SimVBSE: Developing a Game for Value-Based Software Engineering

Apurva Jain and Barry Boehm
University of Southern California
Los Angeles, CA 90089
{apurvaja, boehm}@usc.edu

Abstract

The development of games in aid of improving and enriching a student's learning experience is again on the rise. The beer game [6] in the field of system dynamics was developed to instill the key principles of production and distribution. SimSE [5] provides a simulated game for its players to take on the role of a project manager, and experience the fundamentals of software engineering through cause-effect models. In this paper we present an initial design of SimVBSE as a game for students to better understand value-based software engineering [1], and its underlying theory [3].

1. Introduction

Technology advancements with burgeoning pursuits towards enhancing a student's learning experience has brought educational games to the research forefront. The beer game [6], SimSE [5], and the Army's training simulation [4], have all reported promising results in using such games as a significant value-addition to an instructor's pedagogical toolkit. This paper describes an initial attempt to develop a game that embodies the principles of value-based design in the field of software education. The paper is organized around five sections. Section 2 briefly introduces value-based software engineering (VBSE) as in [1] and its theory [3]for readers not familiar with VBSE; Section 3 discusses the objective and challenges of the game, and presents the high-level design of the game; Section 4 explains the significant aspects of the game, followed by a conclusion and discussion of future work in Section 5.

2. Value-based Software Engineering

Value-based software engineering's primary thesis is that much of current software engineering practice and research is done in a value-neutral setting, in which:

- Every requirement, use case, object, test case and defect is treated as equally important;
- Methods are presented and practiced as largely logical activities involving mappings and transformations (e.g., object-oriented development);
- Earned value" systems track project cost and schedule, not stakeholder or business value;
- A "separation of concerns" is practiced, in which the responsibility of software engineers is confined to turning software requirements into verified code.

Most studies of the critical success factors distinguishing successful from failed software projects find that the primary critical success factors lie in the value domain. The goal of value-based software is to integrate value considerations into the full range of existing and emerging software engineering principles and practices. This primarily involves identifying the success-critical stakeholders of the system and their value preferences; reasoning about the tradeoffs and different aspects of the project and the product using stakeholder value considerations; and identifying a win-win equilibrium that satisfies all success-critical stakeholders primary value preferences.

The underlying VBSE theory establishes this win-win equilibrium as a necessary and sufficient condition for a successful software-oriented enterprise. It provides a set of steps and decision aids for identifying the enterprise's success-critical stakeholders for determining their value-propositions, for enabling stakeholders to negotiate mutually satisfactory or win-win enterprise objectives and plans, and for adaptively controlling the project and adjusting the plans to maintain a win-win equilibrium as the enterprise encounters various forms of emergent needs and external changes. We have been using this approach to teach a software engineering project course using campus service providers and local companies as clients over the last 10 years, with a 92% success rate in delivering client-satisfactory applications within tight budget and schedule constraints [2]. However, we have found that it takes some time for students to assimilate the theory and practices, and have embarked on developing SimVBSE as a way to help students learn this approach.

3. SimVBSE: An Overview

3.1. Objective and challenges

Designing a game that can represent a real software project in its entirety is not feasible – this is mainly because of the unique nature of different projects, the algorithmic complexity that is rooted in the multiple combinations of problems and solutions, the involvement of the human element, and the no-silver-bullet phenomenon. We had to identify a good mix of capturing what's set out for our software engineering students after they graduate, imbibing the principles of value-based software engineering, and illustrating application of theory to practice.

Similitude to reality. Synthesizing an environment that can significantly capture reality is imperative – the game must provide students with the opportunity to experience what's waiting for them as they embark on their software engineering professional career. While the idea behind doing this is similar to any other games with virtual reality – provide a fun learning experience; we also note that this also impacts in motivating the students to optimize their learning of the subject.

Learning what to learn and reasoning. What works in project A, or situation X, might not work for project B, or situation Z The software engineering community has long been grieving the outcomes of uncritically adopting so-called silver bullets. Reasoning before taking an action allows one to make an informed decision, and usually does more good than harm. However, with current economical trends where time is the essence of success, too much learning is also unforgiving. We wanted to provide students with the opportunity to acquire knowledge and resources that will assist them in their game play, but also highlight the hidden costs underlying learning. For example, buying information to reduce risks is only effective to the extent it remains compatible with one's market and resource constraints.

