Learning Risk Management in Software Projects with a Serious Game Based on Intelligent Agents and Fuzzy Systems

C. D. C. de Oliveira[1] M. E. Cintra[2] F. M. Mendes Neto[2]

[1]Federal Institute of Education, Science, and Technology - IFRN – Ipanguacu, RN, Brazil
[2]Federal Rural University of the Semi-Arid (UFERSA)
Dept. of Exact and Natural Sciences – Mossoro, RN, Brazil
e-mail: danilo.camara@ifrn.edu.br, {cintra,miltonmendes}@ufersa.edu.br

Abstract

Traditional approaches for facilitating the learning process in risk analysis and management in software projects usually lack practical activities in solving real problems. This is basically due to time restrictions and the difficulty of simulating real scenarios where such risks occur. In this sense, games have been used as a learning resource to simulate real scenarios and allow users to learn and practice with no real risks or losses. In this paper, we present an on-going project to develop a multi-player, persistent, Web-based game. This game, named eRiskGame, aims to improve the learning process in risk management in software projects based on intelligent agents and fuzzy systems. The intelligent agents control the actions of the game based on fuzzy systems that store expert knowledge in fuzzy rule bases. Users play the role of software project managers who have to compete for market resources by presenting competitive budgets and software management plans. Users also have to deal with real-world problems involving risks, such as managing employees, budgets, and changes in project plans. The game is being validated in two undergraduate software engineering courses. This paper introduces the techniques used in the project and details eRiskGame.

Keywords: serious games, risk management in software projects, intelligent agents, fuzzy systems.

1. Introduction

Knowledge and skills in risk analysis and management is an important requirement for software project managers. Traditional software engineering courses teach risk in software projects by presenting theory and discussing real world problems. Due to time restrictions and the difficulty in simulating real scenarios, such traditional courses are not able to offer practical training in risk management. In fact, traditional approaches usually adapt and simplify problems, thus, reducing their relevance. Problems are also usually linked to prefabricated solutions that do not help them to develop their own ideas to tackle problems. As a consequence, although the main role of a project manager is to lead projects towards a successful conclusion by solving eventual problems, software engineering students do not develop this ability adequately. In practice, a large number of projects fails due to poor management since managers lack certain abilities and experience in dealing with risks [1].

Game-based learning is an integration of games into the learning process to motivate students to learn while increasing learning effectiveness through immersion with the material. This way, students are encouraged to make mistakes and learn from them [2]. In fact, games, as a simulation strategy, are considered a deep learning activity [3]. It is possible to find several serious games approaches to improve learning in computer science in the literature. A serious game can be defined as a game designed for a primary purpose other than pure entertainment. It is possible to find proposals of games that simulate planing and control of software quality and software development [4, 5, 6], as well as software test [7]. Nevertheless, specifically for the learning of risk management in software projects there is a lack of proposals.

In this paper, we describe an on-going project to develop a multi-player game, named eRiskGame, that focus on learning risk management in software projects based on Intelligent Agents (IAs) and fuzzy logic. IAs are used to provide eRiskGame with intelligent actions, based on expert experience. Such decisions are controlled by rule-based fuzzy systems. A system can be considered a fuzzy system if at least one of its inputs is defined by fuzzy sets, according to the fuzzy set and fuzzy logic theories [8, 9]. Fuzzy systems are able to tackle imprecise and uncertain knowledge, modelling the attributes of a problem using fuzzy sets. Such fuzzy sets receive linguistic values, related to the real problem, improving the interpretability of the system.

eRiskGame simulates a real scenario in which each player is a project manager of a software company. The players are required to compete for resources of software projects in a simulated market by presenting competitive budgets and management plans for specific projects. Players also have to hire

and fire employees and manage situations such as sick leaves, project plan changes, budget problems, among others.

This paper is organized as follows. Section 2 presents the theoretical referential of the topics related to our proposal. Section 3 presents a bibliographical review on games to improve learning in software engineering. Section 4 presents eRiskGame, followed by the conclusions and future work in Section 5.

