Method for Game Development Driven by User-eXperience: A Study of Rework, Productivity and Complexity of Use

Mario González-Salazar*, and Hugo Mitre-Hernández†
Software Engineering Group,
Center for Research in Mathematics (CIMAT)
Av. Universidad 222, 98068
Zacatecas, Mexico

Carlos Lara-Alvarez‡
CONACYT Research Fellow
Center for Research in Mathematics (CIMAT)
Av. Universidad 222, 98068
Zacatecas, Mexico

***Abstract*—The growing capabilities and revenues of video game development are important factors for software companies. However, game development processes could be considered immature, specifically in the design phase. Ambiguous requirements in game design cause rework. User-eXperience (UX) is usually assessed at the end of the development process, causing difficulties to ensure the interactive experience between the game and users. To reduce these problems, this paper proposes a method for Game Development driven by User-eXperience (GameD-UX) that integrates a repository based on requirements engineering, a model for user experience management, and an adjusted agile process. Two experiments were conducted to study rework and productivity of video game development. Results of the first experiment revealed that GameD-UX causes less rework than conventional approaches, but it induces lower productivity. A tool for supporting the GameD-UX method was developed by considering the lessons learned. The second experiment showed that the software tool increases the productivity and reduces the complexity of use of GameD-UX.**

Keywords*—Rework; Productivity; Complexity of Use; Video Game Development

I. Introduction

Video games are important economically, they constitute the main entertainment industry, with continuous growth and billions of dollars in sales and revenues [1]. CEO of the Entertainment Software Association (ESA) point out that "Video games are the future; from education and business, to art and entertainment, our industry brings together the most innovative and creative minds to create the most engaging, immersive and breathtaking experiences we've ever seen..." [1]. However, ensuring the correct level of interactive experience between the game and the player is a challenge [2]; additionally, 65% of problems in game development are generated at pre-production stage and are related to unspecified or ambiguous requirements in game design [3], [4] causing rework and low productivity.

This article presents the *Game Development driven by User-eXperience* (GameD-UX) method that is composed by an improved Game Design Document (iGDD) [5] from requirements engineering perspective, a model for Game Experience Management (GEM) [6] from software architecture approaches, and an adapted agile method for game development. As shown in [6] the GEM model is able to improve the User eXperience (UX).

Besides defining the game requirements, the iGDD formalizes the game design by using software requirement principles. The main idea behind the GEM model is to transform the desired experience into game attributes. The quality attributes in conventional software products are security, usability, performance, among others; but in UX, these attributes can be seen as factors such as enjoyment, excitement, frustration, boredom, fear and more. Finally, the iGDD and GEM were included into a modified agile model for game development.

Two experiments were conducted to compare rework, productivity, and complexity of using GameD-UX and a conventional approach to game development. Results of the first experiment revealed that GameD-UX causes less rework, but it also induces lower productivity. The main difficulties found were attributed to failures in capturing and querying information from the iGDD and GEM – i.e. the low productivity was caused mainly by the complexity of using text documents; to overcome these limitations, a tool for supporting the GameD-UX method was developed. The second experiment confirmed that the GameD-UX supported by an appropriated tool produces better results in terms of rework, productivity, and complexity of use.

The rest of this article is organized as follows: section II presents the work related to the game repository, development and UX evaluation. Section III presents the method for Game Development driven by User-eXperience and its tool. Section IV explains the experiments. Section V presents and discusses the results. Finally, conclusions are presented in section VI.

II. Related Work

This section presents the related work in video game development (repository, UX evaluation, and development models), and how the proposed method alleviates the problems found in conventional approaches.

The game development process is composed by three stages: pre-production, production, and post-production [7]. The pre-production stage focuses mainly on creating the game concept and design. The production stage creates and validates the software; this stage also produces visual and auditory assets required by the game. Finally, the post-production

stage distributes and maintains the game; it also manages the feedback coming from different sources – e.g., reviews.

A. Game Design Repository

Games are complex systems requiring significant effort in the first two development stages – pre-production and production. These complexities can increase the amount of rework and consequently, the cost of the game. The rework can be avoidable in most cases by detecting and correcting problems in early stages. A game design document can help to specify and structure the requirements of the game. The following sections have been proposed in the literature:

- **Overview.** Almost all authors suggest that a GDD should include a section that summarizes the key elements of the game to keep the eyes on the road [8]. Some authors even include a subsection of goals or objectives of the game [8], [9], [10], [11], [12], [13], [14].
- **Mechanics.** The term *mechanics* is used to describe game elements – e.g., player characters – and interaction rules – e.g., a player-enemy interaction. Mechanics include characters or assets list [8], [9], [10], [11], [12], [13], [14].
- **Dynamics.** Several proposals have common sections that contains intended interactions with the player such as interfaces, levels or challenges [8], [9], [10], [11], [12], [13], [14].
- **Aesthetics.** It is what the player perceives by his visual and auditory senses. Most authors only cover the visual aspects in a document called the art bible. Baldwin [12] suggests a GDD template that abbreviates an art bible. Auditory assets can also be included in this section [8], [9], [14].
- **Experience.** Creating enjoyable player experience is fundamental for the game success [2]. Player experiences are enriched by mechanics, dynamics and aesthetics of the game. Playability can be used to link game design to player experience [15]. Therefore, defining the expectations of player experiences may lead to the improvement of the game and to the establishment of a base line to test the experiences in production.
- **Assumptions and Constraints.** Some authors include technical limitations in the technical bible [10], [12], [14].

