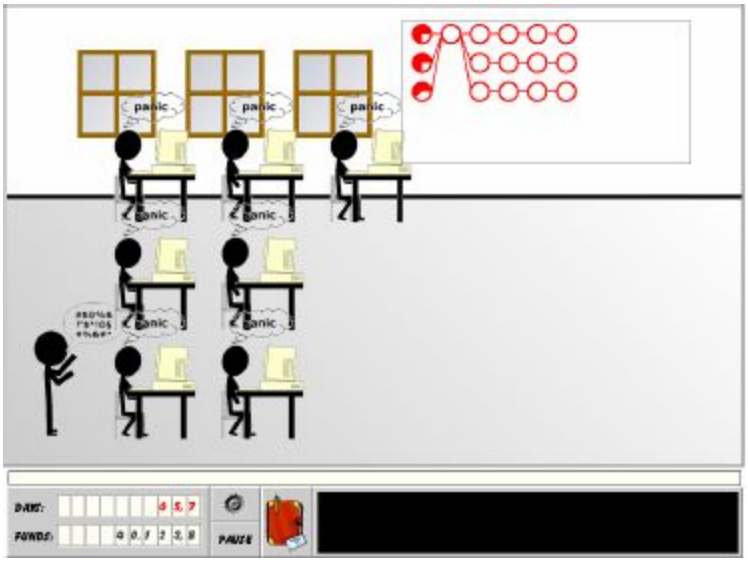

Figure 6: The office room of The Incredible Manager.

Dantas, Barros & Werner (2005) perform a small qualitative evaluation involving 11 BSc students and 13 MSc/DSc students. The evaluation takes the form of a post-test questionnaire where the results of the questionnaire are qualitative and are not supported by any quantitative statistical results. The qualitative results are not discussed in sufficient detail. Dantas, Barros & Werner (2004) describes a small qualitative evaluation consisting of 15 participants and 5 questions. Once again the primary flaw is that there are no quantitative statistical results obtained and the evaluation is too small to provide any detailed insight.

MIS Project Manager

Martin (2000) describes the evolutionary design of MIS Project Manager (http://www.hear-see-do.com/mis/home.asp) formerly known as Information System Project Manager which is a simulation game produced for the instruction of Information Systems (IS) development from a managerial perspective. The game has the following learning objectives: Stimulation of understanding and awareness of issues, concepts and languages associated with IS development, to increase appreciation of the diverse

approaches to IS development and also to appreciate the differences of these approaches, to increase appreciation of the dynamic practicalities and tasks associated with IS and to present them in an incorporated view, to increase confidence and experience in relation to accepting consequences associated with decisions, deal with unforeseen eventualities in project management and deal with acceptable trade-offs the must be made between conflicting tasks and to appreciate that as well as having to consider the hard school of information systems, the soft school must also be considered. This means that aspects such as staff morale must be brought into account. The simulation game has events that are stored in a database and are displayed and organised in relation to the player's decisions. As a result of this the player is only presented with the relevant events that occur rather than being exposed to all of the events that could possibly occur. Players are presented with forms that display: event occurrences, performance indicators – represented by percentage scores and gauges that are automatically updated in relation to decisions that are made. The indicators display scores for harder ratings (technical quality, time, cost, security performance) and softer ratings (staff morale, user acceptability) and project network and Gantt charts indicating project activity floats and current progress. The game presents a series of events that take place in an interactive fashion i.e. the player is shown the events as they occur over the days and weeks of the project. The following two varieties of event can occur in the game: a reporting information event, for example, "Activity 12 – coding has just started." and a planned or unplanned event – these events are directly related to information systems development. The events are designed to be realistic, as they have been extracted from a variety of sources such as: the academic literature, project managers, software engineers, common sense and experience. The events are also selected to give a wide variety of problems including technical problems, operational problems, security issues, delays, disasters, staffing problems and changes to requirements. Events occur with direct correlation to time, staff morale and quality. Players can select a number of options to respond to occurring events. All of the options have consequences and affect such things as scheduling, user acceptability, quality, moral and may trigger other events. The database holds 195 different events and supports interrelated events that are linked together. In Martin's (2000) study, the project that was simulated had eighteen major activities for the purposes of sufficient complexity in terms of the activity network. Players of the game where generally arranged into groups of two or three, however the game could be played on an individual basis with a simulated target of two hundred and eight-two days to completion of the project with a budget of £1,100,100. The primary objective of the project was to make the appropriate decisions for each of the events to achieve optimal performance with regards to one or more of the project indicators. It should be noted that the scoring system is based on the project indicators and finishing the project first does not necessarily mean that the greatest score is achieved. Time in the game is advanced automatically and is also interrupted by the events that take place. It should be noted that players have the option of monitoring their progress at any point in the proceedings. No empirical evaluation of MIS Project Manager was conducted in this study or has been discovered in the literature.