2. Theoretical referential

Our proposal to learn risk management of software projects involves the use of project-based learning, persistent browser-based games, software project management, intelligent agents, and fuzzy systems. Next, we discuss the basic concepts related to each of these topics.

2.1. Project-based learning

Project-Based Learning (PBL) [10] proposes teaching so that individual and collective competences can be built by developing learning capabilities that allow short-term reasoning. Such learning theory allows the creation and sharing of knowledge beyond the individual and the team scopes. PBL assumes that, in a project centred in short-term tasks, participants must keep a balance between action and reflection in order to learn [11].

As previously stated, the learning process of software engineering is usually accomplished considering traditional approaches, based on a reading model. Such model does not allow the interaction of students with real problems and, thus, with the consequences of actions related to managing software projects. PBL is a technique for improving that allows real world situations to be modelled in a fictitious project. The students engage in the project until its completion, thus, acquiring practical skill and theoretical knowledge.

2.2. Serious games and persistent browser-based games

As previously stated, a serious game is a game designed for a purpose other than pure entertainment. Serious games are simulations of real-world events or processes designed for the purpose of solving problems. Although serious games can be entertaining, their main purpose is to train and educate users. In the context of this work, we use the word *game* referring to *serious games*.

Persistent Browser-Based Games (PBBGs) can be defined as electronic games accessed and played over the Internet using an Internet browser. The persistence of the game is related to the fact that the game is able to progress with successive playing sessions [12]. Games are widely accepted and considered a more pleasant way to learn than traditional approaches [13]. Considering the diversity of public whose time and computational environments are more and more specific, the characteristics of PBBGs are a differential in the success or failure of a learning tool since they allow the interaction, competition, and experience sharing regardless of time and computational environments. Moreover, PBBGs with simple graphics can be accessed in low processing power machines, such as mobile devices.

2.3. Software project risk management

One of the main tasks of a software project manager is to analyse and manage risks. The management process consists of trying to foresee risks that might affect the software delivery or the software quality in order to take actions to avoid or attenuate the impact of such risks [14, 15]. In this sense, a risk can be understood as an event with undesirable and negative consequences [16]. Regarding software projects, risks are usually divided into the following three categories [11]:

1. Project risks – affecting the project management plan or the resources of the project;
2. Product risks – affecting the quality or performance of the software;
3. Business risks – affecting the organization that develops or acquires a software.

Once risks are categorized, it is possible to identify their consequences in projects and plan strategies to tackle such risks. The steps to avoid project risks include trying to foresee them, understand their impact, and take suitable actions, as follows [17].

1. Risk identification;
2. Risk analysis – the probability and consequences of such risk are assessed;
3. Risk planning – plans are elaborated to tackle risks, either by avoiding them or by minimizing their effects on projects;
4. Risk monitoring – risks are constantly assessed while plans to mitigate them are re-evaluated whenever new information about them are available.

The impact and probability of each risk must be analysed so the most appropriate strategies can be defined and followed in order to control and manage such risks. Nevertheless, it is important to notice that some risks cannot be avoided, thus, it is necessary to have a contingency plan.

2.4. Intelligent agents

Intelligent Agents (IAs) can be defined as autonomous entities that observe, by means of sensors, and act upon an environment using actuators [18]. IAs direct the environment aiming at

achieving goals. IAs may also learn or use domain knowledge to achieve goals. IAs have been applied to several areas, including computer games [19], due to their goal-driven behaviour, reactivity, reasoning, adaptability, learning, communication and cooperation. A set of autonomous agents collaborating among themselves to solve a problem whose solution is beyond the capacity of a single agent is considered a multi-agent system [20].

Regarding games, IAs can be applied to several purposes, including the representation of partners and opponents, as well as the representation of the environment itself in order to emulate the real scenario as authentic as possible. In this work, IAs are used in conjunction with fuzzy systems to provide suitable decisions concerning players actions.

