Gamifying Software Development Environments Using Cognitive Principles

Position Paper

Naomi Unkelos-Shpigel, Irit Hadar

Information Systems Department, University of Haifa
Carmel Mountain 31905, Haifa, Israel
{naomiu, hadari}@is.haifa.ac.il

Abstract. The topic of enhancing the software development process has received much attention in recent decades. Several models have been developed to this end, typically addressing the characteristics of the process or the organization. We believe that an additional, substantial enhancement of software development can be achieved via encouraging productive behavior among individual software developers. In this paper, we propose a framework for enhancing motivation among software developers, using gamification principles. As a first step in this ongoing research, we designed a prototype for a game, which can be used in various tasks in the software development process. The game is based on cognitive theories, which address motivation among practitioners.

Keywords: Gamification, motivation, flow theory

1 Introduction

Software development processes and their enhancement have been vastly researched in recent years. Many models have been proposed and implemented in industry, replacing the outdated waterfall model, such as the human-centered agile development methodology, and maturity models such as the CMMI.

The use of these models has contributed to the structure and maturity of the software development process. Nonetheless, while progress has been made, many challenges and difficulties still exist, introducing risks to their success. Additional solutions are needed to encourage and guide productive behavior among software engineers to further enhance the software development process and its outcomes [1]. More specifically, we believe that analyzing developers' behaviors and exploring potential solutions from a cognitive perspective would be beneficial to progress this objective.

Several cognitive theories address the topic of motivation in workplaces, including salient examples such as SDT – Self Determination Theory [2], the Flow Theory [3], and the Group Flow Theory [4]. These theories provide guidelines as to how to motivate employees to take active part in the work, and to encourage them to strive for more productive behavior, mainly by encouraging intrinsic and extrinsic motivation [2] and by achieving a state of flow, where the worker is immersed into the task [3].

Persuasive technologies, and specifically gamification, were acknowledged as changing employees' motivation and behavior. Gamification is defined as "the integration of Game Mechanics in non-game environments to increase audience engagement, loyalty and fun" [5, p.2], and was found to encourage users to participate and contribute in computer-supported applications. In recent years, various gamification elements have been embedded in different information systems and applications in general, and, in some rare cases, in applications intended for software engineers in particular. In this research, we aim to explore, and empirically examine, new and effective ways for enhancing software engineering via gamification.

Leveraging on the principles of gamification and based on cognitive motivation theories, our research questions are: (1) What can we learn from cognitive motivation theories toward designing effective gamified environments for software developers? (2) How can we promote software developers' productive behavior via gamifying software engineering practices? (3) What are the actual benefits of embedding gamification techniques in software development environments?

The next section presents the background for our research. Section 3 details our proposed solution. Section 4 presents the planned research method, and section 5 discusses the expected contribution of the research.

2 Scientific Background

2.1 Gamification

Coined by Nick Pelling in 2002 [6] the term "gamification" is used in order to describe how any task can be performed as a game. In recent years, various research works have been conducted with regard to gamification, its mechanisms, and their use. While the early use of gamification was intended for games and application for users, research from the last few years is targeted on using gamification mechanisms for changing behaviors of specific populations for specific purposes. In the context of software development, several attempts to use gamification techniques were conducted.

Sheth et al. [7] gamified a number of software development activities in educational settings, in order to engage software engineering students in development, documentation, bug reporting, and test coverage, using social rewords. The students who used the system showed statistically proven improvement in their work results. The system was later used to encourage students into doing software testing, using a method they called "Secret Ninja Testing," where students were presented with quests using characters from various action movies, and were asked to act as these characters while solving testing problems [8]. The system helped the students to be exposed to the complete lifecycle of software development, and encouraged students to choose software engineering as a major in their studies.

Research was also conducted in the context of using gamification at early stages of software development. Dubois and Tamburrelli [9] suggested a framework to successfully integrate gamification elements into software engineering, starting from requirement elicitation. They identified three types of activities needed to be performed when engaging gamification into software engineering: analysis, integration and evaluation and found that students performing these activities had better results in software engineering. Another attempt to use gamification at early stages of software development showed that using gamification in virtual teams during requirement elicitation assisted the teams to locate experts and share their knowledge [10].Another effort to gamify software development in educational settings was done in the context of early stages of software development, successfully integrating gamification elements into requirement elicitation [11]. This study identified three types of activities needed to be performed when integrating gamification into software development: analysis, integration and evaluation, and found that students performing these activities produced better outcomes.

