RISKWARE: A GAME FOR TEACHING SOFTWARE PROJECT RISK MANAGEMENT

Carlos Mario Zapata Jaramillo
Universidad Nacional de Colombia
cmzapata@unal.edu.co

María Clara Gómez Álvarez
Universidad de Medellín
mcgomez@udem.edu.co

Guillermo González-Calderón
Universidad de Nacional de Colombia, Universidad de Medellín, Colombia
ggonzal@unal.edu.co

ABSTRACT

Risk can highly affect the management of software development projects, because risky situations commonly arise in this kind of projects. Well-known strategies for teaching software engineering are not enough for developing management skills of the students and, particularly, risk management is a difficult issue for practicing in the classroom. Since experience-based games have been successfully designed and applied to teaching, we propose in this paper riskware, an experience-based game for teaching and practicing risk management issues. We also apply the game to a group of undergraduate students belonging to the system engineering program. The results are promising and can be replicated to similar knowledge areas.

INTRODUCTION

Risks, in the context of software projects, are defined as uncertain events or conditions which, if they occur, have either a positive or negative impact on the project goals (Project Management Institute, 2000). Risk management emerges as an important discipline, because projects can be negatively affected by risks and, consequently, software application quality can be endangered.

The main concern of risk management is identifying, studying, and eliminating the risk sources before they begin to threaten the needed accomplishment of a software project. Hence, risk management should be considered a mandatory component of the software project management courses (McConnell, 1997), by using well-known software engineering teaching methods like lectures and "toy" practical projects (Baker *et al.*, 2005). However, such methods are not enough to provide the technical and managerial skills needed by students in the context of software engineering risk projects. Hence, experience-based games have been suggested for complementing software engineering teaching methods. Games are modern student-centered pedagogical tools, *i.e.*, students are active partners of the teacher and their classmates, and they are guided by the teacher for discovering knowledge. Problems and programmers (Baker *et al.*, 2005), requirements game (2009), and risk management game (Taran, 2007) are some examples of these games.

Based on past work, in this paper we propose *riskware*, an experience-based game for teaching software project risk management. Players—acting like software companies—should finish a software project by defining a strategy for dealing with risks while they are performing the project.

This paper is organized as follows: first, we discuss the importance of software project risk management; then, we present games as learning strategies; next, we propose riskware, a game for teaching risk management skills; after that, we discuss some results of the game application; finally, we summarize conclusions and future work.

IMPORTANCE OF RISK MANAGEMENT IN SOFTWARE ENGINEERING

Risk is defined as the possibility an adverse event, misfortune, or setback causes a loss (Pressman, 2005). Since risk is only a future possibility, some management practices can prevent risks to be materialized. In software development projects, risks can materialize in the form of low quality of the software application, unaccomplished goals and planning, and software project failure. Accordingly, risk management process is one of the best practices for identifying, studying, and eliminating risk sources before the successful project accomplishment is threaten by them (Pressman, 2005).

By using best practices, we can also identify potential risks, classify them, and design strategies for dealing with them. In this way, we can diminish uncertainty, even though risk management is a barely used activity in Latin American software development projects (Gasca, 2010).

Table 1 lists some risks and the possible mitigation strategies (Sommerville, 2005).

American Systems Corporation (2012) defines the following set of the main rules related to the software project risk management:

1. Rule 1. Projects without risk management are actually making risky progress.
2. Rule 2. Risk management is not for free. Organizations need to compromise resources, establish plans and processes, and keep emergency funds.
3. Rule 3. Risk management responsibilities should be centralized, in order to avoid coordination of a distributed management.
4. Rule 4. Organizations should prioritize and focus on critical risks. However, all risks should have a mitigation strategy.
5. Rule 5. Project managers are responsible for taking actions in the software development process, while project risk managers are responsible for identifying and controlling risks.
6. Rule 6. Risk management process should be defined and consistently followed throughout the entire organization. All actions should be devoted to accomplish risk management organization policy requirements.