2.5. Rule-based fuzzy systems

Due to their high comprehensibility and capability to represent strategies and to capture knowledge from experts, rules are often used for classification and regression tasks. In this sense, fuzzy rules not only take advantage of the general characteristics of rules, but can also represent imprecision and uncertainty. Fuzzy rules usually have the **IF-THEN** format and establish a match between the variables, also called attributes, of their antecedent and consequent parts. Both the antecedent and consequent part of the rules are formed by atomic propositions or a conjunction or disjunction of them. An atomic proposition specifies a linguistic variable V as a linguistic term A, which is a fuzzy set defined according to the fuzzy set theory [9].

Classification is an important task of machine learning that consists of analysing objects, described by a vector of attributes, and assigning them to classes (or categories) that are determined according to the values of their attributes. Fuzzy classification systems are rule-based fuzzy systems formed by a knowledge base, which contains a fuzzy data base, a fuzzy rule base, and an inference mechanism. The fuzzy data base stores the definitions of the variables in terms of fuzzy sets. The fuzzy rule base contains a set of fuzzy rules representing the knowledge of the domain. The inference mechanism uses both, the fuzzy rule base and the fuzzy data base to classify an input example.

The output of a rule-based fuzzy system can be defuzzified in order to obtain a continuous value. Such task, named defuzzification [21], can be implemented using several defuzzification methods, including centre of area, centre of gravity, first of maximum, mean of maxima, among others. Each method considers a different combination of the fuzzy sets, and their membership degrees, that are used in the rules, in order to provide a continuous value for the system. In this work, we use rule-based fuzzy systems to provide IAs with intelligent decisions regarding the management of the game.

3. Bibliographical review

Several proposals regarding serious games can be found in the literature related to software engineering. Proposals of traditional games, using boards and cards, can also be found. Next, we present some relevant proposals related to this work.

TIM: The Incredible Manager [4], is a single-player game that simulates planing and control (including budget, delivery plans, and quality) of software management in which each player has to set deadlines and assign tasks to each software developer and determine a number of software inspections. The final product is then presented for a possible acceptance.

In [5], the authors present a software development simulation model based on games in which artificial agents are used to represent software developers, who have their behaviour patterns. The game proceeds using extreme programming [22]. A similar proposal can be found in [6], which uses an educational game focused on mobile technologies based on SCRUM [23], an iterative and incremental software development framework. Another proposal, JETS (software test team game) [7], is a multi-player game that focus on software test training in under-graduation courses in computer science. JETS simulates the iterations of a software test team in a software development company.

Games specifically focused on teaching risk management in software projects are scarce. In [2], the author presents a traditional board game for up to four players designed specifically to teach risk management in software projects. Our proposal, eRiskGame, focuses on creating automatic decision making processes in order to improve the learning process in risk management in software projects. Such decisions have a direct influence on the learning process of the players. The proposal is presented and discussed next.

4. Proposal

In this Section, we describe eRiskGame, presenting its scenario, its agents, the fuzzy system to control the vitality of employees, the continuous evaluation and feedback of users, and the validation of the project.

4.1. Scenario of eRiskGame

The game simulates the competition among different software companies for accomplishing their software projects. Each user plays the role of a software project manager who is responsible for the following tasks:

1. Search the market for new resources for a specific software project;
2. Define budgets and project management plans to compete for new resources;

3. Control budgets during the execution of the projects;
4. Evaluate the productivity and the quality of the software produced by the work teams;
5. Hire new employees – the game provides a pool of available professionals with a list of characteristics used for their selection. Since the game is multi-player, such professionals can be hired by any player at any time, similarly to the real world;
6. Fire employees with low productivity;
7. Deal with changes in the administration of the organization, such as employees receiving better salary offers from competitors and asking to quit the company;
8. Tackle problems such as delays in delivery plans, and poor software quality, among others.

Players are offered a tutorial in order to speed their adaptation to the environment. Players can follow their projects by means of charts and graphs.

The work teams have to be planed based on the individual evaluation of each employee of the team, as well as possible professionals available in the market. Such characteristics include: team work ability, leadership, focus, technical knowledge and skills, previous jobs, cost per hour, and average code writing productivity. Such characteristics are defined in terms of fuzzy linguistic variables whose domains range from 1 to 10, defined by the following linguistic terms *Very Low*, *Low*, *Average*, *High*, and *Very High*. These fuzzy sets are evenly distributed in the domains of the variables.

