CLEI ELECTRONIC JOURNAL, VOLUME 10, NUMBER 1, PAPER 3, JUNE 2007

Requirements Game: Teaching Software Project Management

CARLOS M. ZAPATA J.
Universidad Nacional de Colombia, Facultad de Minas, Escuela de Sistemas,
Grupo de Investigación UN-INFO.
Medellín, Colombia
cmzapata@unalmed.edu.co

and

GABRIEL AWAD-AUBAD
Universidad Nacional de Colombia, Facultad de Minas,
Escuela de Ingeniería de la Organización.
Medellín, Colombia
gawad@unalmed.edu.co

Abstract:

Several business areas, like Management and Negotiation, have used games like a didactic way to simulate world reality, to introduce students to the day-to-day generated problems, and to extract from the application of games, concrete learning for theoretical knowledge securing. In this paper we present "Requirements Game", a practical way to simulate software development conditions in a competitive environment similar to real life. This game has been played by several groups of students from Facultad de Minas, Universidad Nacional de Colombia, and this paper summarizes the results from this experience.

Keywords: Teaching and Learning Support Environments, Teaching Methodologies, Software Project Management, Learning by games.

1 INTRODUCTION

Brooks [3] outlines the software project difficulties and the software development obstacles that caused the so-called "Software Crisis". Software Engineering was created as a response to these difficulties and obstacles. Software Engineering has established a set of guidelines for software projects; it is necessary that this kind of projects achieve their proposed goals, accomplish limit dates and budgets, and guarantee the adequate use of resources.

In order for these goals to be achieved, Software Engineers must be able to manage projects, given the fact that management is currently a lesser-important issue in Software Engineering curriculum. It is important to highlight that one of the most important challenges in current teaching is the easing of the transition from the classroom to the industries [14]. However, Software Engineering teaching has focused on metrics and development methods conceptualization, and little importance has been assigned to management aspects; these aspects are important for developing abilities for the Software Engineer in order to successfully manage software projects.

Modern trends in Software Engineering education propose that Software Engineers must acquire knowledge, and develop personal control and analytical abilities [10]. For this reason, pedagogical processes in Software Engineering should focus not only on this field of knowledge, but on conceptual elements belonging to other knowledgeable fields, specially those related to the development of social abilities (team work, leadership, communication, and so forth).

Furthermore, learning models have evolved from teacher-centered models to student-centered models. The responsibility for spreading the concepts is assigned to the teacher in teacher-centered models; as a contrast, in student-centered models, the student must develop concepts with the support of the teacher [8, 10]. Student-centered models have more chances to face the mentioned challenge of current teaching, and the reason is that effective learning—and we can assume that learning is the process of generating a permanent change in knowledge or behavior by means of experience—involves logical issues (analytical or verbal processes) and creative activities development (for example patterns and relationships), and this kind of learning is better managed by student-centered models.

Some of the proposed educational strategies for reaching effective learning are: case study method, lectures, guided practices, technical reviews, gaming, and so on. Traditional teacher-centered methods are currently complemented with student-centered strategies like case study method and gaming; this combination gives the student responsibility for self-learning.

Collaborative learning gives theoretical support to gaming—teaching by means of games—and it is based on knowledge-generation pedagogical processes instead of knowledge-transmission pedagogical processes of traditional education. Collaborative learning confronts a student with a problem to be solved (by gathering, analyzing, and discussing in groups); the student gains knowledge, social skills, and personal control abilities [10]. In Collaborative learning, the teacher is an active facilitator, instead of an information repository [15]. Collaborative learning has benefits like [10]:

- Improvement of cognition.
- Generation of solving problem abilities.
- Development of team work abilities.
- Improvement of verbal and social skills.
- Self-esteem reinforcement.

Sciences like Management or Negotiation have regularly used games, to generate management and control abilities in their students. "Beer game" [16], microworlds [16] and "Negotiation game" [5], use playful activities for developing management abilities to students and professionals around the world. However, Software Engineering has used traditional teaching for developing these kinds of abilities, and collaborative learning is still unusual in it. As a matter of fact, a review of 99 papers on gaming for learning, made in 1996 reported not a single experience of this kind in Software Engineering [6]. Recently, some work has been done about gaming in Software Engineering; the game "Problems and Programmers" is a good example of this situation. This game is a card game and it is focused on reviewing the impact of requirements analysis decisions on code development [1].