An effort to integrate the previous sections into a single repository, the authors of this paper have proposed the improved Game Design Document (iGDD) [5]; iGDD sections are related to the Software Requirement Specification (SRS) characteristics as described in Table I. The method proposed in this paper also uses the iGDD as repository.

TABLE I. DESCRIPTION AND CHARACTERISTICS OF SECTIONS OF THE IGDD

iGDD Section	Description	SRS Characteristics
Overview	Describes briefly the most important aspects of the game.	Relations with other documents, and common language for better understanding
Mechanics	Describes the elements of the game.	Organization of game requirements (objects organization).
Dynamics	Describes how the elements of the game will take action in the game.	Organization of game requirements. Relation of complexity with gamer profile.
Aesthetics	Describes what the player perceives directly through their sense, like what he sees and hears.	It is not related to the SRS.
Experience	Highlights important aspects of the game and what you hope to achieve from these aspects.	Decision-making based on trade-offs of game parts. Quality attributes on video games.
Assumptions and constraints	Narrates the aspects of the design assumptions and limitations of the game, either technical or business.	Knowledge of game parts for reviews. Limitations or boundaries of video game

B. Game User eXperience Evaluation

Guaranteeing an enjoyable User eXperience (UX) is critical for game companies. There are some related works seeking to solve the problem:

- **Core Elements of the Gaming Experience (CEGE)**. Calvillo et al. [16] suggest that core elements to ensure UX are: puppetry (control, ownership, and facilitators) and video game (game-play and environment). They also propose a questionnaire for evaluating these elements.
- **Game Experience Questionnaire (GEQ)**. Engl and Nacke [17] consider that immersion, flow, competence, tension, challenge, positive and negative affect are UX evaluation factors. They also propose a comparative evaluation instrument.
- **Heuristics.** Hochleitner et al. [18] propose a framework of heuristics (design guidelines for aesthetics and mechanics in a game genre) categorized in game play/story, and virtual interface to asses UX.

Although heuristics are part of the game design, they are considered general guidelines that belong to a game genre; similarly, the GEQ instrument does not ensure the UX because it can be only used after the game is finished. Conversely, integrating CEGE components for designing the game could avoid unpleasant experiences. In a previous work, the CEGE was compared to the *Game Experience Management Model*; games developed using the GEM, improves the UX [6].

The GEM is based on software architecture because of its advances in software systems design. The interaction experience between game and player can be interpreted as a set of quality attributes in software engineering [2]. In traditional software systems, quality attributes include: security, usability, performance, etc. In video games these attributes could be considered as factors of User eXperience (UX) as: enjoyment, excitement, attention; these attributes are closely related to emotions and cognitions of the player.

The quality attribute approaches in software architecture design can be categorized into [19]:

- **Quality Attribute Requirement Focused (QARF).** These approaches perceive Quality Attribute requirements as the main focus in the software architecture design phase, and consider each design decision based

on its implications on the prioritized quality attributes [20];

- **Quality Attribute Scenario Focused (QASF).** These approaches map architectural quality goals into concrete scenarios to characterize stakeholders concerns throughout the software architecture design phase [20], [21];
- **Influencing Factor-Focused (IFF).** These approaches focus on the inter-dependencies among factors and constraints that would affect the choice of design decisions [21].

Software architecture design is an area that can bring the solution idea for UX management in pre-production stage. The QARF and QASF approaches are suitable for game design due to its aspects of quality attributes requirements, prioritized QA, and the scenarios of design. Into these categories we can find the Quality Attribute Workshop (QAW) and the Attribute Driven Design (ADD) methods. The QAW [20] is a facilitated, early intervention method used to generate, prioritize, and refine quality attribute scenarios before the software architecture is completed. The ADD defines software architecture by basing the design process on the quality attributes that the software must fulfill [21].

Initially, the GEM model [6] defines a high-level game design that associates game goals of the iGDD with the desired experience; the experience is described in design drivers, detailed in guidelines and verified by test cases. A design driver is a high-level property that the game should have in order to generate the intended experience in the player; i.e., the user experience metrics – emotions as fear, happiness, angry, etc. In each iteration of game development, game elements are created and checked to confirm that the game is achieving the goals. A game design guideline is a description of how game elements need to be created in order to achieve the intended experience established in the game design drivers. Finally, the test cases evaluate guidelines in terms of fulfillment of their goals; it could include a questionnaire or an emotional evaluation model and its relation to parts of the game. There must be at least one test case per guideline. In sum, GEM is able to design and manage the expected UX in the proposed method.

C. Video game development models

The video game development is a form of software development that adds additional requirements, – e.g., artistic aspects; hence, many of the management tools and standards from the software industry can be useful for game development. Game projects are usually more complicated than software projects because they involve a multidisciplinary team and they usually have more uncertainty around project goals. Software development models – e.g. waterfall, iterative, or extreme can be used for developing video games [7]. In general, the waterfall model is considered inadequate because it is highly structured and it cannot be adapted to changes in the requirements; therefore, more flexible models are needed [22], [2], [23].

Agile methodologies – i.e. Scrum [24] or eXtreme Programming [25] – are better suited for the challenges of game development [26], [27]; they have been adapted to game development by using other tools as complements: user stories [26], game design documentation [28], or workshops for strengthening the interaction between clients and developers [29].