Thus far, gamification for software engineering has focused on education, using gamification principles borrowed from the domain of applications and website usage. We did not find research in this context relying on cognitive theories in order to design games for software engineers, or using gamified environments in industry in order to motivate practitioners to enhance work performance.

2.2 Motivation Theories

Several cognitive theories address the topic of encouraging motivation for work tasks. Here we briefly present three of the most influential theories in this field.

The Self Determination Theory (SDT) [2] presents a continuum of motivation types, from intrinsic motivation that emerges from the employee, to extrinsic motivation created by rules and regulation in the workplace. Although intrinsic motivation is considered to be linked to positive human behavior, SDT suggests that proper use in extrinsic motivation can lead to motivated behavior. According to the Theory of Flow [3] there are five elements of reaching to a state where the individual is immersed into the performed task (some of which can be extrinsically induced): Clarity, Centering, Choice, Commitment, Challenge. Sawyer [4] extended these elements to the context of group flow, to contain, among others, the following characteristics: A compelling, shared goal, a sense of being in control, blending egos, equal participation, familiarity, constant and spontaneous communication, and the potential for failure. We relied on these characteristics when designing our solution, creating an environment that would encourage group flow. The proposed solution is described in the next section.

3 The Proposed Solution: Gamifying Software Development

The objective of this ongoing research is to identify different opportunities within the software development process for enhancing productive behavior via gamification. In this paper we refer, as examples for demonstrating our vision, to six of the major tasks of the software development process. We view them in pairs:

- Software architecture design and software architecture review
- Coding and code review
- Customization (adapting the solution to the customer) and Integration Testing

We chose to focus on these pairs since we can identify in each pair two parties: The creator (architect, programmer, customizer) and the reviewer (architecture reviewer, code reviewer, tester). Each of these pairs will have its own game, according to the following principles (see Fig. 1):

Create – The creator creates a segment of work, according to her task (architecture, coding or customizing). Creating the segment assigns points to the creator and to her team.

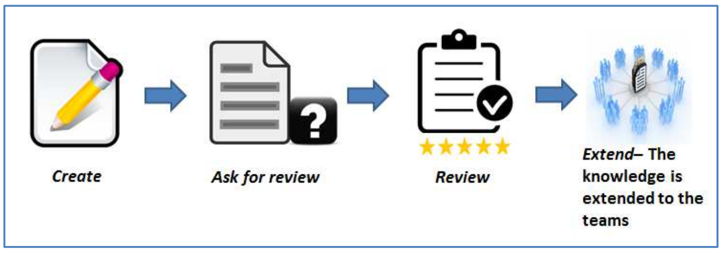
Request review – The creator sends the artifact for reviewer. The reviewer is randomly selected from a group of potential reviewers. The creator does not know which reviewer was selected. The reviewer and the team receive points for this action.

Review – The reviewer receives the anonymous artifact, reviews it and writes a review. She then submits the review to the system, with a review score, which reflects the quality of the artifact. The reviewer and the team receive points for this action. Additional points are given to the reviewer for writing comments for improvement.

Extend the knowledge – upon receiving the review, the creator and reviewer can further extend the knowledge created from the review (the artifact and the review comments), to their team or to an extended group of practitioners in the organization. Each such knowledge extension results in additional game points for the creator, the reviewer and the team. When the knowledge is used (actively checked-out) by another practitioner from the organization, the team is given additional points.