We present in this paper "Requirements game", a practical and playful way to foster capabilities development concerning management in Software Engineers. This paper is organized as follows: in Section 2 we discuss the role of games in teaching; "Requirements game" is presented in Section 3; in Section 4 we discuss the results of applying this game to various groups of students; in Section 5, we discuss conclusions and future work.

2 THE ROLE OF GAMES IN LEARNING

Gaming has replaced case study method as a pedagogical tool [12], due to the fact that individual behavior differs from group behavior [14], and games stimulate emotional components of people. Gaming is considered a modern pedagogical tool, but its origins can be traced to the mammal "education" of brood by means of games [9].

Some definitions of games, from the pedagogical point of view, are:

- ➢ According to Crawford, cited by Kasvi, "a game is a closed formal system that subjectively represents a subset of reality". Every time a game is played, a new version of the history is completed. The final result of the game depends on the global conditions of the game, the features of the participants, and the relationships among them [9].

- Dempsey, Rasmussen, and Luchasen establish that a game is a learning rule-guided format, and it fosters competition against competitors, machines, or previously stated standards [6].

- Bushell defines a game as an interactive activity for simulating real-world conditions in order to stimulate decision making learning [4]. Games are competitions where people agree with certain rules of behavior, and where people make decisions for changing self and each other states [12].

Some advantages of games in learning, from the point of view of the previous definitions, are:

- According to Heyman, cited by Kober and Tarca, games promote "learning-making" process, increase motivation, and develop communication among people. Additionally, games stimulate peer learning and take into account the impact of emotions in learning [12]. Furthermore, in interdisciplinary activities, learning by means of games develop elements like critical thinking, group communication, and decision making [15]; these elements are difficult to obtain in an isolated manner, and from a theoretical point of view.

- Games are effective learning activities, and the reason is that they increase learning velocity, improve knowledge retention, and foster concept reminding [11]. Crawford, cited by Kasvi, states that games are opportunities to learn concepts in a safe environment, where possible mistakes have no dangerous effects on participant's life [9]. Furthermore, team work fosters agreement among participants about tasks, goals, and methods [13].

- Many authors consider experience as an excellent teacher. In this context, simulation games promote learning in absence of real-life experience [6]. Besides, McKeachie, cited by Klassen and Willougbhy, emphasizes the hardening of learning by means of decision making process and its consequences [11].

- According to Bushell, games have other advantages: knowledge of the real world, understanding of team building and personnel behavior, and motivation improvement by means of competence, as well as acknowledgment of the importance of information in the decision making process, risk assessment, and time management [4]. Also, games are important when assigning value to the abilities required by people in organizations, and searching for the combination of highly differenced skilled-people [4].

Bushell [4] mentions the following drawbacks for games:

- Games must be simple. A Game can only reproduce a limited amount of business components.
- Feedback from decision making process is faster in games than in real world. This fact implies that most of the time participants are making decisions without receiving any feedback.
- Time constraints the decision about the type (operative or strategic) of the game.

Games clearly increase motivation and student concerns about a theme, but, according to Pivec et al., it is not clear if they create meaningful changes in cognitive learning strategies [15]. A big amount of game studies have been made in children and young people, but there are few references to games efficiency in adult people [7].

We describe, in the next section, "Requirements game", a way to diversify teaching in project management skills in Software Engineering.

3 REQUIREMENTS GAME

Requirements game is a one-and-half-hour to two-hour simulation of the software development process. Participants are in charge of the development of a software application, with special requirements previously established by the customer. This game reproduces some of the real situations of method-oriented software development projects.