GAMES AS LEARNING STRATEGIES

A game is a closed formal system subjectively representing a subset of the reality, in which every run of the game generates a new version of the history. The final outcome depends on the global conditions of the game, the features of the players, and the interaction among players (Kasvi, 2000). Bushell, as cited by Dempsey *et al.* (1996) defines gaming as an interactive activity intended to replicate expected real-world conditions. Such activity stimulates the decision-making process. Some behavioral rules are accepted by the players in the competition, and they make decisions affecting themselves and the other players (Dempsey *et al.*, 1996). In order to be considered "educational," a game should exhibit the following features (Burgos *et al.*, 2006): (i) a premise to-be-solved as a starting point; (ii) at least one true solution; (iii) at least one learning strategy—e.g., new knowledge generation, previously acquired knowledge reinforcement, skill exercise, concept discovery, creative development, and experience sharing.

According to the previous definitions, some authors propose the following advantages for using games in the teaching-learning process:

1. Games ease the learning-by-doing approach, because they develop communication skills and consider emotions in the learning process. Consequently, they promote learning by peer interaction (Kober & Tarca, 2000).
2. Critical thinking, communication skills, debate, and decision making are developed by the learning-by-games strategy in a practical way substantially different from the theoretical approach of lectures (Baker *et al.*, 2005).
3. We can increase with games the speed of learning and improve the knowledge acquisition (Klassen & Willougby, 2003). By working in teams, students are promoted to achieve consensus on tasks, goals, and methods (Laneima, 2004).

RISKWARE GAME DESCRIPTION

GAME GOALS

We aim to achieve the following goals after playing riskware:

1. Recognizing the importance of software project risk planning before the project start up (proactive management) in order to early take actions for mitigating risks.
2. Identifying risks, as well as mitigating resources and controls, linked to software development projects.
3. Recognizing two factors—occurrence probabilities and impact measurement—for mitigating software development project risks.

GAME EQUIPMENT

1. Board: contains a honeycomb structure divided into three representative stages of software development

Table 1. Some risks and their mitigation strategies

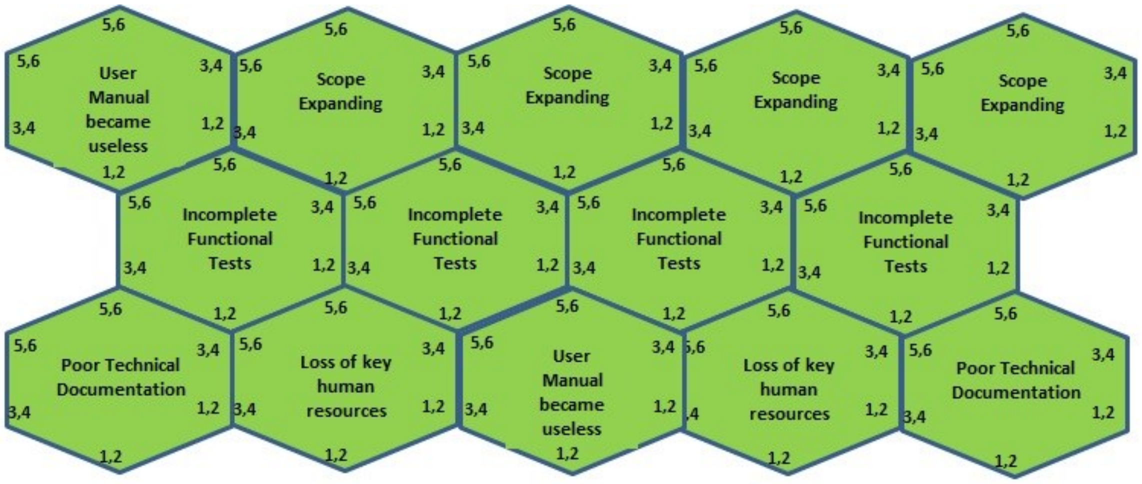
Risk	Mitigation strategies
Too many require-ments changes	Definition of a process for managing requirements changes. Agreement between analyst and stakeholders about policies for requirements change management.
Lack of technically qualified personnel	Creation of a repository of qualified personnel. Definition of training activities inside a software development project.
Lack of business knowledge	Establishment of continuous stakeholder interaction schemes. Definition of stakeholder-development-team knowledge transfer plans.
Poor performance	Definition of individual improvement plans. Establishment of supporting additional resources.

