Is requirements engineering useless in game development?

Jussi Kasurinen, Andrey Maglyas, Kari Smolander

Software Engineering and Information Management
Lappeenranta University of Technology, Finland
{jussi.kasurinen, andrey.maglyas, kari.smolander}@lut.fi

Abstract. [Context/motivation] Game development is characterized by a high level of creativity when compared to other fields of software development. Games cover a multitude of themes and genres, and represent a heterogeneous group of different products with varying requirements and business goals. **[Question/problem]** Requirements engineering (RE) should be relevant to game development, but is this true and if it is, how does game industry apply RE in practice? **[Principal ideas/Results]** We interviewed 27 software professionals in seven organizations to understand how requirements engineering is applied in game developing organizations. The results suggest that in game development business practicalities and drive for "fun" dominate the areas associated with requirements engineering. Additionally, game development organizations apply approaches and methods that are comparable to requirements engineering and requirement management, but do not consciously apply common RE practices. **[Contribution]** This paper extends our understanding of requirements engineering in video game development and contributes to the requirements engineering body of knowledge.

Keywords: game development, requirements engineering, requirements management, game design.

1 Introduction

The game industry is characterized by a high level of creativity and uncertainty [1]. Its products are directed at a mass public and they are developed for entertainment rather than for a clearly utilitarian purpose [2]. The game industry products include hits (successful and popular games) and misses (failed and unpopular games) [3] and its products differentiate horizontally, which means the creation of novel products without making them fundamentally different from other products [4]. Due to the creativity-oriented approach to the development and entertainment, the value of games and the role of requirements engineering in the game industry remain unclear. However, computer games have several features distinguishing them from other consumer products. The end products of other creative industries like fashion, music, and movies are unchangeable after the release or production, but games are similar to conventional software products that can evolve incrementally with updates. These updates may add new characters, levels, and tools to the existing game and at the same time, extend the time a user spends playing the game. Nevertheless, the game evolution cycle is often based on the game company vision rather than on the requirements collected from the users. This raises a question whether the methods, tools, and practices of requirements engineering can be applied to game development, or are game products results of chaotic creativity.

In this paper, we aim to answer to the following research questions:

(i) What is the role of requirements engineering in game development?
(ii) How does requirements engineering fit together with a high level of creativity and uncertainty of game development?
(iii) How do game companies develop prototypes of new games?

Our research team interviewed 27 game development professionals in seven game developing organizations to examine their game development processes and to understand how these organizations function. We were especially interested in understanding if requirements engineering methods were applied – or not - in game development processes.

The rest of the paper is structured as follows: Section 2 discusses related research. Section 3 introduces the research process, and Section 4 reports the results of the analysis. Section 5 discusses the results, their validity and applicability. Section 6 concludes the paper.

2 Related research

Callele at al. studied requirements engineering in the video game industry and concluded that the success of games depends on solving the communication issues between stakeholders with technical and art backgrounds, the impact of previously developed games, integration between media and technology, and the impact of non-functional requirements [5]. The role of non-functional requirements like fun, aesthetic, look and feel is especially important in games but they are difficult to manage and trace [5].

Fun, or enjoyment, has been called as the main aim of computer games [6]. This relates to the intrinsic motivation to play games but other, extrinsic, motivators such as learning can be involved in playing as well [6]. In order to be attractive and played repeatedly, digital games should deal with the emotions of a player and, as a result, games include emotional requirements. These requirements can be managed using emotional terrain maps, emotional intensity maps, and emotion timelines [7].

Another attempt to apply requirements engineering in user experience design, so-called experience requirements, were done in order to provide a mechanism for game developers to predict impressions and experiences of a player. The idea was to allow companies to apply requirements engineering techniques early in the game development, but the complexities of using this technique were greater than anticipated [8].

The creation of games is also tightly coupled with iterations and prototyping. It is usual to create several prototypes in order to meet the requirements for fun and enjoyment [8]. Prototyping takes place in the preproduction stage in order to help the game designers to find the type of game they would like to create. Kanode and Haddad discuss that requirements engineering should take place at the end of the preproduction phase when the final game idea has been identified [9]. Other researchers argue that requirements engineering practices should be introduced to the game project at the earlier stage [7, 8]. In this regard, the place of requirements engineering in the process models of game development has not been confirmed yet.