3.1. Game Preliminaries:

Requirements game need the following resources (the amount of every resource depends on the number of participants):

- Pencils
- Pens
- Blank folds with “sketch fold” label
- Blank folds with “final documentation fold” label
- Specification folds with the minimum conditions of the game
- Clock
- Two computers with a software development application (for example MS-Access® or similar)
- Templates for the record of the teams

At the beginning, participants must be divided into 4- or 5-players teams. Every team needs one Director and a set of operative members (analysts, designers, and programmers). Coordinator of the game publicly states the rules, and then he/she sets a time period for raw material buying and computer time planning process. He/she also represents the University of the case study (the customer and the raw material supplier). In the game, the role of assessor is played by participants with previous experience in the game (or capacitated in this role by the Coordinator). Following are the functions of the actors in Requirements Game:

- Representative of the University: Explains the game and gives answers to the assessor doubts.
- Assessor: Only answers the Director doubts, and receive and assess documentation and software applications.
- Director: Coordinates the team work, delivers products to the assessor, and communicates with the assessor. The Director can not execute operative work (documentation making or software development).
- Team member: Elaborates documentation and develops software applications. Team members can only communicate with each other and with their own Director.

Coordinator of the game must highlight to the participants the awards for developing documentation and software applications on time, and the fines for not doing so. 1-cycle or 2-cycle Requirements game is selected by means of the available time for playing. In the case of 2-cycle Requirements game, after the first cycle the Coordinator of the game compiles the results for every team, and then he/she exchanges the developed software applications (for example, he/she delivers software produced by “B” team to “C” team, and *viceversa*) in the beginning of the second cycle. Coordinator of the game must have a previously developed software application, in the given case that none of the groups can complete the software development in the game time period.

3.2. Specification fold for the first cycle of the game:

“A” University requires the development of a software application for calculating grade notes of a course. Then, “A” University has invited “B” and “C” software development companies. Every company has a project Director and a variable amount of analysts, designers, and programmers.

“A” University has stated some of the software requirements in the following functions: register students, register student’s grade marks (there are four grade marks with 10%, 20%, 30%, and 40% respectively), and consult in the screen the summarized student’s grade marks.

Documentation of the software application must be integrated by: verbal model of the proposed solution, entity-relationship diagram, relational model, sketches of a minimum of three graphical user interfaces, and an estimation of the accomplishment of cause-and-effect diagram of Figure 1, related to the proposed solution.

Parameters for the documentation assessment are: Completeness, Consistency, Correctness, Aesthetics, and Clarity[1].

The University representative will act as a customer and a supplier. In the role of supplier, he/she will provide some elements to software development companies, like pencils, pens, sketch folds, and final documentation folds. Work teams can discuss within limited time what raw materials to acquire, and how much computer time to contract. After this time, no more raw materials can be acquired. Remaining raw material can not be used in the other cycle.

Documentation must be manually made, and computer time must be only used in software application programming.

Raw materials, computer time, and team member work time, are valued with the same currency. For example, Table 1 shows these values with a generic currency. With the same currency are defined the values that “A” University will pay for the software development. Table 2 shows these values.

3.3. Specifications fold for the second cycle of the game

“A” University needs to upgrade the software application for calculating grade marks of a course. Then, “A” University has invited “B” and “C” software development companies. Every company has a project Director and a variable amount of analysts, designers, and programmers.

“A” University has stated the following requirements (functionality can be seen as an unstated requirement of the software application): register student’s grade marks in different semesters (students can repeat the same course); semesters must be registered in YYYYSS format, where YYYY is the year and SS is the 2-digit number of the semester. The software application must register the name of the professor in every semester. The software application must build a report, activated from an existent interface, for reviewing the grade mark average in a requested semester.