Fig.1 describes the principles of the game:

Figure 1. The principle of the game of CARE



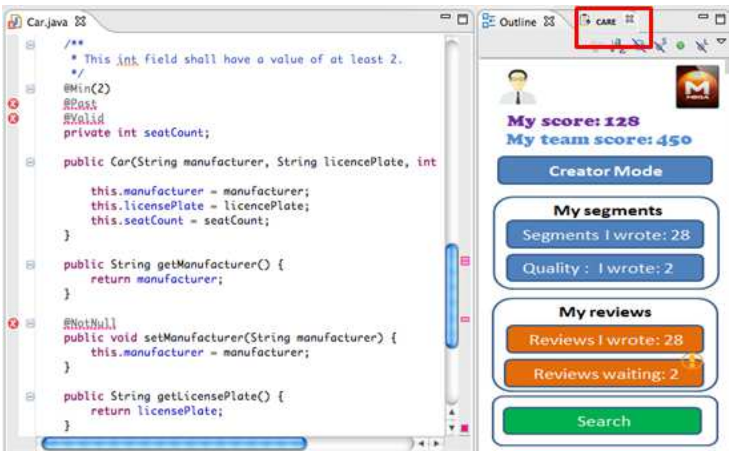
The game we designed has several key principles:

1. The game is embedded in the eclipse IDE. There is a special tab for the game in IDE, where the players can view their profile and all game related information.
2. Each player can see her profile as a creator or as a reviewer (see *Creator/ Reviewer Mode* button in Fig. 2 and 3 respectively). The creator's screen contains information about previous segments created and their average quality score (see Fig. 2, *My segments*). The reviewer's scree contains a section, which refers to the current segment (Fig. 3, segment of code marked in red). She can write a textual review, and provide a quality score for the segment (Fig. 3, *Segment review*).
3. When a segment of code is created and ready, the creator asks for a review, and is immediately rewarded with points.
4. The reviewer reviews the relevant segment, inserting both comments and a quality score. If the reviewer approves the code, she is granted with points as well. Additional score is given for writing a review, which helps the programmer to

improve the code. For bug detection, the reviewer will be rewarded extra points for each bug found.

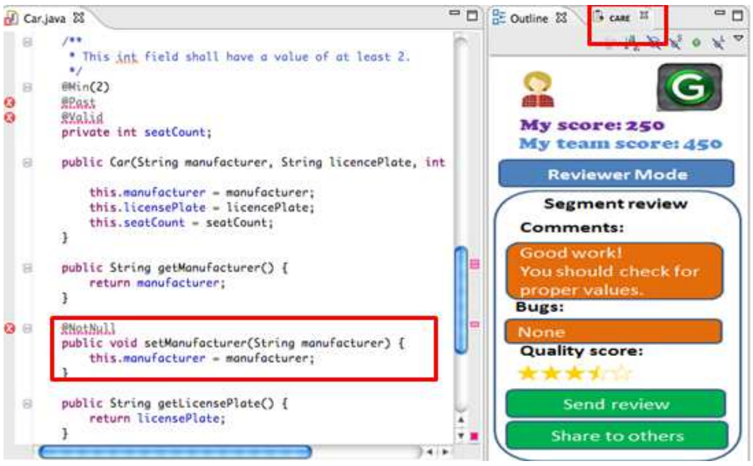
5. The reviewers can choose to share their review comments with members of other teams (pending creators’ permission), raising both individual and team score. When the reviewer wishes to share the comment, a message will be prompt to the creator to ensure he agrees to share the segment and the comments. The names of the teams that used the comments are displayed in the team`s profile (see Fig.4, bottom). Additional mechanism is needed to evaluate the quality of the shared information, and its contribution to other stakeholders in the project.
6. The creator can also search for tips and lessons learnt from the review with other creator and reviewers (Fig. 2, *Search*). Using this knowledge (by checking it into the project) raises both individual and team score
7. The creators and reviewers are also given badges according to their individual scores. The badge indicates their level in the game, labeled: kilo, mega, or giga, etc., according to the number of points they earned (Fig 2 and 3, top). All the badges of the team members are displayed in the team`s profile, sorted in groups according to levels (Fig. 4).
8. In addition to the individual scores, there is also a team score managed, which is updated according to the individually rewarded tasks (Fig. 2, top).
9. The teams are rewarded each month according to their scores. The reward can be in the form of monetary incentive or other rewards (e.g., breakfast with a high management representative or coupons for fun activities).
10. If other creators or reviewers use the knowledge and tips shared, the individual who wrote and/or shared this knowledge receives additional points.