The adjustment of software development to a specific context is well studied in software engineering through patterns. Patterns [30] are used to solve a generic problem: given a narrative and context of the problem to be solved, they propose a solution. They can be used for formalizing the knowledge about the development process. In [31] the authors propose the Software Development Project Pattern (sdPP) framework. For testing this approach [31], generates four instances of the sdPP with agile development models; one of these instances Scrum sdPP is suitable for game agile development because it allows to follow an iterative process without sacrificing creativity. The resulting workflow and productflow can guide game developers between the activities and their corresponding input and output products.

An sdPP instance of Scrum was adapted for agile game development. In this modified pattern instance, the iGDD and GEM were integrated. The main activities in relation with iGDD and GEM are described in the proposed method.

III. Proposed Approach

This section first describes the GameD-UX activities and how they are related to the iGDD and the GEM, then it describes the software tool improvements based on the lessons learned of the initial experiment.

Game Development driven by User-eXperience (GameD-UX) is a method to design and develop video games from the required user experience. It uses two components: (i) the improved Gamed Design Document (iGDD) for the repository of all structured game elements, and (ii) the Game Experience Management (GEM) model to capture and manage the required UX along game development.

A. Method for Game Development driven User-eXperience

GameD-UX is composed of a repository containing game design (iGDD); a model to design, track and manage user experience (GEM); and an adapted Scrum method for game development, with the aim to design and develop video games based on user experience. Scrum was selected because is a flexible framework that can be adapted to other methods or tools – e.g., user stories, Kanban board –, its cycle life is iterative, incremental, and evolutionary. It does not sacrifice creativity, and it is well documented [26].

The GameD-UX activities are based on the general Scrum activities. Fig. 1 illustrates these activities and their relationship to iGDD and GEM. The following paragraphs describe these activities:

1) **Initiate the project.** A game development project may have different sources: an original idea for a game, or an opportunity found in a specific market. Once a game idea from some source initiates a game development project, the first activity is to assign resources and to transform the game idea into the game concept.

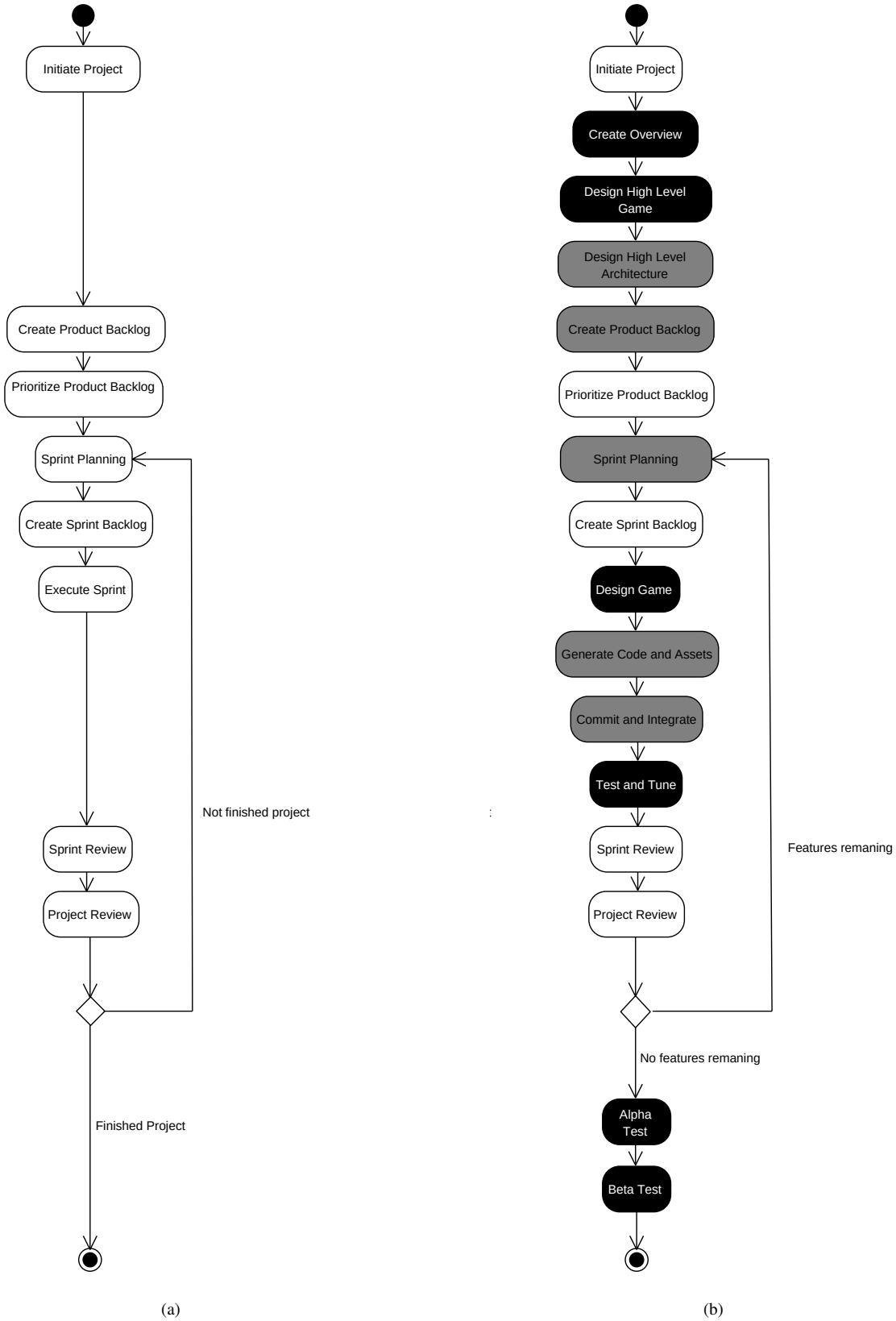


Fig. 1. Comparison of the general activities between: (a)Scrum, and (b)the proposed approach. The proposed approach adds (black) and modifies (gray) activities.