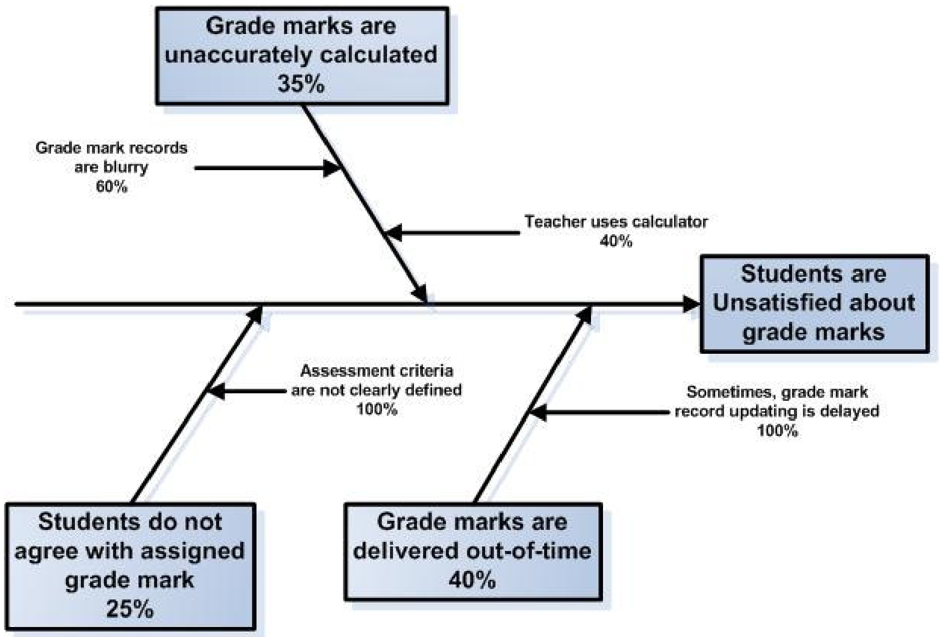

Grade marks are unaccurately calculated 35%
Grade mark records are blurry 60%
Teacher uses calculator 40%
Students are Unsatisfied about grade marks
Assessment criteria are not clearly defined 100%
Sometimes, grade mark record updating is delayed 100%
Students do not agree with assigned grade mark 25%
Grade marks are delivered out-of-time 40%

Figure 1: Cause-and-effect diagram of the grade mark software application.

[1] The assessment of Completeness, Correctness, and Consistency is reached by means of the definitions provided by Zowghi and Gervasi [18]. The assessor must interpret these definitions. Aesthetics and Clarity of software documentation will be evaluated by the assessor with his/her own criterion.

Table 1: Costs of raw materials and computer time.

ITEM	VALOR
Specification fold (requirements and c-e diagram)	2.000
Sketch folds	100
Final documentation folds	300
Pens	200
Pencils	400
Director Salary	8.000
Work team member salary	4.000
Computer time minute	1.000
Additional computer time minute	2.000

Table 2: Values to be paid by “A” University for software application.

ITEM	VALOR
New Interface	15.000
Modified interface	10.000
Documentation item qualified by the assessor	10.000
Award per minute in advance	3.000
Fines per minute setbacks	3.000
Fines for functionality mistakes	5.000
Fines for calculus mistakes	1.000
Fines for mismatches against requirements	5.000

Documentation of the software application must be integrated by: current and modified entity-relationship diagrams, the relational model, and sketches of the proposed modification to be implemented upon graphical user interfaces. Parameters for the documentation assessment are: Completeness, Consistency, Correctness, Aesthetics, and Clarity.

Table 1 summarizes cost of raw materials and computer time, and Table 2 contains values to be paid for the software application.

3.4. Assigned time per cycle:

Table 3 suggests minute-time periods for every cycle of 1-cycle or 2-cycle Requirements game. These periods are subject to availability of time for playing the game, but it is recommended the use of the periods stated in Table 3 as a minimum, in order to give participants enough time for developing the entire software application.

Next section summarizes some of the results of game application.

4 SOME RESULTS OBTAINED FROM SEVERAL GROUPS OF STUDENTS PLAYING THE GAME

Requirements game was played by five groups of students in the Facultad de Minas, Universidad Nacional de Colombia. Table 4 shows some information about these groups. Table 5 shows some of the insights of the experience of Requirements game in these groups.

With the purpose of exhibiting formative features acquired by participant students in the Requirements game, we use the 7-S model. This model comprises the following seven aspects: Structure, Style, Systems, Strategy, Shared Values, Staff, and Skills [2].

Table 3: Minute-time periods recommended for playing Requirements game.