Figure 2 .The creator`s screen



```
/**
 * This int field shall have a value of at least 2.
 */
@Min(2)
@Past
@Valid
private int seatCount;

public Car(String manufacturer, String licencePlate, int

    this.manufacturer = manufacturer;
    this.licensePlate = licencePlate;
    this.seatCount = seatCount;
}

public String getManufacturer() {
    return manufacturer;
}

@NotNull
public void setManufacturer(String manufacturer) {
    this.manufacturer = manufacturer;
}

public String getLicensePlate() {
    return licensePlate;
}
```

Figure 3 .The reviewer`s screen



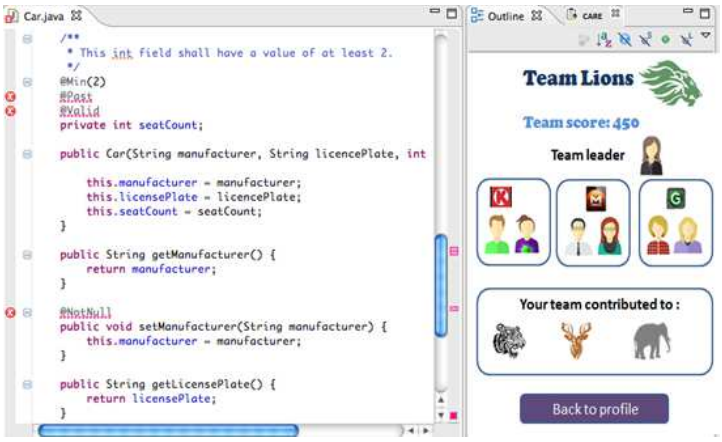
Car.java

```
/**
 * This int field shall have a value of at least 2.
 */
@Min(2)
@Past
@Valid
private int seatCount;

public Car(String manufacturer, String licencePlate, int

    this.manufacturer = manufacturer;
    this.licensePlate = licencePlate;
    this.seatCount = seatCount;
}

public String getManufacturer() {
    return manufacturer;
}

@NotNull
public void setManufacturer(String manufacturer) {
    this.manufacturer = manufacturer;
}

public String getLicensePlate() {
    return licensePlate;
}
```

My score: 250
My team score: 450

Segment review
Comments:
Good work!
You should check for proper values.
Bugs:
None
Quality score:

Figure 4 .The team's screen



Car.java

```
/**
 * This int field shall have a value of at least 2.
 */
@Min(2)
@Past
@Valid
private int seatCount;

public Car(String manufacturer, String licencePlate, int

    this.manufacturer = manufacturer;
    this.licensePlate = licencePlate;
    this.seatCount = seatCount;
}

public String getManufacturer() {
    return manufacturer;
}

@NotNull
public void setManufacturer(String manufacturer) {
    this.manufacturer = manufacturer;
}

public String getLicensePlate() {
    return licensePlate;
}
```

Team Lions
Team score: 450
Team leader
Your team contributed to :

The game includes the following gamification elements:

Personal profile – the team members have individual profiles, where they can view their personal and team score. Each team has a team profile, presenting the team members and the team’s score.

Badges – the team members are assigned with badges according to their individual scores. The badge indicates their level in the game, according to the number of points they earned.

Scoreboard – each team has its members rated by their scores, and at the end of each month one team member is rewarded as "the player of the month."

The game supports SDT[2], since it offers rules and regulation (in the form of a game), to encourage employees to ask for review, and to share their knowledge with practitioners outside of their team, thus creating extrinsic motivation.