2) **Create overview** *(product: overview of the iGDD)* The overview describes the game in a brief abstract, identifies the main objectives of the game, the genre of the game, asks questions e.g., why the game is

worth doing, defines which type of players would like to play the game, and what will be the main activities that the player will be doing while playing the game.

3) **Design high level game** *(product: overview of the iGDD, drivers and guidelines of the GEM).* The high level game defines some main features of the game: the game modalities (single player, multiplayer, online, arcade mode, history mode, among others), the platform or platforms on which the game is intended to run, the game theme (medieval, futuristic, western, among others), the game story and an initial scope of the levels, size and time that the game may require. Based on the overview information, it defines the high level properties (drivers) that the game should have in order to bring the desired UX. For each driver, it defines one or more guidelines on how to create specific game element(s) in order to fulfill the driver goal. Team members must approved guidelines.
4) **Design high level game architecture** *(product: assumptions and constraints, mechanics, dynamics of the iGDD and guidelines of the GEM).* The high level architecture reviews the technical settings to modify the assumptions and constraints. Technical settings include: the standards, conventions, technology, resources and architecture selected for the game. This activity creates a high level version of the game elements related to the guidelines.
5) **Create product backlog** *(product: overview of the iGDD).* The main features in the game listed in the overview are used to create the requirements in the product backlog.
6) **Prioritize product backlog** *(no iGDD or GEM section associated).* The team prioritize the product backlog requirements based on the value that each requirement give to the game.
7) **Organize product backlog** *(no iGDD or GEM section associated).* The product backlog lists everything that might be needed in the game, the resulting list has the requirements to be implemented in the project.
8) **Sprint planning** *(no iGDD or GEM section associated).* In this activity, the requirements with the higher priority from the backlog are selected and planned.
9) **Create sprint backlog** *(no iGDD or GEM section associated).* In this activity, each task derived from the chosen requirements is estimated and assigned to the team members as long as there is time left in the sprint.
10) **Design Game** *(products: mechanics, dynamics of the iGDD and guidelines of the GEM).* This activity designs each game element needed to fulfill the requirements to be developed on the sprint and verifies that the game elements follow the guidelines associated to them (if there is any). It also validates that designed game elements correspond to guidelines. Finally, it creates test cases to validate guidelines (if needed).
11) **Generate code and asset** *(products: mechanics, dynamics of the iGDD and guidelines of the GEM).* This activity creates game elements based on the game design and their corresponding guidelines. These elements include code and assets – e.g., music or animations.
12) **Commit and integrate** *(products: guidelines of the GEM).* This activity validates that the developed game elements follow the guidelines or gives a valid justification of why they could not follow them. It also integrates game elements in a version suitable for release.
13) **Test and tune** *(products: guidelines and test cases of the GEM).* This activity tests the resulting product of the sprint in order to verify the quality e.g., fulfill the guidelines. Small adjustments can be made to polish the game, but radical changes should be placed in the product backlog to consider them in the next sprint. The result of this activity will be a potentially shippable product.
14) **Sprint review** *(products: GDD all and GEM all).* The retrospective presents the results of the sprint: reviews of the product, process, tools, people, and any other relevant aspect of the project. The feedback given by members of the team and other stakeholders is evaluated.
15) **Project review** *(products: GDD all and GEM all).* The information of previews sprints is used to evaluate the project, if needed the team adjusts the project duration; modifies, eliminates or adds requirements in the product backlog. While there are pending requirements in the backlog go to activity 7.
16) **Do alpha test** *(products: test cases of the GEM).* Alpha test finds and removes bugs and verifies that game elements fulfill quality criteria [26], [32], [11].
17) **Do Beta test** *(products: test cases of the GEM).* This test evaluates UX.

B. Software tool to support GameD-UX

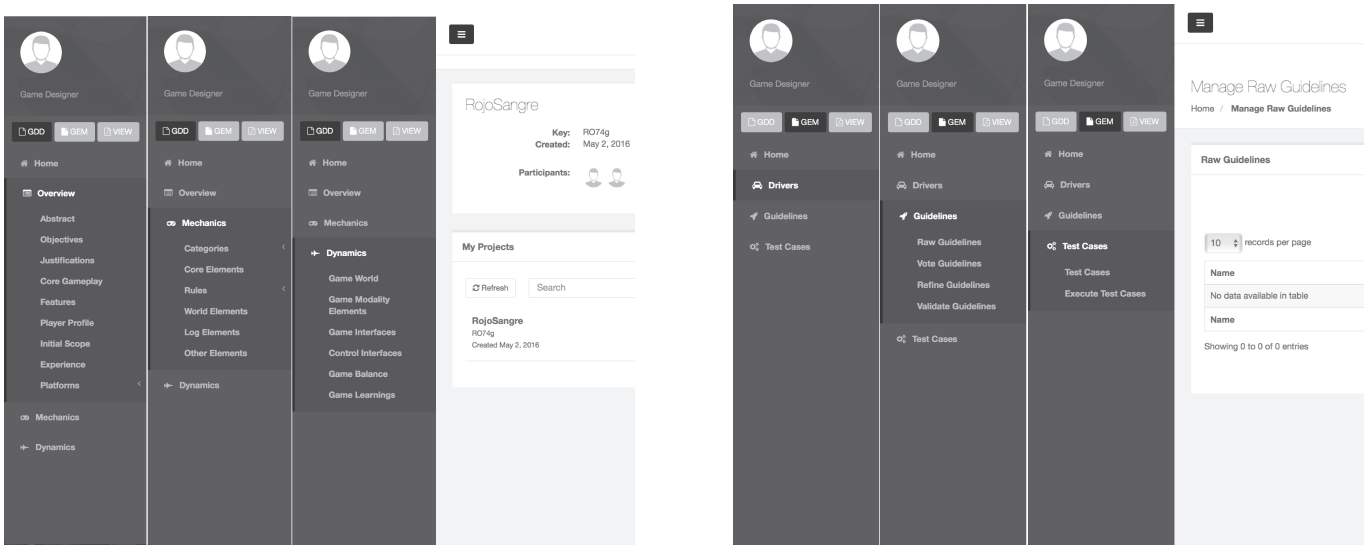
GameD-UX can help to improve the experience of the player [6] and reduce the rework [33] of the game development team. Nevertheless, in opinion of developers after the execution of the first experiment, the complexity of using iGDD and GEM together provokes low productivity. For this reason, we designed a software tool aiming to reduce the complexity of use of GameD-UX; hence, the requirements presented in Table II were defined.