Description	**2-cycle**	**1-cycle**
Explanation of the game	5	5
Raw materials buying	3	3
Raw materials delivering	2	2
Planning time period	5	5
Playing time period	45	45
Extended time period	10	10
Documentation and software application review	5	5
Raw materials buying	5	0
Planning time period	5	0
Playing time period	20	0
Winner selection	5	5
Discussion	10	10
GRAND TOTAL	120	90

A summary of decision making process and experiences of every team are:

- Structure. Every team must establish organizational procedures. How many people were required in every organizational function? Was task specialization convenient for executing tasks? Was an organizational hierarchy needed? How the information must have flowed? Were functions clearly defined?
- Style. It reflects the behavior patterns inside the organization. What kind of organization was used? What was the preferred behavior of the leaders (leadership or authority)? How was the organizational atmosphere?
- Systems. Formal processes and procedures for organizational management. What was the kind of planning? How was the organization of task execution? Were processes established?
- Strategy. What was the goal of the team? How was the plan for achieving this goal? What factor had higher importance: accuracy in deliveries, quality of the final product, or financial impact of decision making process?
- Shared values. What values were identified in the organization? Did the player behavior reflect their own ethics?
- People. If you could select people for your organization, what would the criterion be? Did the companies make good use of people abilities? Were synergies possible?
- Skills. Did the companies identify skills, competence and expertise of their employees?

Table 4. Some information about the participant groups in Requirements game.

Item	1st group	2nd group	3rd group	4th group	5th group
Profile	Systems Engineering undergraduate students	Systems Engineering undergraduate students	Systems Engineering Master students	Systems Engineering undergraduate students	Systems, Industrial, and Administrative Engineering undergraduate students
Semester	7th	5th	1st	5th	8th
Number of participants	12	15	8	20	30
Did some team generate profits to the companies?	No	No	Yes	No	No

Table 5. Insights from the experience of Requirements game

Item	1st group	2nd group	3rd group	4th group	5th group
Were there assessors?	No	No	No	Yes	Yes
Game type	2-cycle	1-cycle	2-cycle	2-cycle	2-cycle

Furthermore, Requirements game make students aware of several organizational aspects, like:

- The impact of the decision making process on the financial results of the organization.
- The importance of the adequate use of raw materials and computer time period.
- The importance of planning and task-specialization on the development of a high-quality product.
- The respect for the requirements stated by the customer.
- The importance of good documentation for software application updating purposes.
- The value of an on-time delivering, and the disadvantages of out-of-time delivering.

For the sake of exemplifying, Figure 2, 3, and 4 presents graphical user interfaces developed by the winners of the second described group. This group had a single design strategy, and they were able to make the software application within the play time period.

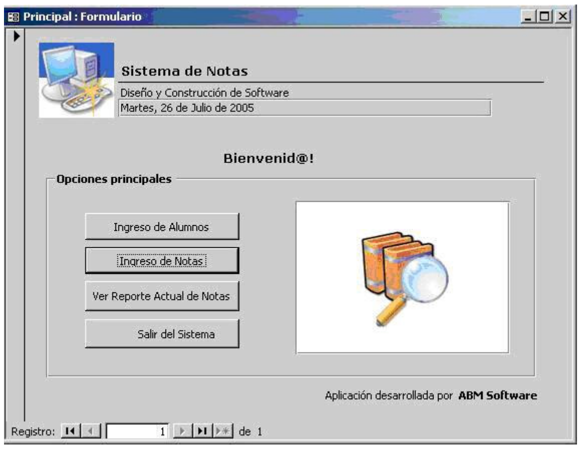
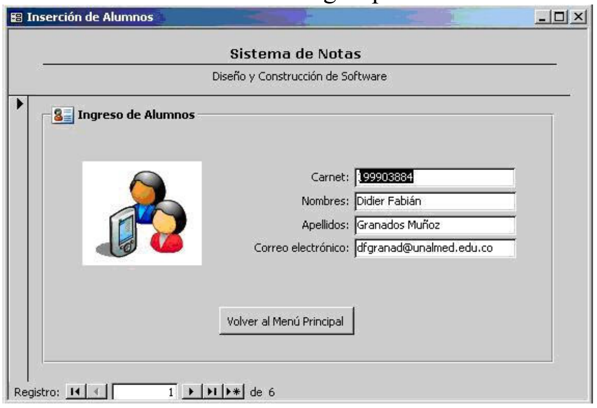
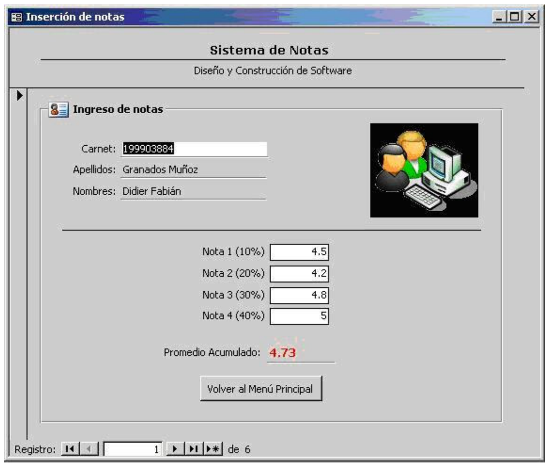