One of the main goals of each player is to foresee challenges related to hiring and firing employees, negotiating new human resources, and having to deal with problems related to team workers. The offer of professionals and other resources, as well as the control over the characteristics of the employees, are defined and controlled by Intelligent Agents (IAs), described next.

4.2. Multi-agent system

A real software project management environment is quite complex, including project, product and business risks. In order to represent such scenario, IAs are used, supplying the game with such risks in different stages and situations of each project. The IAs are responsible for:

- Inserting new resources in the market;
- Stimulating competition among users for new resources;
- Generating new professionals with individual characteristics in the work market;
- Controlling the vitality, focus, and productivity of employees;
- Generating conflicts between employees;
- Dealing with quitting requests;
- Generating competition for hired professionals by offering better salaries.

Each IA has a probability attribute to a given event. Such probabilities were empirically defined.

The domain knowledge of the system is implemented using rule-based fuzzy systems. Regarding the insertion of conflict among employees, an IA checks the number of workers in each project, the time spent on the projects, the average number of errors in the project, and informs the player if there is any tension in the team regarding factors such as organization changes, changes in the project, salaries, or even the cancellation of a project. The game expects the user to take actions in case of conflicts. Such actions include pay raises, assign employees with paid extra hours, hiring new professionals, and offering training courses. Occurrences in the game are informed to users by messages sent to their message in-box.

Another agent controls the number of professionals available in the market, aiming at simulating a real scenario. In order to make eRiskGame more dynamic and realistic, the attributes of each worker are not constant: they suffer alterations so that a worker can present low productivity and focus, for instance, affecting their performance. Such problems can be solved by offering training courses from a list, which includes technical and motivational courses. Since such courses cost money, the user has to control their impact on the company budget.

eRiskGame has its own time counting strategy, *i.e.*, days pass by regardless of players intervention. Time in eRiskGame runs faster than in the real world in order to accelerate the completion of projects and to allow learning in suitable time. A complete work day happens at every 4.8 hours so that a whole week is simulated in a single day. This definition of simulating a week in approximately 5 hours was empirically defined, based on the fact that players need to log in at least twice a day in order to be able to control most of the occurrences. In fact, since eRiskGame simulates its evolution hourly, updates in the attributes of the employees might happen and require actions. This way, if a player does not check the game for more than two real days, there might be irreversible problems.

4.3. Fuzzy system to control employees

In order to illustrate the use of the fuzzy logic in the game, next, we describe the rule-based fuzzy system connected to the IA responsible for controlling the vitality of the employees. The lack of vitality might lead to sick days paid by the company. The IA monitors the motivation and vitality of all employees allocated to projects and identifies workers who need temporary licences (sick leaves). Figure 1 shows the definition of the input variables of the system, *Motivation* and *Vitality*, as well as the output, *Health*.

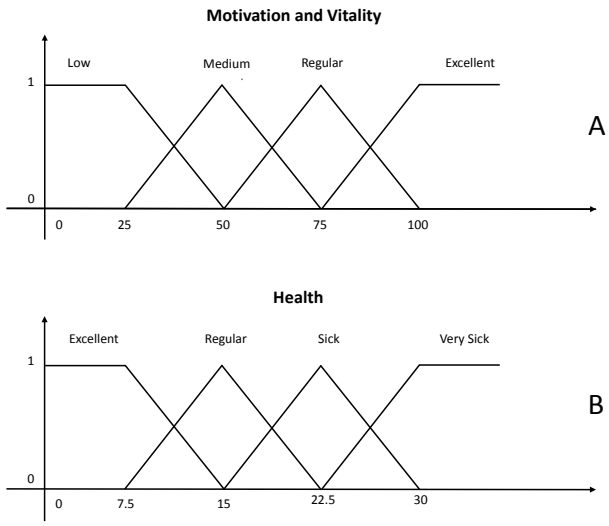

Figure 1: Definition of the fuzzy sets representing the *Motivation*, *Vitality* and *Health* characteristics of professionals.