A process model is defined as "*an abstract representation of a process architecture, design, or definition*" [10]. Its goal is to improve process understanding in order to facilitate human communication, process improvement, and management [10, 11]. There is a number of process models developed for managing requirements changes [12], project management [10], and others, but a process model viewpoint has not been widely studied in the game industry, which is assumed to be dominated by the waterfall model [9].

Overall, requirements engineering in the game industry has been periodically studied by researchers in order to bring "more engineering" into creativity. However, these practices are not widely used in companies mainly because of the creative side in game companies, including designers, artists, and producers who are against bringing strict engineering approaches into their day-to-day work [7]. Therefore, we conducted this study in order to understand the place, if this place exists, of requirements engineering in game development.

3 Research method

The goal of this study was to understand and clarify how requirements engineering practices are used in game companies. It was designed as an interpretive qualitative study using the Grounded Theory research method. The Grounded Theory was chosen because it is suited well for discovering and analyzing the activities in companies within their social and organizational context [13].

Developed by Glaser and Strauss in 1967 as a pragmatic approach for conducting social science research [14], Grounded Theory is built upon two main concepts: *constant comparison* and *theoretical sampling* [15]. The idea of *constant comparison* is that every new piece of collected data is compared with other data to find similarities and differences. Therefore, data is collected and analyzed simultaneously. The concept of *theoretical sampling* represents an iterative process of theory building in which the next source of data, such as an interviewee, is selected based on the analysis of the previous samples [15].

In this study we follow the Strauss and Corbin version of grounded theory. This method relies on systematic codification and categorization process for observations [16]. It enabled us to study and understand the processes and underlying connections between different activities in a large context such as game development. The interpretation of the field study results was completed in accordance with principles derived from [17] and [18].

Table 1. Organizations participating in the study

Case	Organization size	Development team size	Release platform	Future platforms	Organizational maturity
A	Medium	Large	PC, Consoles	Handheld devices	>10 products
B	Small	Small	Handheld devices	PC	<5 products
C	Medium	Medium	Consoles, PC		>5 products
D	Small	Medium	Handheld devices		<5 products
E	Small	Small	Handheld devices		<5 products
F	Medium	Medium	PC		Making the first product
G	Small	Small	Browser-based		Making the first product

3.1 Data Collection

The initial set of companies for the interviews was selected from our research partners and then supplemented with other volunteering organizations. Our objective was to have a heterogeneous group of different target audiences, development platforms, and organizational histories. In total, we selected seven organizations representing small to medium-sized professional game developers to the sample. We applied the EU SME scale as the size measure for the companies [19]; less than 50 employees for a small organization, less than 200 for a medium and more than 200 for a large. However, since we also observed that the case organizations applied outsourcing and insourcing assets to a significant degree, we graded the development team and project sizes separately from the organizations. As it can be observed in Table 1, several case organizations (A, C and D) had larger projects than what the company size would have indicated. For example, Case A had less than one hundred own employees, but frequently developed products that had approximately three hundred contributors. In contrast, our smallest observed project had three developers and two outsourced artists.

We aimed to cover differences between organizations and therefore used the polar type selection to include cases from different target platforms and different sizes of development. Five of the seven were either recent business startups or new game development companies that have released less than five products. The other two companies were more experienced in product development and had released more than five products. The target release platforms also varied from different handheld devices (smartphones, tablets) to PCs and console systems (PlayStation, Xbox, etc.) and to browser-based games played online. Two of the seven interviewed organizations also reported that they would expand to new platforms in the future, Case A to handheld devices and Case B to PCs. All cases were commercial companies and game development was their main source of income.