Figure 2. Snapshot of the main menu corresponding to the software application developed by the winners of the second group.



Carnet: 199903884
Nombres: Didier Fabián
Apellidos: Granados Muñoz
Correo electrónico: dfgranad@unalmed.edu.co

Figure 3. Snapshot of the graphical user interface of Student registering of the software application developed by the winners of the second group.



Carnet: 199903884
Apellidos: Granados Muñoz
Nombres: Didier Fabián

Nota 1 (10%)	4.5
Nota 2 (20%)	4.2
Nota 3 (30%)	4.8
Nota 4 (40%)	5

Promedio Acumulado: 4.73

Registro: 1 de 6

Figure 4. Snapshot of the graphical user interface of Grade mark registering of the software application developed by the winners of the second group.

Table 6 shows the assessment values assigned to the winners of the second group of 1-cycle Requirements game. We can see in this Table that, despite the single design for making the software application, this group could not generate profits for the company. The reasons for this problem were:

- They used additional development time; as a consequence, costs were increased and fines were applied.
- They were exhaustively devoted to the making of software application, and, consequently, they overlooked the software documentation. The obtained value was, as a result, very poor. Documentation is important for future support and maintenance of the software application.
- They begin late to make the software application. Then, the customer requirements were not completely reflected in the software application.

In the next section, we discuss main conclusions and future work derived from this game.

5 CONCLUSIONS AND FUTURE WORK

The main conclusions of this work can be expressed as:

- Traditional Software Engineering teaching has employed few strategies for fostering Project management skills in the students. This teaching has emphasized on methodological and development aspects of engineering.
- Learning by games as a pedagogical strategy has been subemployed in Software Engineering. However, there are currently some works on it.
- The importance and positive impact of learning by games is beginning to be recognized. Learning by games is being considered as a complement for traditional teaching.
- Requirements game is a pedagogical strategy for simulating problems that Software Engineering students will experience in real life.

Some future work that can be generated from this experience is:

- Playing Requirements game with analysts, designers, and programmers from real companies, in order to gain feedback from the results of the game. It is also the purpose to compare results of both types of participants.
- Development of other games, which could be considered complementary for Requirements game. Some of the issues related to Software Engineering, in order to develop new games, could be: Consistency, Modeling, Viewpoint management, Software life cycle, and so on.

Company number	3	ABM

SOFTWARE APPLICATION MAKING

Description	Quantity	Unit value	Total value
Specifications (Requirements and cause-and-effect diagram)	1	2.000	2.000
Sketch folds	4	100	400
Final Documentation folds	3	300	900
Pens	3	200	600
Pencils	1	400	400
Director salary	1	8.000	8.000
Work team member salary	4	4.000	16.000
Computer time minute	13	1.000	13.000
Additional computer time minute	10	2.000	20.000
		Total Costs	61.300
New Interface	3	15.000	45.000
Modified interface	0	10.000	0
Documentation item qualified by the assessor	0.6	10.000	6.000
Award per minute setbacks	0	3.000	0
Fines per minute backwards	10	3.000	30.000
Fines for functionality mistakes	0	5.000	0
Fines for calculus mistakes	0	1.000	0
Fines for mismatches against requirements	0.2	5.000	1.000
		Total Income	20.000
		Net profit or loss	-41.300

Table 6. Assessment of the winners of the second group of 1-cycle Requirements game.