Assessment and feedback. Failing to provide quick assessments with relevant feedback can seriously reduce the learning of the students, and the overall game's effectiveness. Many educationally motivated games usually provide a final score, or a post-game analysis report to its players. However, we believe that not providing quick assessments and feedback affects the overall effectiveness of the game due to our limited short-term memory capacity.

Adherence to VBSE and theory validation. This game was conceptualized as a pedagogical aid in teaching value-based software engineering to students. We felt that this also provides an excellent medium to test our hypotheses about the theory of value-based software engineering. Designing different case studies (analogous to maps in traditional games), and a player's post-game feedback on encountering failure modes in the theory will help us strengthen ongoing VBSE research and generate insights for the VBSE community at large in orienting their future research agenda.

3.2. Game design

The current (prototype) version of SimVBSE has been designed using the Sierra Mountainbikes (SMB) supply chain case study as presented in [Boehm and Jain 2005]. We used the SMB case study to design the initial prototype of SimVBSE to leverage on the familiarity our software engineering students had with the study through their coursework, and to be able to assess the game's efficiency in explaining the concepts of VBSE as compared to providing relevant publications.

SimVBSE in its current version comprises of seven distinct rooms – these are the CEO's Briefing Room, the Board Room, the Strategic, the Tactical, the Back to School, the Lounge, and the Situation Room. The game starts with a visit to the CEO's briefing room wherein the student through animated videos is asked to assume the role of a project manager, and briefed on the current organizational situation and the student's overall objective in the game.

Following the CEO's briefing, the student visits the Lounge and gets an opportunity to meet the initial set of the system's success-critical stakeholders. The next room, the Board Room, essentially serves as a mirror to different stakeholder values, and their current level of satisfaction based on the students decisions through the course of the game. The Strategic and the Tactical Rooms are windows to the project management controls for the student. Both these rooms have been designed to display a collection of exogenous (variables with indirect or no control) and endogenous (direct control) variables such as the development team and its profile, agreed list of system features, current bank balance etc.

The Situation Room is the action hub of SimVBSE. This room presents a comprehensive set of single-thread successive scenarios to the student. Each scenario typically includes scenario description, its objective, a decision node, and a set of finite choices. Students are then asked to make an exclusive choice from the options provided, and encouraged to use the Back to School room for quick tour of tutorials on subjects of immediate interest.

3.3. Game flow

Figure 1 provides a high level dynamic structure of the game. Making a move in the game then involves visiting the different rooms as seen towards the right in figure, and choosing one or more of the available options as seen towards the left in the figure. Visiting rooms does not consume the player any resources however each option in the left may cost a fixed amount of resources. For example, changing project parameters will not have a single fixed cost however it will have an effect on the project metrics, organizational metrics, and the project risks. On the other hand, applying a process such as the results chain analysis [7] will cost a fixed amount of resources, but will also positively impact risks and projects metrics since results chain analysis typically reduces a project's requirements volatility. Making strategic investments like applying processes will have similar outcomes in that they may have a positive or negative effect on the risks, and they can also affect the project and organizational metrics.

The underlying game engine in its current (prototype) version is a rule-based engine that follows certain cause and effect rules that have been defined at design time. The game engine is mainly responsible for changing the parameters in context of the player's chosen action, applying transformational rules that generate effects of changing the parameters on other dependent controls, and accordingly updates them too.

The rooms as discussed earlier either provide different views of the project, or provide non-project specific information such as tutorials or consultant and vendor sales pitches.