projects: definition and analysis; design and implementation; maintenance and operation.
2. Hexagonal pieces: represent 30 risks commonly associated with stages of the software development process and distinguished by colors (see Exhibit 1). Each piece has pairs of numbers in the corners in which way that the joint of three pieces completes the set of numbers 1–6. The numbers are used for defining the materialization of a risk.
3. Die: a common 6-sided die
4. Path control fold: a facsimile of the board in which players can define a path to face the game (see Exhibit 2).
5. Game instructions fold: contains twelve short rules for understanding and remembering the game instructions.
6. Risk and payment sheets (see Tables 1 and 2): risk sheet describes the risks, the mitigating resources and controls, and the impact. Payment sheet includes the exponential payment related to each level of the game, varying from COP$5.000 to COP$400.000 (game currency; it is not real but fake money).
7. Tokens: at least 6 colored tokens representing the players.
8. Initial money amount: each player receives COP$200.000 in game currency.
9. Control and resource cards: 44 cards (4 per control/ resource) including the name and value of the control/ resource. Types of cards are the following (see Exhibit 3): signed contract, available expert, documentation methodology, historical data, license purchase, quality management system, IDE actualization, written certificate, regular backup, motivated employees, and test suite.

GAME RULES

1. All players take hexagonal pieces and fit them in any order on the board stage according to the color.
2. Players should move their token by following lines from corner to corner of a hexagonal piece. Two arrows are the signals of the starting point.

Exhibit 1
Hexagonal pieces


User Manual became useless
Scope Expanding
Scope Expanding
Scope Expanding
Scope Expanding
Incomplete Functional Tests
Incomplete Functional Tests
Incomplete Functional Tests
Incomplete Functional Tests
Poor Technical Documentation
Loss of key human resources
User Manual became useless
Loss of key human resources
Poor Technical Documentation
5,6
3,4
1,2

Risk sheet
Table 1

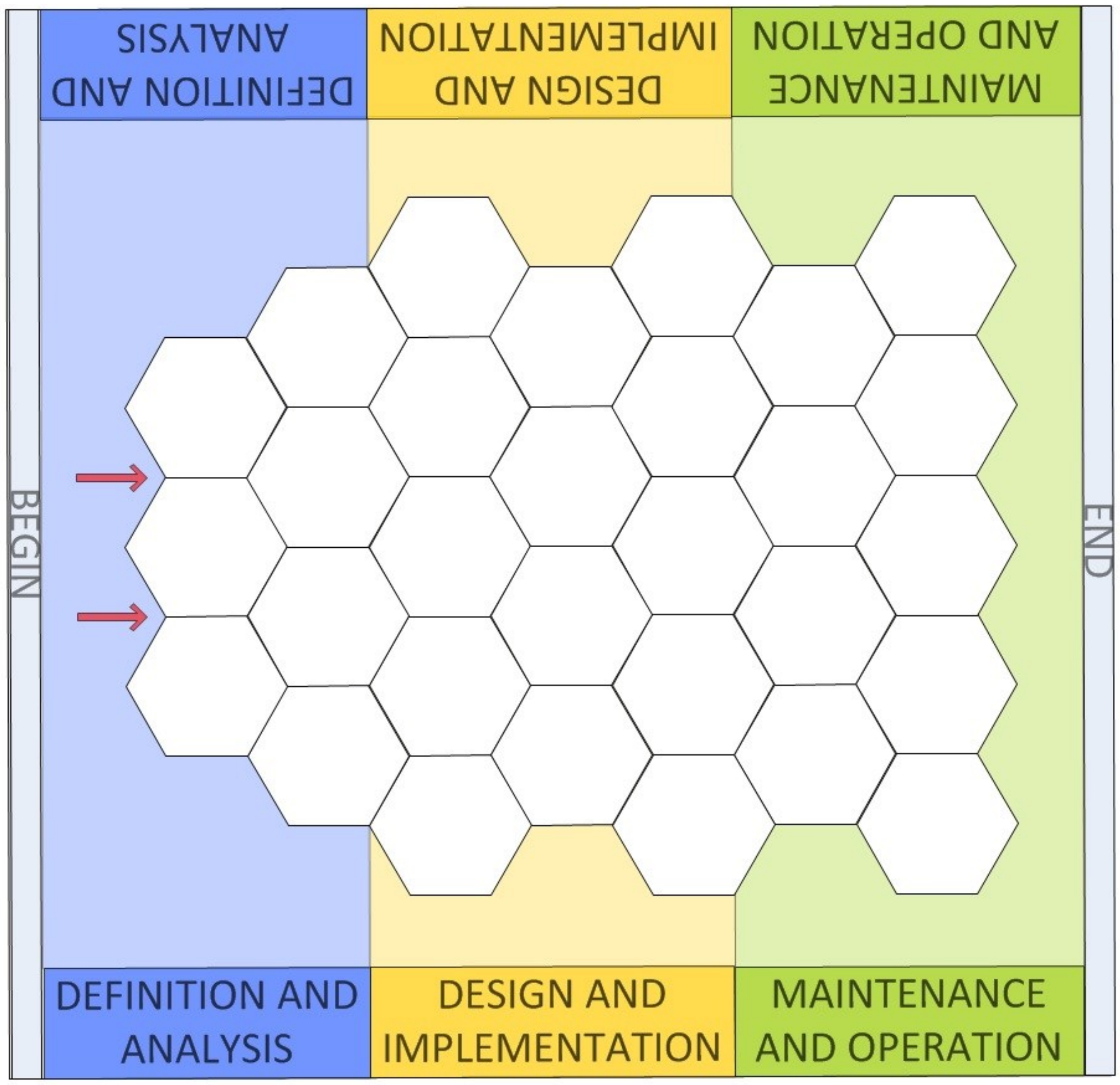
Risk	Impact	Resource/Control	
Incomplete functional tests	**Medium** (lose two turns)	Written certificate **COP$10.000**	Available expert **COP$70.000**