REFERENCES

[1] Baker, A., Navarro, E., and Van der Hoek, A. An experimental card game for teaching software engineering processes. *The Journal of Systems and Software*, No. 75, (2005), pp. 3-16.

[2] Bradach, J. Organizational Alignment: The 7-S model. *Harvard Business Review,* (Noviembre 1996), 11 p.

[3] Brooks, F. P. No Silver bullet. Essence and accidents of software engineering. *IEEE Computer*, Vol. 20, No. 4, (1997), pp. 10-19.

[4] Bushell, T. The role of the business game in management education. *Proceedings of the Conference Reflections On Teaching: Maintaining Quality in Changing Times,Low Wood, Lake Windermer* (Abril 2001), 11 p.

[5] Christopher, E. and Smith, L. *Negotiation Training through Gaming: strategies, tactics and manoeuvres*. Kogan Page, (1991), 189 p.

[6] Dempsey, J. V., Rasmussen, K. and Lucassen, B. *The Instructional Gaming Literature: Implications and 99 Sources.* COE Technical Report No. 96-1. College of Education, University of South Alabama, (1996). Available on the Internet: http://www.coe.usouthal.edu/TechReports/TR96_1.PDF (Consulted on 20/05/2005)

[7] Gander, S. Does Learning Occur through Gaming?. *Electronic Journal of Instructional Science and Technology,* Vol. 3, No. 2, (Marzo 2000), pp. 28-43.

[8] Greer, L., Robinson, T., and Sweetman, T. Approaches to learning: An international comparison of higher education in the former socialist states of central and eastern Europe. 2002. *Proceedings of the Conference Reflections On Teaching:Supporting the Teacher, Challenging the Learner,* 2002.

[9] Kasvi, J. Not Just Fun and Games: Internet Games as a Training Medium. *Cosiga - Learning With Computerised Simulation Games*, P. Kymäläinen & L.Seppänen (Eds.), Skidoo: Helsinki University of Technology, (2000), pp. 22-33.

[10] Kimber, D.. Collaborative Learning in Management Education: Issues, benefits, problems and solutions: A literature review. *Proceedings of the ANZAM (Australian and New Zealand Academy of Management) conference in New Zealand*, (1996).

[11] Klassen, K. and Willoughby, K. In-Class Simulation Games: Assessing Student Learning. *Journal of Information Technology Education*, Vol. 2, (2003), pp. 1-13.

[12] Kober, R., and Tarca, A. For fun or profit? An evaluation of a business simulation game. *Accounting Research Journal*, Vol. 15, 2000, pp. 98-111.

[13] Lainema, T. Redesigning the Traditional Business Gaming Process—Aiming to Capture Business Process Authenticity. *Journal of Information Technology Education*, Vol. 3, (2004), pp. 35-52.

[14] Neville, K. and Adam, F. Integrating Theory and Practice in Education with Business Games. *Informing Science*, Volume 6, (2003), pp. 61-73.

[15] Pivec, M., Dziabenko, O., and Schinnerl, I. Aspects of game-based learning. *I-KNOW 03, the Third International Conference on Knowledge Management*, Graz, Austria, 2-4 July, 2003, 10 p.

[16] Senge, P. *The Fifth Discipline: The art and practice of the learning organization.* New York, Doubleday. 1990.

[17] Woolfolk, A. *Educational Psychology.* Needham Heights, MA: Allyn and Bacon. 1993.

[18] Zowghi, D. and Gervasi, V. The Three Cs of Requirements: Consistency, Completeness and Correctness. P*roceedings of 8th International Workshop on Requirements Engineering: Foundation for Software Quality*, (REFSQ'02), Essen, Germany, (September 2002). 10 p.