Figure 1-A shows the input attributes of the system: *Motivation* and *Vitality*. Figure 1-B. shows the output of the system, the worker's *Health*. All attributes are represented by four fuzzy sets, two trapezoidal and two triangular shaped ones. The domain of the input variables is the interval [0, 100] and contains the fuzzy linguistic values *Low*, *Medium*, *Regular*, and *Excellent*. The domain of the system output represents the number of days a worker is supposed to be home and contains the fuzzy linguistic values *Very Sick*, *Sick*, *Regular*, and *Excellent*. In this case, the fuzzy system defines the absence from work from 0 to 30 days. The defuzzification method used is the centre of gravity [8]. Table 1 presents the rules of the fuzzy rule base of the fuzzy system.

Table 1: Rule set of the fuzzy rule base of the fuzzy system.

Motivation	Vitality	Health
Low	Low	Very Sick
Low	Medium	Sick
Low	Regular	Regular
Low	Excellent	Regular
Medium	Low	Very Sick
Medium	Medium	Sick
Medium	Regular	Regular
Medium	Excellent	Regular
Regular	Low	Sick
Regular	Medium	Regular
Regular	Regular	Regular
Regular	Excellent	Excellent
Excellent	Low	Regular
Excellent	Medium	Regular
Excellent	Regular	Excellent
Excellent	Excellent	Excellent

As with any occurrence in the game, in case workers have health problems and need to be away from work, players receive information in their message in-box.

4.4. Continuous evaluation and feedback of users actions

In order to evaluate the actions of the users and inform them right after they have taken such actions, eRiskGame includes an evaluation module. The idea here is to use a rule-based fuzzy system in order to evaluate user actions. eRiskGame also presents some statistics based on user actions. Such statistics are related to number of projects, budget balance of each project, number of employees hired and fired, number of employees in each project, team and individual productivity, and a status of each project reporting how many days the project is been developed and its project management plan.

The rule-based fuzzy system that connects to the IA that controls this module of the game is currently being empirically defined.

4.5. eRiskGame validation

Teaching and learning are difficult to evaluate. Opposite to the quality of goods, which can be measured by indexes or indicators such as durability and number of defects, the quality of educational services has a more abstract nature, making it difficult to evaluate them [24].

Since this is an on-going project, eRiskGame is currently being validated as a support teaching tool in two software engineering courses at the Federal Rural University of the Semi-Arid and the Federal Institute of Rio Grande do Norte, Brazil. After the training, students and professors are expected to answer specific questionnaires to lead us to updates and to the proposal of new functionalities. We intend to use their feedback to validate the project.

5. Conclusions and future work

The training of professionals in risk analysis and management in software projects requires direct contact with the challenges related to the area. Due to time and space restrictions, students of traditional software engineering courses can benefit from serious games to complement their education and professional qualification.

In this paper, we describe an on-going project to develop a serious game, eRiskGame, focused on the learning of risk management in software projects. eRiskGame simulates an environment where many companies, each represented by a player who plays the role of software project manager, competes for resources in the market for their software projects. Such projects have to be developed by workers teams, hired and managed by the user. The game provides the user with risks related to the real activity of managing software projects, such as changes in the projects, worker quitting the team, having to hire new workers, deal with problems in the workers performances, controlling projects, as well as presenting proposals in order

to get new resources in the market. The system is modelled using intelligent agents that operate based on rule-based fuzzy systems. The fuzzy systems contain sets of rules that store domain knowledge in order to control changes in the system. eRiskGame is browser-based and persistent. Thus, players can access the system at any time and independently from platform. The simple graphics included in the game make it light enough to run even in low processing power machines, such as mobile devices.

eRiskGame is being validated as a support teaching tool in software engineering courses at two universities.

As future work, we intend to improve the system based on feedback from students and professors who use the system by adding new functionalities and updating and tuning the fuzzy knowledge bases.