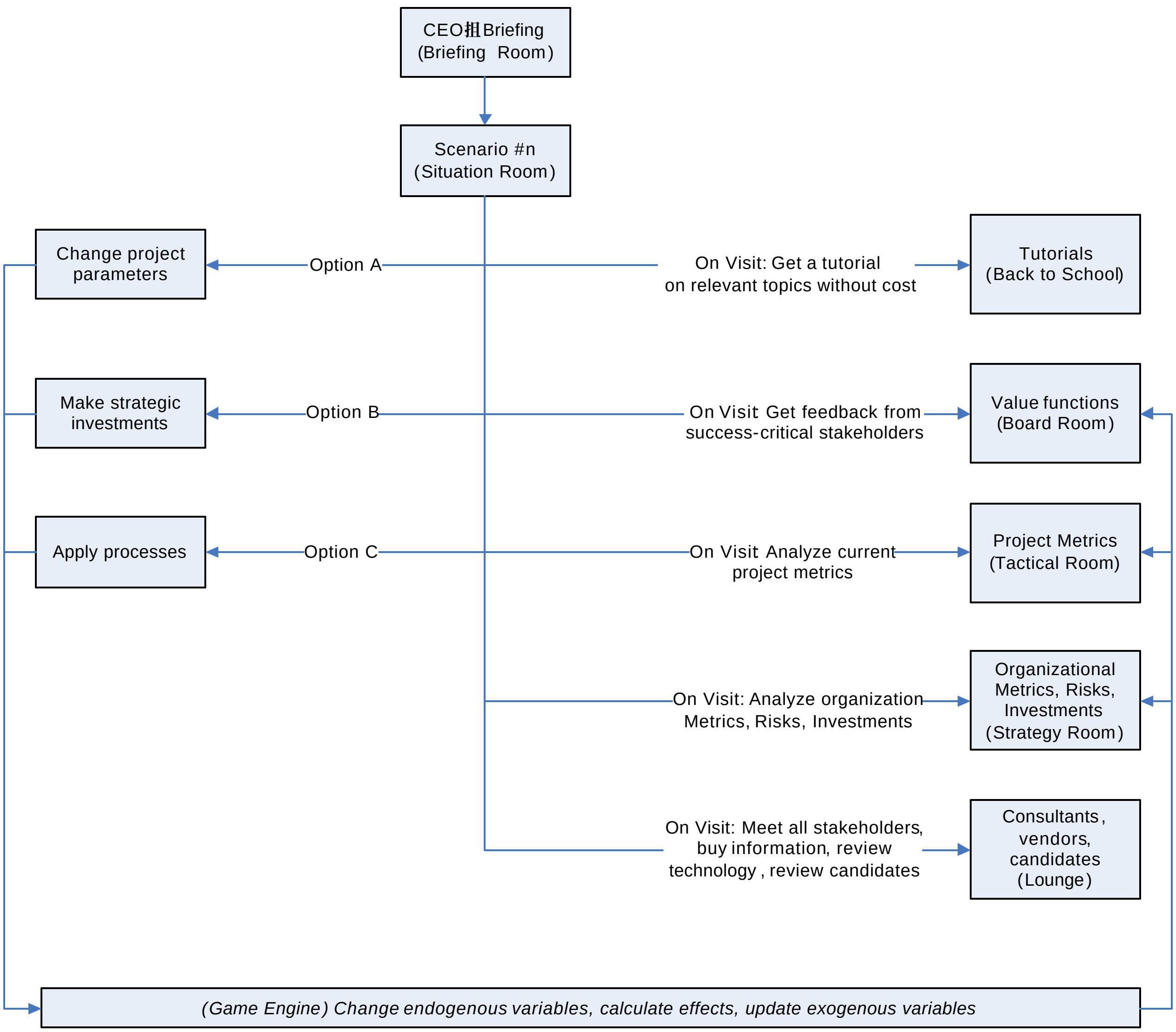

Figure 1: SimVBSE Architecture

4. SimVBSE: The Rooms

4.1. Briefing room

The briefing room or the entry point in the game introduces the CEO of the company through a brief biography and briefs the player by way of an animated video (as seen in Figure 2). The CEO's briefing includes an overview of the organization, description of the current problem, and his vision for the organization over the course of next five years.