The tool to support GameD-UX has two menus: the iGDD (Fig. 2a) and the GEM (Fig. 2b). The iGDD menu can create, modify or disable game categories and elements. The game designer can change the status of a category or an element. The tool enforce to follow the structure of the iGDD – e.g., if the user wants to create a ninja, it is necessary to create the enemies category.

Analogously, the GEM menus enforce to follow the GEM structure. It is necessary to have a goal in the iGDD overview to associate a driver, a driver to create a guideline, and so on.

IV. Material and Methods

Two experiments are presented in this section following the suggestions of Wohlin et al. [34]. The purpose of the first experiment is to evaluate GameD-UX in terms of rework and productivity. The second experiment is intended to evaluate the productivity and complexity of using the GameD-UX tool.



(a) (b)

Fig. 2. Software Tool: (a) menus that support the iGDD. (b) menus that support the GEM.

TABLE II. SOFTWARE TOOL REQUIREMENTS

N	Description
R_1	The tool can have information on the status of each element created in the iGDD, to know if an element is in progress (which means it can't be implemented yet) or it is already finished (which means ready to implement).
R_2	The tool can keep track of the game elements in the iGDD with the drivers and guidelines in the GEM and show with which guidelines and drivers the game elements are linked.
R_3	The tool can filter of all the information of the game elements in the iGDD, to let the team know which elements are finish (ready to implement) and which are in progress. It can also help to filter the information by role (it let know which game elements are related to the art roles or software engineering roles).
R_4	The iGDD in the GameD-UX method is a repository with a precise structure and relations, by ensuring with a tool that the game designers keep this structure and relations. Requirements can be easily linked to the game element by the method taxonomy. If the requirement involves the development of a core game element, or a challenge, these key categories can be present in the requirement description as a tag with a color associated to them.
R_5	The tool can ensure that the game elements are created correctly following the structure and inter-dependencies pre-determined. For instance, if the game designer wants to create a challenge, but there are no core game elements, the challenge will be empty until the core game elements are created and can be integrated in the challenge. Same way, if the game designer want to create a core game element but there is no category to which this must belong the element cannot be created. In sum, the constraints of iGDD elements and its relations are automated in the tool.
R_6	The tool can notify with which guidelines (GEM elements) the game elements are related and what the guideline says about the game element to be developed. This makes easy to confirm if the game element follows the guideline(s).

A. Context

Students who successfully completed the *video game development course* in the software engineering master degree program, in the Center for Research in Mathematics (CIMAT) were eligible to participate in the study. Besides the fundamental concepts of video game development, this course also focuses in Unity® as game engine and C# as programming language.

B. Participants

Eighteen junior software engineers carried out this empirical study; these engineers (hereafter, participants) had experience in Scrum. Three groups were considered:

Group A. Three development teams of two participants that use GameD-UX.

Group B. Three development teams that use the conventional approach composed by: Taylors GDD [8], and the agile game development with Scrum [26].

Group C. Three development teams that uses the GameD-UX tool.

Each team (composed of two participants) developed a single video game. The game overview can be summed as: a tower defense game for teaching basic multiplication operations to children in elementary school. The goal of a tower defense game (a special case of strategy video games) is to stop enemies from reaching a specific point on a map; for this, the player can build towers to kill enemies.

C. Metrics

a) Rework: is defined as any additional effort required for finding and fixing problems after documents and code are formally signed-off as part of configuration management [35]. For measuring the rework, any artifact put to test for the first time starts to register rework time after the test is done. To compare different products, rework effort is sometimes normalized by being calculated as a percentage of development effort [35].

b) Productivity: is the amount of requirements that a team can complete in an hour.

c) Complexity of use: of GameD-UX tool is evaluated with a post-mortem survey applied to participants (teams A and C) after finishing the project. The survey evaluates the complexity of using GameD-UX with the software tool, it contains the following assertions evaluated in a likert scale (1 strongly disagree – 7 strongly agree):

Assertion 1 (A_1): The tool or text documents easily allow to associate the guidelines (GEM) with the game elements (iGDD) to be developed.
Assertion 2 (A_2): The tool or text documents makes easy to identify which game elements (iGDD) are in progress and which are finished.
Assertion 3 (A_3): The tool or text documents facilitates the validation of game elements (iGDD) with their corresponding guidelines (GEM).