According to the theory of flow, the conceptual design of the gamified environment we propose supports the five elements of flow [3]: *Clarity* – The game and scoring in the game are simple and clear; *Centering* – the game is designed to make players feel they are in the center, gaining individual points and making their contributing to the team visible; *Choice* – players can choose whether to share their knowledge; *Commitment* – since all the players are team members, they are encourages to perform activities which raise team score; *Challenge* – the game provides challenges to all the stakeholders in the process, when they are required to improve the quality of their work, or the work they review, in order to earn additional individual and team points.
The game`s conceptual design also supports the group flow elements [4]: *A compelling, shared goal* – all the players have the shared goal of getting a high team score; *a sense of being in control* – since players send their work when they choose, they have full control on their progress in the game; *blending egos* – since there is a team goal, along with the personal goal, all the players' egos are blended to achieve high team score; *equal participation* – each of the players is allowed to participate in the game equally; *familiarity* – all the players in the same team are personally familiar (with at least part) with each other; *constant, spontaneous communication* – the virtual game allows all the players to communicate with each other; *the potential for failure* –As there is an ongoing race among the individual players and among the teams, low achievements relative to other players can be considered as failures..
To conclude, our proposed gamified environment supports principle of three cognitive theories – referring to individual and group motivation – designed to meet the challenge of achieving full commitment to the task, from both individual and team points of view.

4 Validation

This ongoing study will apply both qualitative and quantitative research method for validating and further refining CARE. As derived from the research objective and questions, the research focuses on human-related processes, calling for the use of qualitative research methods [12], and on performance, which can be quantitatively measured.

The qualitative study will focus on understanding how different gamification techniques affect software developers' motivation and behavior. The main population of this study will be software developers with different levels of seniority. Additionally, as the research settings and its preliminary results will require, we will expand the research population to other roles within the software development process.

Following the findings of the qualitative study, we will refine our conceptual design of the gamified development environment. This design will be implemented and evaluated according to the principles of design research [13]. More specifically, we will focus on measuring the quality and quantity of the software developed, with and without the use of the gamified environment. We plan to conduct this study with the participation of software engineering students and, if possible, practitioners, to find if, and to what extent these means improve their performance and promote desired behavior.

5 Expected Contribution

Gamification has been quite thoroughly researched among different types of users in recent years. However, we find only few examples of gamification research in the context of software development, most of which are intended for students and discussed in the context of education.

Since we wish to contribute to the software and information systems engineering industry, we plan to elicit data and validate our findings and results with practitioners, thus receiving non-biased opinions, aiming at the target population. The results of our study are expected to help organizations in increasing software developers' motivation to complete their tasks successfully and efficiently.

This research will contribute to the academic research community by providing empirical insights into the use of gamification as a means for enhancing software development processes, and the cognitive and social implications thereof.

6 References

[1] Hadar, I.: When Intuition and Logic Clash: The Case of the Object Oriented Paradigm, Science of Computer Programming, vol. 78 (2013), 1407-1426 (2013)

[2] Ryan, R. M., and Deci, E. L.: Self-determination theory and the facilitation of intrinsic motivation,social development,and well-being. American psychologist, 55(1),68 (2000)

[3] Csikszentmihalyi, M.: Flow and the Psychology of Discovery and Invention. Harper Perennial, New York (1997)

[4] Sawyer, K.: Group Genius: The Creative Power of Collaboration, Basic books.(2008)

[5] Deterding, S., Khaled, R., Nacke, L., and Dixon, D.: Gamification: Toward a definition. In CHI 2011 gamification Workshop Proceedings ,12-15 (2011)

[6] Hägglund, P.: Taking gamification to the next level (2012)

[7] Sheth, S. K., Bell, J. S., and Kaiser, G. E.: Increasing Student Engagement in Software Engineering with gamification (2012)

[8] Bell, J., Sheth, S., and Kaiser, G.: Secret ninja testing with HALO software engineering. Proceedings of the 4th int'l worksop on Social software engineering ,43-47ACM (2011)

[9] Dubois, D. J., and Tamburrelli, G.: Understanding gamification mechanisms for software development. In Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering , 659-662 ACM (2013)

[10] Marshburn, David G. and Henry, Raymond M.: Improving Knowledge Coordination in Early Stages of Software Development. SAIS 2013Proceedings. Paper 23 (2013)

[11] Hevner, A. R., and March, S. T.: The information systems research cycle. Computer, 36(11), 111-113 (2003)

[12] Bogdan, R. C., and Sari K. B..: Qualitative research in education. An introduction to theory and methods. Allyn & Bacon, A Viacom Company, 160 Gould St., Needham Heights, MA 02194; (1998).

[13] Hevner, A. R., and March, S. T.: The information systems research cycle. Computer, 36(11), 111-113 (2003).