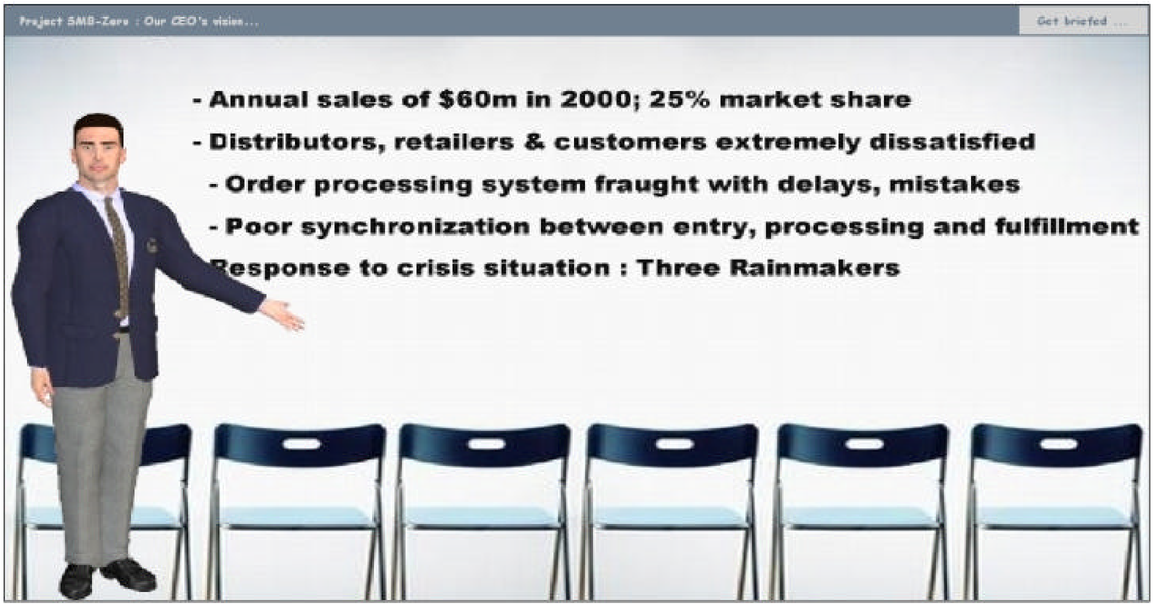

Figure 2: Snapshot of the Briefing Room – Getting briefed from the CEO

4.2. Board Room

The board room (as seen in Figure 3) provides the player with an opportunity to understand each success-critical stakeholder's value preferences, and to get feedback on their current level of satisfaction. Stakeholder value preferences, and their satisfaction thresholds are again fed into the system at design time. Feedback videos are also rendered at design time, and played based on the current satisfaction level of the stakeholder. For example, in Figure 3, the value preference of the CEO for the system's initial operating capability suggests that anything that pushes the IOC release past 7 months is not acceptable to him. Based on preset threshold levels, all the videos that are relevant to the situation are queued up in the feedback queue and are made available for the player to watch.

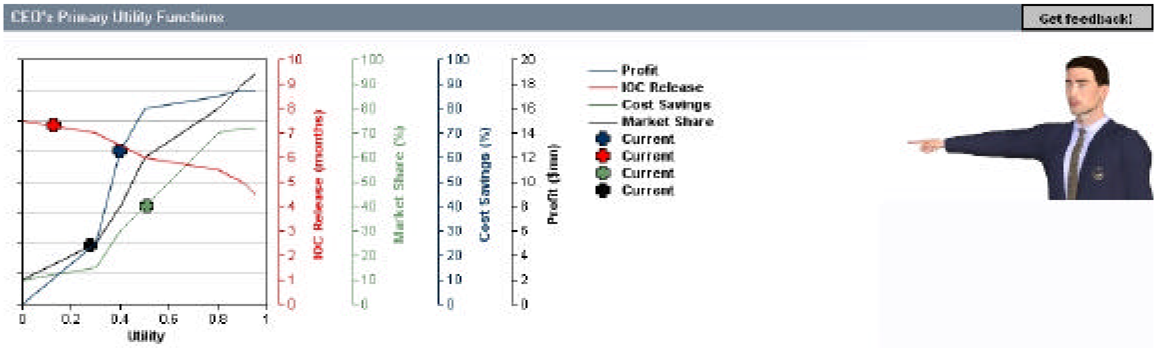

Figure 3: Snapshot of the Board Room – Getting feedback from the CEO

4.3. The Strategic and Tactical Rooms

The collection of controls in the strategic and tactical room lie at the heart of every project manager, and related management executives. As seen in Figures 4 and 5, these include organizational scorecards, project scorecards, and a top-n risk list. An interesting aspect of these controls is the distinction between the exogenous and the endogenous controls as in Table 1. Organizing the metrics and scorecards in such a way typically allows a player to understand his or her direct control over some variables and indirect or no control over others. While in the current version of the game we do not allow directly modifying any of the parameters in both the rooms, we do however encapsulate these potential changes within options available for each scenario.