An overview of the two experiments (A, B) and their relationships to the study groups, metrics, and lesson learned are shown in Fig. 3.

D. Experiments

The experiment A was conducted to compare rework and productivity of GameD-UX and a conventional approach to game development. For this aim, projects of groups A and B are compared in terms of rework and productivity.

The experiment B evaluates the software tool developed for supporting GameD-UX in terms of productivity and the complexity of use. For this aim, projects of groups A and C are compared in terms of productivity and the post-mortem survey.

V. Results

Experiment A

As shown in Fig. 4, the normalized mean of rework for group A was 2.73%, while for group B was 11.50%. A Wilcoxon signed-rank test showed that the GameD-UX induces significantly less rework than the induced in the group B ($p < 0.01$). This proves that GameD-UX generates less rework than the conventional approach.

Concerning productivity, the mean of requirements finished in Group A was 11, the mean in Group B was 11.66; the productivity mean in Groups A was 0.22 requirements/hour and in Group B was 0.29. A Wilcoxon signed-rank test showed that GameD-UX (group A) induces significantly less productivity than the productivity induced in the group B that used the conventional approach ($p < 0.01$). This means that GameD-UX without supporting tool has the disadvantage of low productivity in comparison with the conventional approach. To illustrate this disadvantage of our method, Fig. 5 shows a boxplot of productivity distribution in both groups.

After the groups delivered their developments, a post-mortem evaluation was performed to evaluate the good and bad experiences of using the main elements of GameD-UX. The post-mortem analysis of the game developers experiences (Group A) brings us potential evidences to explain the poor productivity in our method. The resumed lessons learned of the six participants that use the iGDD and GEM are:

- It is unclear when to modify the iGDD content.
- There exists a missing link between requirements and GEM guidelines.
- It is hard to recognize elements to be developed by role of the team; e.g., artist, programmer, designer.

TABLE III. Productivity (Hours) for Groups A and C

	Requirement Media						
	R_1	R_2	R_3	R_4	R_5	R_6	Total
Group A	21.5	32	12	17	31	18	131.5
Group C	20	31	10	16	28	11	116.0

- It is hard to visually associate requirements to game elements of the iGDD.
- It is hard to recognize the correct sequence of sections to be developed.
- Developers are usually overloaded with manual validation of GEM elements.

Experiment B

Results of experiment A show that manual work (using text documents) to manage GEM, iGDD, and the links between them generates low productivity for video game development. To overcome this issue, a tool (described in section 3.2) was developed following the lessons learned.

In general, the productivity of group C (total median of 116.0 hours) is better than the productivity of group A (total median of 131.5 hours). Table III shows the results for each requirement.

The complexity evaluation was measured by applying an independent sample t-test to examine if there was a significant difference among the means of assertions answers. The boxplot shows the median results for the survey (Fig. 6). A statistical difference was observed for A_1 ($p < 0.01$) and A_2 ($p < 0.01$). It means that the supporting tool facilitates the use of GameD-UX because it easily associates guidelines with game elements and it makes easy to identify which game elements are in progress and which are finished.

Although the scale for A_3 was higher when using the supporting tool (Mean=5.56) compared to the GameD-UX (Mean=4.63), there was not a significant difference.

VI. Conclusions

This paper presents the GameD-UX method for video game development based on UX. It is composed of a repository of game elements (iGDD), a model to design, track and manage user experience (GEM), and an adapted Scrum method for game development.

The GameD-UX method induces less rework than a conventional approach used to develop video games (Taylors GDD and the agile game development with Scrum). Sections of the iGDD and their relation to the Software Requirement Specification (SRS) characteristics are key factors that improve the conventional repository.

The lessons learned from initial video game developments using GameD-UX – e.g., low productivity – were improved with the supporting tool. The requirements developed by teams that uses the GameD-UX consumes 13.33% less time when using the tool.

According to the survey applied to game developers, the complexity of use of GameD-UX was reduced with the tool. Specifically, the GameD-UX tool: (i) allows to associate the

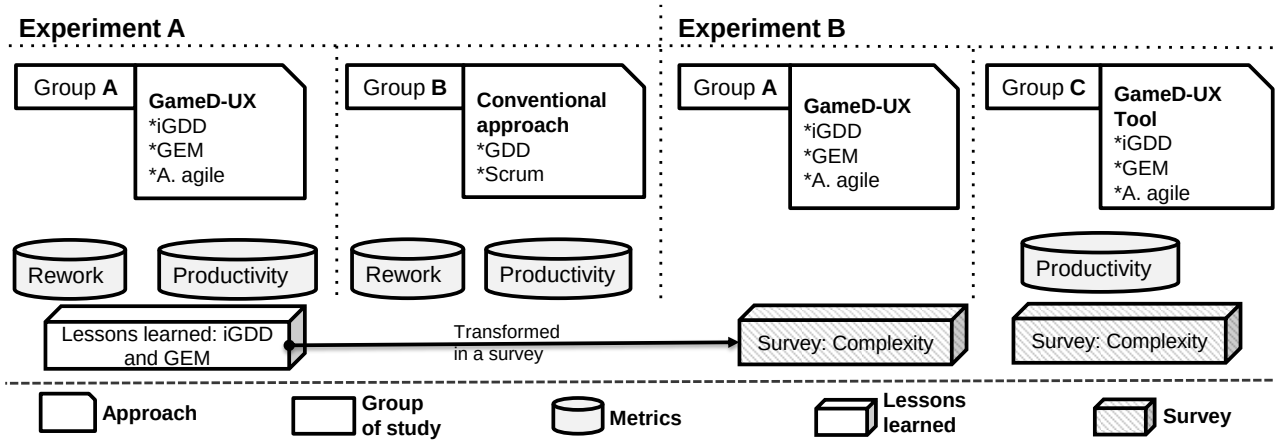


Fig. 3. Groups, metrics and instruments for experiments A and B.