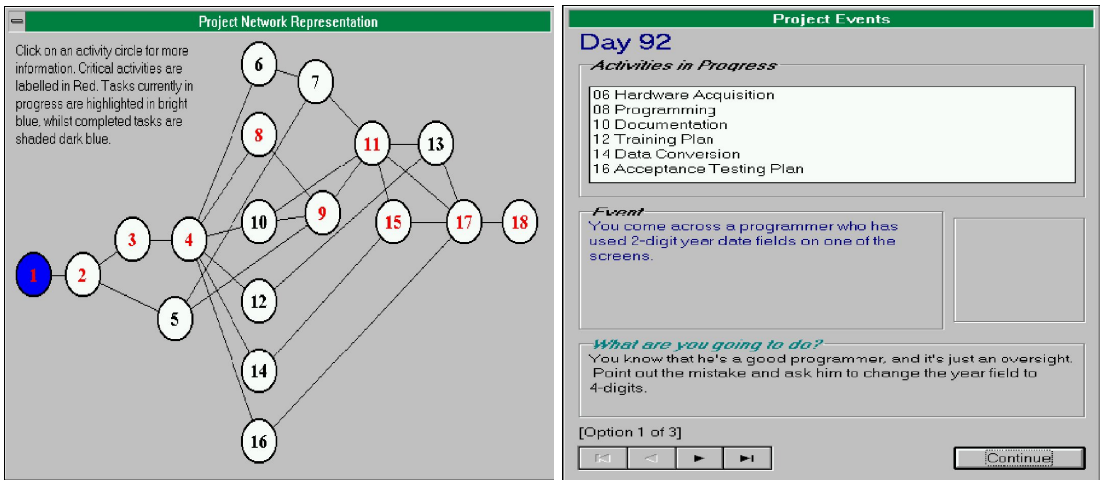

Figure 3.29: Screen shots of MIS Project Manager (Activity network on the left and project event occurrence on the right)

Open Software Solutions

Sharp and Hall (2000) developed Open Software Solutions to be a major part of the Open University's M880 software engineering distance education course aimed at software professionals. The case study section of the course, which consumes 80 hours of study is presented through the Open Software Solutions multimedia game. The game is based on a company called Open Software Solutions founded in 1978 and focuses on four projects: quality accreditation based on consultancy work, production cell based on a standard problem in computer science research and teaching, CIRCE based on a software system that was developed by the Open University for management purposes which is presented as a consultancy project to establish a development method, and SummerSun - a fictional travel agency. Each floor in the multimedia game contains a one of the four different projects that are reachable by the use of a lift or quick access keys. The floors have basically the same layout and contain all of the attributes that the player requires to complete the project such as books, access to meetings and simulations etc. In quality accreditation the students draft quality procedures that conform to a pre-prepared quality manual. In the CIRCE case study the player must conduct a series of interviews with the goal of completing an evaluation form. There are 35 available questions to ask and 12 available individuals to interview, meaning that the player must exercise their judgment in terms of whom to ask which questions to produce the evaluation form as quickly and efficiently as possible. In Production Cell the students experiment with a simulation to build their problem solving skills then model the observed behaviour using statecharts and Z mathematical notation. In SummerSun students conduct structured interviews with clients to gather requirements and then produce an ER diagram using a CASE tool. In Quality Certification the students draft quality procedures to conform to the technical directors standards and a pre-prepared manual. The empirical evidence associated with Open Software Solutions consists of a qualitative evaluation using the Measuring the