PROJECT METRICS

Features #1 through #n	
Name	*(feature name)*
EAF	*(from COCOMOII estimates)*
Cost	*(budgeted vs. current estimate vs. actual)*
Schedule	*(budgeted vs. current estimate vs. actual)*
Effort	*(budgeted vs. current estimate vs. actual)*
Defects	*(preferred rate vs. current known rate)*
Risks	*(from risk list)*
Priority	*(feature priority)*

Summary	
Features	*(# of features)*
SF	*(from COCOMOII estimates)*
Cost *(Total)*	*(budgeted vs. current estimate vs. actual)*
Schedule *(Total)*	*(budgeted vs. current estimate vs. actual)*
Effort *(Total)*	*(budgeted vs. current estimate vs. actual)*
Defects *(Avg)*	*(preferred rate vs. current known rate)*

Figure 4: Structure of the Tactical Room

Exogenous	Endogenous
– Personnel changes (e.g. developers quit) – Market changes (e.g. change of consumer preference, innovation etc.) – Technology change (e.g. obsoleteness, innovation etc.) – Stakeholder values (e.g. what they are, how important they are) – Actual cost and schedule – Actual number of defects	– Hiring options (e.g. adding personnel with relevant experience, stronger analytical capability) – System's level of service (e.g. reliability) – Investments in process maturity (e.g. CMM assessment) – Investment in tools and technologies (e.g. automated test data generator) – Cost and schedule budgets through negotiation (e.g. project scoping)

Table 1: Examples of exogenous and endogenous controls

ORGANIZATIONAL METRICS

FINANCIAL	
Metrics #1 through #n	
Name	*(name of the metric)*
Data	*(benchmarked vs. current est. vs. preferred)*

CUSTOMER…
PROCESS…
LEARNING AND GROWTH…

RISKS

MARKET RISKS	
Risks #1 through #n	
Description	*(description of risk and its effect)*
P(L)	*(probability of loss)*
S(L)	*(size of loss)*
RE	*(current vs. preferred)*
Rank	*(risk rank)*

TECHNOLOGY RISKS…
PERSONNEL RISKS…
COST & SCHEDULE RISKS…
REQUIREMENT MISMATCH RISKS…

INVESTMENTS

TOOLS	
Tools #1 through #n	
Name	*(name of the tool)*
Description	*(expected benefit of the tool)*
Outcome	*(actual benefit of the tool)*

TECHNOLOGIES…
PROCESS…
INFORMATION…

Figure 5: Structure of the Strategic Room

4.4. The Lounge and the Back to School Room

The lounge provides a collection of external resources that are made available to the project manager but usually at a fixed cost. These include resumes of potential candidates, pitch talks by consultants and vendors that either showcase a technology, or their consulting experience. The back to school room is a library of presentations and publications that a player may consult before making decisions.

4.5. The Situation Room

The situation room is the action hub of the game. Students here are presented with single-thread scenarios with each scenario focusing on an aspect of software engineering such as dealing with volatile requirements or introducing late changes. Resolving the problem then requires one or