Acknowledgments

This research is funded by the Brazilian National Council of Technological and Scientific Development - CNPq.

References

[1] J. L. Brewer. Project managers: can we make them or just make them better? In ACM, editor, *6th Conference on Information Technology Education*, pages 167–193, 2005.

[2] Gil Taran. Using games in software engineering education to teach risk management. In *Conf. on Software Engineering Education & Training*, pages 211–220, 2007.

[3] Robert W. Wagner. Edgar dale: Professional. *Theory Into Practice*, 9:89–95, 1970.

[4] A. R. Dantas, M. O. Barros, and C. M. L. Werner. A simulation-based game for project management experimental learning. In *Int Conf on Software Engineering and Knowledge Engineering*, pages 19–24, 2004.

[5] R. Agarwal and D. Umphress. A flexible model for simulation of software development process. In *48th Annual Southeast Regional Conference, pp. 1-4. ACM*, 2010.

[6] A. M. C. Campos, A. Signorett, P. Lima, E. Luis, M. Fontes, and K. Dantas. A game for project management practice. In *Brazilian Symposium on Informatics in Education*, 2011. (in Portuguese).

[7] F. M. Muller T.G. Silva, G. Bernardi. Software test teaching support approach based on serious games and virtual worlds. In *Brazilian Symposium on Informatics in Education*, 2011. (in Portuguese).

[8] G. J. Klir and Bo Yuan. *Fuzzy Sets and Fuzzy Logic - Theory and Applications*. Prentice-Hall, 1995.

[9] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[10] K. Ayas and N Zeniuk. Project-based learning: Building communities of reflective practitioners. *Management Learning*, 32(1):61–76, 2001.

[11] R. C. Hsu and W. Liu. Project based learning as a pedagogical tool for embedded system education. In *Int. Conf. on Information Technology: Research and Education*, 2005.

[12] .-J. Kuo. How does an online game based learning environment promote students' intrinsic motivation for learning natural science and how does it affect their learning outcomes? In *Digital Game and Intelligent Toy Enhanced Learning*, pages 135–142, 2007.

[13] K. L. McClarty, A. Orr, P. M. Frey, R. P. Dolan, V. Vassileva, and A. McVay. A literature review of gaming in education. In *Gaming in education*, pages 1–35, 2012.

[14] E. Hall. *Managing Risk: Methods for Software Systems Development*. Addison-Wesley, 1998.

[15] M. A. Ould. *Managing Software Quality and Business Risk*. John Wiley & Sons, 1999.

[16] S. L. Pfleeger and J. M. Atlee. *Student Study Guide for Software Engineering: Theory and Practice*. Prentice Hall Press, 2009.

[17] I. Sommerville. *Software Engineering*. International Computer Science. Addison-Wesley Longman Publishing Co., 8 edition, 2006.

[18] S. Franklin and A. Graesser. Is it an agent, or just a program?: A taxonomy for autonomous agents. In *Int. Work. on Agent Theories, Architectures, and Languages*, pages 21–35, 1996.

[19] A. Dziuk and R. Miikkulainen. Creating intelligent agents through shaping of coevolution. In *IEEE Cong. on Evolutionary Computation*, pages 1077–1083, 2011.

[20] A. A. Pontes and G. A. L. Mendes Neto, F. M.and Campos. Multiagent system for detecting passive students in problem-based learning. *Adv. in Soft. Computing*, 71:165–172, 2010.

[21] W. V. Leekwijck and E. Kerre. Defuzzification: criteria and classification. *Fuzzy Sets and Systems*, 108-2:159–178, 1999.

[22] S. Shahzad. Learning from experience: The analysis of an extreme programming process. In *Int. Conf. on Information Technology: New Generations*, pages 1405–1410, 2009.

[23] K. Schwaber and M. Beedle. *Agile Software Development with Scrum*. Prentice Hall, 2001.

[24] V. A. Zeithaml, L. L. Berry, and A. Parasuraman. The behavioral consequences of service quality. *The Journal of Marketing*, 60 (2):31–46, 1996.