Payment sheet
Table 2

STAGE	1	2	3	4	5	6	7	8	9	10	11	12
PERCENTAGE	1	3	4	5	10	16	25	33	48	61	79	100
PAYMENT	5.000	10.000	15.000	20.000	40.000	65.000	100.000	130.000	190.000	245.000	315.000	400.000

3. Tokens should be moved step-by-step, from one corner to the next one.
4.
5. The board boundaries cannot be included in the path defined by a player.
6. Only forward moves are allowed (from the “begin” signal to the “end” signal).
7. One player turn have two actions: (i) moving the token to the next corner and (ii) throwing the die for defining the risk to be selected. Related to the risk, the corner have three hexagonal pieces numbered with the pairs (1,2), (3,4), and (5,6). The number matching the die reveals the candidate risk to be materialized.
8. At the beginning of the game, each player receives COP$200.000. This game currency can be invested in buying controls/resources—valid during the entire game period—in order to mitigate risks and avoid risk materialization.
9. Once the game rules are explained, in the next 15 minutes all players should define the path to follow in the board and the controls/resources he/she wants to buy.
10. In order to mitigate a risk, players need both a control and a resource. If a player has only one of them, the risk is not mitigated.
11. If a risk is materialized and the player has both the resource and the control for mitigating the risk, the player can continue the game. In other case, the player receives the defined impact of the materialized risk.

Exhibit 2
Path control fold


DEFINITION AND ANALYSIS
DESIGN AND IMPLEMENTATION
MAINTENANCE AND OPERATION
BEGIN
END
DEFINITION AND ANALYSIS
DESIGN AND IMPLEMENTATION
MAINTENANCE AND OPERATION

12. Three kind of impact are defined: high, medium, and low. High impact implies losing two turns and medium impact means losing one turn. A game situation with low impact has choices: (i) player can pay half of the money needed to buy the control and the resource; or (ii) if he/she lacks the money or he/she does not want to pay the money, turn is lost.
13. The end of the game is reached when the first player comes to the "end" signal.
14. By the end of the game, each player receives a payment related to the advance level he/she reaches. Risk and payment sheet has the information belonging to each board stage.

GAME WINNER

By the end of the game, the winner can be determined by counting the game currency available for each player (the amount resulting from the usage of the initial money and the payment for crossing the board).