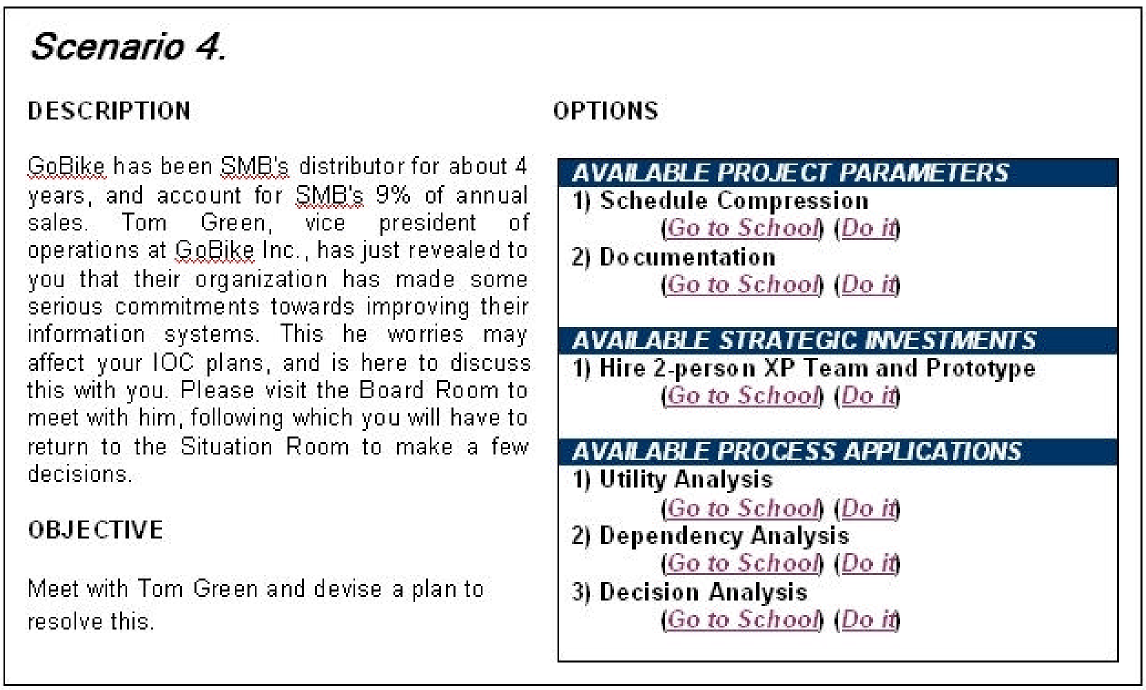

Figure 6: Example scenario in the Situation Room

multiple actions in the three given categories, (a) relevant project parameters that are available for change, (b) list of possible investments that may resolve the situation, and (c) relevant "4+1" processes as mentioned in [3]. In the example provided in Figure 6. the student after reading the description is asked to visit the board room and watch the video of Tom Green explaining his change proposal. The above example however does not show another side screen that displays further description of each choice, such as the up-front cost (e.g. cost of hiring a 2-person XP team for a month) or change in project parameters (e.g. change in effort due to increased or decreased compression) due to making the change. Following each scenario, we also provide an assessment report that discusses the strengths and weaknesses of each of the options that was made available to the student based on his or her choices. For example, if Tom Green's proposal required a small change, but the student made the decision to compress an already highly compressed schedule, then he would get a negative evaluation that explains the significant increase in project's schedule risk, and also a recommendation to apply dependency analysis in future situations to determine the major cost, schedule, quality and risk tradeoffs the proposed change would involve.

5. Conclusions and future work

An initial evaluation of the game through peer feedback and in-class demonstrations has generated significant enthusiasm for the developers and the software engineering class in general. Our future work on the game is directed towards modeling various other case studies, and having software engineering students playing them. We also hope to release a stable version of the game for the software engineering community in the near future.

6. References

[1] B. Boehm, "Value-Based Software Engineering: Overview and Agenda" in S. Biffl, A. Aurum, B. Boehm, H. Erdogmus, P. Gruenbacher (eds.), Value-Based Software Engineering, Springer Verlag, 2005.

[2] B. Boehm, A. Egyed, J. Kwan, D. Port, A. Shah, R. Madachy, "Using the WinWin Spiral Model: A Case Study" , Computer, July 1998, pp 33-44.

[3] B. Boehm and A. Jain, "An Initial Theory of Value-Based Software Engineering" in S. Biffl, A. Aurum, B. Boehm, H. Erdogmus, P. Gruenbacher (eds.), Value-Based Software Engineering, Springer Verlag, 2005.

[4] J. Gratch and S. Marsella, "Fight the Way You Train: The Role and Limits of Emotions in Training for Combat" , The Brown Journal of World Affairs. Vol X (1), Summer/Fall 2003.

[5] E. O. Navarro and A. van der Hoek, "Teaching Software Engineering Using Simulation Games", Proceedings, International Conference on Simulation in Education, San Diego, 2003.

[6] J. Sterman, "Modeling Managerial Behavior: Misperceptions of Feedback in a Dynamic Decision Making Experiment", Management Science, 1989, 35(3), pp 321-339.

[7] J. Thorp, The Information Paradox, McGraw Hill, 1998.