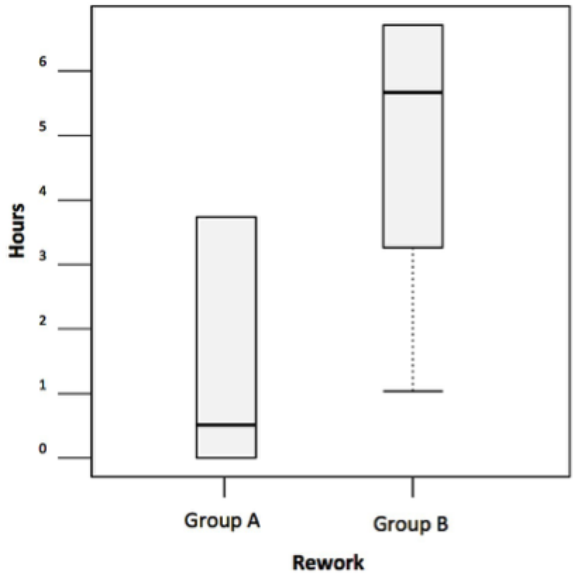


Fig. 4. Comparison of rework for groups A and B.

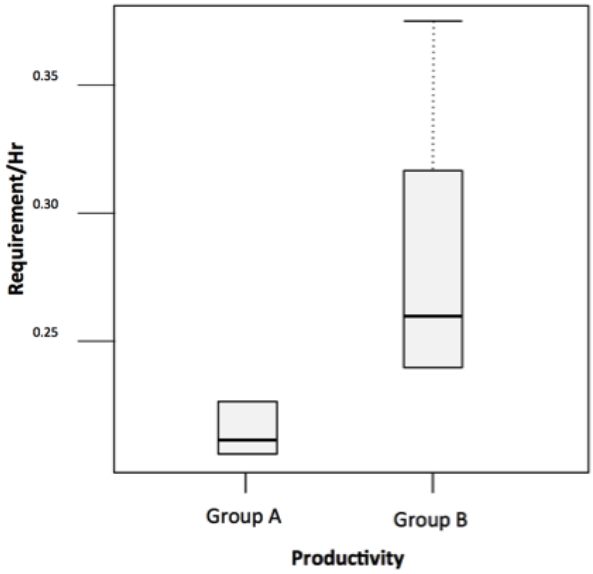


Fig. 5. Comparison of productivity for groups A and B.

guidelines (GEM) with the game elements (iGDD), and (ii) it facilitates the identification of game elements according to their status (finished or unfinished). The tool was designed to track game elements and their association to guidelines and drivers. But we believe that a better performance can be obtained by improving the tool – e.g., including real-time notifications of status changes to developers and reviewers.

In further works, we will extend the GEM to involve player-centric practices – e.g., players can help to define their profile. A more accurate understanding of the potential players, will originate more useful game design drivers to create better an experience for these players.

In affective and cognitive computing, we want to investigate the human behavior with the video game elements in order to achieve the desired emotion or cognition, and integrate it in the GEM the comparative results of diferent versions of mechanics and asthetics.

REFERENCES

[1] E. S. Association *et al.*, "Essential facts about the computer and video game industry: 2010 sales, demographic and usage data 4 (2010)," *Washington, DC. Disponível em: http://www. theesa. com/wp-content/uploads/2014/10/ESA_EF_2014. pdf. Acesso em mai*, 2016.

[2] J. Schell, *The Art of Game Design: A book of lenses*. CRC Press, 2014.

[3] F. Petrillo, M. Pimenta, F. Trindade, and C. Dietrich, "What went wrong? a survey of problems in game development," *Computers in Entertainment (CIE)*, vol. 7, no. 1, p. 13, 2009.

[4] ——, "Houston, we have a problem...: a survey of actual problems in computer games development," in *Proceedings of the 2008 ACM symposium on Applied computing*. ACM, 2008, pp. 707–711.

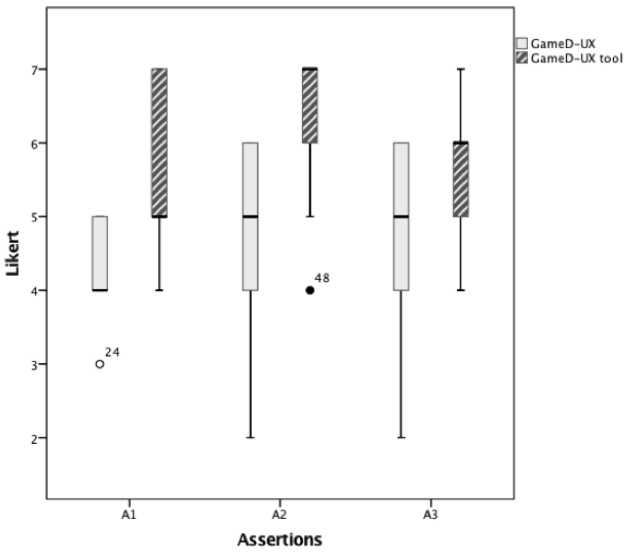


Fig. 6. Comparison of complexity of use GameD-UX between groups A (GameD-UX) and C (GameD-UX tool)

[5] M. Gonzalez-Salazar, H. A. Mitre, C. L. Olalde, and J. L. G. Sánchez, "Proposal of game design document from software engineering requirements perspective," in *Computer Games (CGAMES), 2012 17th International Conference on*. IEEE, 2012, pp. 81–85.