RESULTS OF THE GAME APPLICATION

Riskware was applied to 18 undergraduate students belonging to the software project management course of the Universidad Nacional de Colombia system engineering program. Results were collected from students by filling in the following poll:

1. How actual is the game compared with a software development project?
2. What fun factor would you assign to the game (1-deficient, 2-acceptable, 3-good, 4-very good, 5-excellent)
3. How simple to play seems to be the game?
4. What do you learn from the game?
5. What do you think is the best strategy for winning the game?
6. What would you change to the game?
7. Would you please mention two risks belonging to any software development stage and the controls/resources needed to mitigate them?

The purpose of this poll is obtaining feedback related to the realism degree, the simplicity, and the fun degree for players. Also, we need to assess the accomplishment of the teaching purpose of the game and the learned lessons (the strategy followed by players in the game and the knowledge obtained about risk management).

The first question is oriented to identify the realism degree perceived by players of the game. Students agree on the fact that simulating software project risks by using riskware is very close to reality, and they argue some reasons shown in Table 3.

Related to the second question, most of the students (53%) assign to riskware a value of very good in the fun factor. 37% of the students believe the game is excellent and 11% of the students tell the game is good related to the

Exhibit 3
Control and resource cards

SIGNED CONTRACT
$50.000
AVAILABLE EXPERT
$80.000
HISTORICAL DATA
$20.000

Table 3
Reasons for justifying riskware realism

No.	Result	Student percentage
1	Riskware presents some issues involved in software development projects, *e.g.,* phases, risks, planning, etc.	37%
2	Riskware presents the possible risks, impact, occurrence probability, and mitigation choices of a software development project	26%
3	Riskware shows the importance of planning for mitigating risks can materialize when executing a software development project. Some issues are highlighted: budget and risk prioritization	26%
4	Riskware simulate randomness as an external factor affecting a software development project	21%

fun factor. This feedback reinforces the motivation as a factor for designing this kind of teaching strategies.

Related to the third question, most of the students (53%) believe riskware is very simple to play because the game rules are clearly stated from the beginning of the game, while 47% of the students tell that the game is moderately simple to play, because the player should completely understand the rules and define a good strategy simultaneously, which is not so simple to achieve.

Table 4 summarizes the answers to the fourth question. Planning, choices, randomness, and budget are identified as the main learning issues of the game. The second result is directly linked to the third riskware goal. Be advised that the second riskware goal—risk identification—is the main topic of the seventh question. The third and fourth results are linked to the actual situations we need to face in actual software development projects.

The strategies the players try to apply for winning the game (fifth question) are shown in Table 5. Results are directly associated with the risk management discipline, since intuitively we can discover some key factors: risk materialization throughout the board (risk occurrence probability), turn losing during the game when risks are not mitigated (risk impact), and focus on the importance of some controls/resources to prioritize the purchase (budget limitations).

Improvement suggested for riskware (the sixth question) is defined as "none" for most of the students. However, some of the students suggest: (i) secret purchase of controls/resources, for avoiding revealing the strategy of the players; (ii) numbering the stages of the game for the players clearly locating the position of each player of the game; and (iii) increasing the amount of initial money to be invested, in order to simulate more actual situations. The first two suggestions are easy to implement, but the third one is not suitable because of the game limitations. Precisely, we need to simulate the lack of resources for promoting good choice of the available resources/controls.

Seventh question—directly related to the second riskware goal—was correctly answered by 32% of the students. 47% of the students mentioned the two risks and some of the resources/controls needed for mitigating the risks, and 95% of the students only mentioned two risks. This question was the most challenging of the poll, because some of the students are so immersed into the strategy for winning the game that they do not focus on the name of the risks and the resources/controls purchased. The game is useful for both giving risk management issues to the students and showing some of the most frequent risks can be materialized in a software development project.

CONCLUSIONS

Experience-based games are positioned as an ideal organizational learning strategy for spreading tacit knowledge among the members of an organization, because while they are applied the players:

1. Are involved in an active way in the teaching-learning process.
2. Gain instant feedback from the actions taken during the game. So, they can assess the consequences of such actions.
3. Actively interact with each other and exchange opinions about the several action courses followed during the game. Hence, they reveal factors to be accounted by experts in risk management when they are making decisions.