Usability of Multi-Media Software (MUMMS) questionnaire with 31 participants. There is a second qualitative evaluation discussed with 126 questionnaires discussed briefly. The main problems with this study are that no appropriate form of control is used and the analysis is not sufficiently rigorous with detailed statistical analysis.

References

Boehm, B. (1988). A Spiral Model of Software Development and Enhancement. *Computer*, *IEEE*, 21(5), 61–72.

Christoph, N. (2007). *The role of metacognitive skills in learning to solve problems.* PhD Thesis submitted to the University of Amsterdam. Retrieved 28th November, 2007 from http://dare.uva.nl/document/22568

Dantas, A.R., Barros, M.O. & Werner, C. (2005). Simulation Models Applied to Game-Based Training for Software Project Managers. In *Process Simulation and Modeling Workshop (ProSim)*, St Louis, USA, May 2005, 110-116.

Dantas, A.R., Barros, M.O. & Werner, C. (2004). A Simulation-Based Game for Project Management Experiential Learning. In *Sixteenth International Conference on Software Engineering and Knowledge Engineering (SEKE'04)*, Alberta, Canada, June 2004, 19-24.

Jain, A., & Boehm, B. (2006). SimVBSE: Developing a Game for Value-Based Software Engineering. In *Proceedings of 19th Conference on Software Engineering Education and Training (CSEET).* Turtle Bay Resort, Oahu, Hawaii. 103–114.

Leemkuil, H. (2005). *Is it all in the game? Learner support in an educational knowledge management simulation game*. Unpublished doctoral thesis. University of Twente, Enschede, The Netherlands.

Martin, A. (2000). The design and evolution of a simulation/game for teaching information systems development. *Simulation & Gaming*, 31(4).

Oh Navarro, E. & Van der Hoek, A. (2005a). Software Process Modeling for an Educational Software Engineering Simulation Game**.** *Software Process Improvement and Practice.*

Oh Navarro, E. & Van der Hoek, A. (2005b). Design and Evaluation of an Educational Software Process Simulation Environment and Associated Model. In *Proceedings of the Eighteenth Conference on Software Engineering Education and Training*, Ottawa, Canada.

Oh Navarro, E. & Van der Hoek, A. (2004). SimSE: An Interactive Simulation Game for Software Engineering Education**.** In *Proceedings of the 7th IASTED International Conference on Computers and Advanced Technology in Education,* Kauai, Hawaii, August 2004.

Oh, E. & Van der Hoek, A. (2001). Adapting Game Technology to Support Individual and Organizational Learning. In *Proceedings of the 13th International Conference on Software Engineering and Knowledge Engineering,* Buenos Aires, Argentina, June 2001.

Sharp, H., & Hall, P. (2000). An Interactive Multimedia Software House Simulation for Postgraduate Software Engineers. In *Proceedings of the 22nd international conference on software engineering.* Limerick, Ireland.

Shaw, K. & Dermoudy, J. (2005). Engendering an empathy for software engineering. In *Proceedings of the 7th Australasian Computing Education Conference (ACE2005),* Newcastle, Australia. 42, 135–144.

Zhu, Q., Wang, T., & Shenglong, T. (2007). Adapting Game Technology to Support Software Engineering Process Teaching: From SimSE to Mo-SEProcess. In *Proceedings of Third International Conference on Natural Computation (ICNC).* 777–780.