[6] H. Mitre-Hernandez, C. Lara-Alvarez, M. Gonzalez-Salazar, J. Mejia-Miranda, and D. Martin, "User experience management from early stages of computer game development," *International Journal of Software Engineering and Knowledge Engineering*, vol. 26, no. 08, pp. 1203–1220, 2016.

[7] E. Bethke, *Game development and production*. Wordware Publishing, Inc., 2003.

[8] C. Taylor. (1999) MS Windows NT design document. [Online]. Available: www.designersnotebook.com/ctaylordesign.zip

[9] S. Rogers, *Level Up! The guide to great video game design*. John Wiley & Sons, 2014.

[10] K. Oxland, *Gameplay and design*. Pearson Education, 2004.

[11] A. Rollings and E. Adams, *Andrew Rollings and Ernest Adams on game design*. New Riders, 2003.

[12] M. Baldwin. (2005) MS Windows NT game design document outline. [Online]. Available: http://ccasummer2014.tumblr.com/post/91982395367/baldwin-game-design-document-template-doc-file

[13] B. Bates, *Game Design [Paperback]*. Premier Press; 2nd Revised edition edition, 2004.

[14] R. Rouse III, *Game design: Theory and practice*. Jones & Bartlett Learning, 2010.

[15] L. Nacke, A. Drachen, K. Kuikkaniemi, J. Niesenhaus, H. J. Korhonen, W. M. Hoogen, K. Poels, W. A. IJsselsteijn, and Y. A. De Kort, "Playability and player experience research," in *Proceedings of DiGRA 2009: Breaking New Ground: Innovation in Games, Play, Practice and Theory*. DiGRA, 2009.

[16] E. H. Calvillo-Gámez, P. Cairns, and A. L. Cox, "Assessing the core elements of the gaming experience," in *Game User Experience Evaluation*. Springer, 2015, pp. 37–62.

[17] S. Engl and L. E. Nacke, "Contextual influences on mobile player experience–a game user experience model," *Entertainment Computing*, vol. 4, no. 1, pp. 83–91, 2013.

[18] C. Hochleitner, W. Hochleitner, C. Graf, and M. Tscheligi, "A heuristic framework for evaluating user experience in games," in *Game User Experience Evaluation*. Springer, 2015, pp. 187–206.

[19] H. P. Breivold, I. Crnkovic, and M. Larsson, "A systematic review of software architecture evolution research," *Information and Software Technology*, vol. 54, no. 1, pp. 16–40, 2012.

[20] M. R. Barbacci, R. J. Ellison, A. Lattanze, J. Stafford, C. B. Weinstock, and W. Wood, "Quality attribute workshops," 2002.

[21] R. L. Nord, W. G. Wood, and P. C. Clements, "Integrating the quality attribute workshop (qaw) and the attribute-driven design (add) method," DTIC Document, Tech. Rep., 2004.

[22] R. Hunicke, M. LeBlanc, and R. Zubek, "Mda: A formal approach to game design and game research," in *Proceedings of the AAAI Workshop on Challenges in Game AI*, vol. 4, no. 1, 2004.

[23] B. W. Boehm, "A spiral model of software development and enhancement," *Computer*, vol. 21, no. 5, pp. 61–72, 1988.

[24] K. Schwaber and M. Beedle, *Agile software development with Scrum*. Prentice Hall Upper Saddle River, 2002, vol. 1.

[25] K. Beck, *Extreme programming explained: embrace change*. addison-wesley professional, 2000.

[26] C. Keith, *Agile game development with Scrum*. Pearson Education, 2010.

[27] J. Kasurinen, R. Laine, and K. Smolander, "How applicable is iso/iec 29110 in game software development?" in *International Conference on Product Focused Software Process Improvement*. Springer, 2013, pp. 5–19.

[28] A. Godoy and E. F. Barbosa, "Game-scrum: An approach to agile game development," *Proceedings of SBGames*, pp. 292–295, 2010.

[29] R. Kortmann and C. Harteveld, "Agile game development: lessons learned from software engineering," in *Learn to Game, Game to Learn; the 40th Conference ISAGA*, 2009.

[30] C. Alexander, *The timeless way of building*. New York: Oxford University Press, 1979, vol. 1.

[31] D. Martín, J. G. Guzmán, J. Urbano, and J. Llorens, "Patterns as objects to manage knowledge in software development organizations," *Knowledge Management Research & Practice*, vol. 10, no. 3, pp. 252–274, 2012.

[32] I. van de Weerd, S. de Weerd, and S. Brinkkemper, "Developing a reference method for game production by method comparison," in *Situational method engineering: Fundamentals and experiences*. Springer, 2007, pp. 313–327.

[33] H. A. Mitre-Hernández, C. Lara-Alvarez, M. González-Salazar, and D. Martín, "Decreasing rework in video games development from a software engineering perspective," in *Trends and Applications in Software Engineering*. Springer, 2016, pp. 295–304.

[34] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering*. Springer Science & Business Media, 2012.

[35] S. Pfleeger and B. Kitchenham, "Software quality: The elusive target," *IEEE Software*, pp. 12–21, 1996.