Table 4
Learning issues of riskware

No.	Result	Student percentage
1	Good (strategic) planning is important for mitigating risks	74%
2	Impact and occurrence probability are the main issues to choice in mitigating risks	47%
3	Randomness affects software development projects because of the uncertainty about the risks with some probability of materializing	32%
4	Risks cannot be mitigated because of budget limitations	16%

Table 5
Strategies for winning riskware

No.	Result	Student percentage
1	Defining the path with risk occurrence probability in mind	37%
2	Investing more money in resources/controls which help to mitigate most of the risks	32%
3	Identifying the path with most common risks and looking for mitigate the risks with higher impact	26%
4	Configuring the board in such a way that common risks are close	21%

Riskware—a game for software project risk management—gave satisfactory results for teaching risk management to software engineering students. Riskware goals were almost completely achieved and, at the same time, this game exhibited high degrees of fun and simplicity for the players. Riskware was designed by following a systematic approach to design experience-based games, based on well-known games acting as templates for the new ones. This kind of design keep the features that make successful the original games while introduce new features for teaching purposes. Consequently, we can balance playfulness and teaching.

Finally, riskware is highly flexible to be applied in other knowledge areas like plant layout, piping design, and investment project management. We can keep most of the rules and we need only modifying the risks, resources, and controls. In this way, we can demonstrate the importance of the proactive risk management in several domains.

FUTURE WORK

Some issues of future work can be proposed from the design and application of riskware:

1. Incorporating the proposed improvement in a new version of the game.
2. Evolving the main structure of this game to a videogame, in order to be applied outside the classroom. This strategy can also make this game shareable among several players world-wide.
3. Applying riskware in graduated software engineering courses for reinforcing some concepts of risk management in actual software engineering practitioners.
4. Using riskware for teaching risk management in other domains, by using the same structure of the game but changing the context in which other risks are materialized and mitigated.

ACKNOWLEDGMENT

The authors want to acknowledge the help of Eng. Sebastián Sogamoso in the early stages of this project. We hope Sebastián is now practicing all of the brilliant ideas we discuss when planning and elaborating this project.

REFERENCES

American Systems Corporation. (2012). Retrieved from: http://www.americansystems.com/Pages/Home.aspx

Baker, A., Navarro, E. & Van Der Hoen .(2005). An experimental card game for teaching software engineering processes. *The Journal of Systems and Software*, (75), 3–16.

Burgos D., Tattersall C. & Koper R. (2006). Can IMS Learning be used to Model Computer-based Educational Games?. *Binaria Magazine*

Dempsey J., Rasmussen K. & Lucassen, B. (1996). The Instructional Gaming Literature: Implications and 99 Sources. *COE Technical Report*, 96(1), College of Education. University of South Alabama, USA.

Gasca, G.P. (2010). *Metodología de Gestión de Riesgos para la adquisición de software en pequeños entornos – MEGRIAD*. Ph.D. dissertation, Facultad de Informática Universidad Politécnica de Madrid.

Kasvi J. (2000). Not Just Fun and Games: Internet Games as Training Medium. *Cosiga—Learning with Computing Simulation*, Helsinki University of Technology.

Klassen K. & Willoughby K. (2003). In-Class Simulation Games: Assessing Student Learning. *Journal of Information Technology Education*, 2, 1–13.

Kober R. & Tarca A. (2000). For fun or profit? An evaluation of a business simulation game. *Accounting Research Journal*, 15, 98–111.

Lainema T. Redesigning the Traditional Business Gaming Process—Aiming to Capture Business Process Authenticity. *Journal of Information Technology Education*, 3, 35–52.

McConnell S. (1997). *Desarrollo y Gestión de proyectos informáticos*. Madrid: Mc Graw Hill.

Presmann, R. (2005). *Software engineering: a practitioner's approach*. New York: Mc Graw Hill.

Project Management Institute. A Guide to the Project Management Body of Knowledge, USA.Págs. 127-129, 2000

Sommerville, I. (2005). *Software Engineering*. Reading: Addison-Wesley.

Taran, G. (2007). Using Games in Software Engineering Education to Teach Risk Management. *Proceedings of the 20th Conference on Software Engineering Education & Training.* Dubin, Ireland.

Zapata C., & Awad G. (2009). Requirements Game: Teaching Software Projects Management. *CLEI Electronic Journal*, 10(1). Retrieved from: http://www.clei.cl/cleiej/paper.php?id=133