The selection of interviewees was guided by our existing contacts in the studied organizations. The companies selected their most representative employees based on our short description of the interviewee roles (see Table 2). Overall, 27 interviews were conducted during the spring, summer, and fall of 2012 by seven researchers from two research laboratories. The interviews were grouped into four rounds. The goal of conducting several rounds was to gain a broader understanding of the game development practice and to identify the general factors affecting design and innovation in game development. The semi-structured interviews [20] were guided by questionnaires developed in advance by our research team. In total we developed four set of questions corresponding to the interview rounds and included questions related to development methods, quality requirements, and design processes. Before the first interview round the first set of questions was peer reviewed internally to check its sanity. Between the interview rounds some follow-up-questions were added to collect a richer data set. All of the sets of questions are available at http://www.it.lut.fi/project/SOCES/.

The interviews lasted approximately one hour and were recorded for further transcription and analysis. They were arranged with one or two participants from the case organizations with one or two researchers present.

The project managers were interviewed first to understand the software development practices in the studied companies. These interviews allowed us also to compare game companies with the observations and experiences we had from conventional software development companies [18]. The more technical second round of interviews was conducted with developers and testers. During these interviews we discussed software development and programming techniques, quality requirements, software development processes and tools. In the third round of interviews with the owners and the upper management representatives, we concentrated on the overall process of game development starting from the idea to its release to the market. During this round additional themes beyond the software development, such as marketing, innovation and financing, were collected to better understand the context in which the game industry operates. The last, fourth, round of interviews investigated the creativity aspects of game development. During this round we discussed the im-

Table 2. Interview rounds and their descriptions

Interview rounds	**Interviewee**	**Description**
Round 1: Qualitative interview with 7 organizations	Team leader or project manager	The interviewee is responsible for the management of the development of one product, or one phase of development for all products.
Round 2: Qualitative interview with 6 (+1*) organizations	Developer or tester	The interviewee was responsible for the development tasks, preferably also with the responsibilities of software testing activities.
Round 3: Qualitative interview with 7 organizations	Upper management or owner	The interviewee was from the upper management, or a business owner with an active role in the organization.
Round 4: Qualitative interview with 7 organizations	Lead designer or Art designer	The interviewee was a game designer, or managerial level person with the ability to affect the product design and selection of the implement features.

* Interview themes discussed during later rounds with other representatives of the organization

portance and impact of the creativity aspects to the final design of the developed product with game designers and managers.

3.2 Data Analysis

In grounded theory the fundamental process to analyze data and generate a theory is coding. The coding consists of three basic steps: open coding, axial coding, and selective coding [16]. Open coding is "*the interpretive process by which data are broken down analytically*" [15]. The goal of open coding is to understand what data really means, compare different pieces of data in order to find differences and similarities, and attach a conceptual label to each observation/phenomena/action. Then, the identified concepts are grouped together to form categories with subcategories, dimensions, and properties that represent a higher level of abstraction than the original data. Often the process starts with *seed categories* [33] that come from the goals of the study, the research questions, and predefined variables of interest. In this study, the selection of seed categories was guided by utilizing the concepts from research questions and included categories like *creativity, requirements engineering, game company,* and *game industry.* Overall, at the end of the open coding, we had 172 codes with 1547 observations, collected from over 1400 minutes of recordings from 27 interview sessions.

In axial coding relationships between the categories are established and tested against data. For example, the identified codes like *Design process: objectives, Test process: effect on product, Marketing: effect on product* formed a chain of evidence on how the organizations design their software, on what their actual objectives are and on what kind of impact different stakeholders and process activities have on the design work. In our data, these categories occurred repeatedly and therefore we were able to establish a connection "is related to" between them as in most organizations testing and marketing had a clear effect on the design process.

Selective coding aims at identification of the core category and relating it systematically to the other categories. The core category can be one of the existing categories or a new category that is broad enough to cover the central phenomenon and explain its relationships to other categories observed [16]. In this study, the core category was formed by abstracting the categories as none of the existing categories was considered influential enough to explain the entire phenomenon. Since the objective was to assess requirements engineering in game organizations, we collected a number of observations from business aspects, testing methods, development processes and general development process models to provide a chain of evidence. As the core category we selected an abstract category *"Requirements and change management in game development"*, which explains how requirements are handled, verified and validated during the game development process. By concentrating on this we were able to discuss the applicability of requirements engineering in game developing organizations.

4 Results

The data analysis uncovered seven categories which had relevance to requirements engineering, which led to four main observations on the applied RE activities. In addition, the analysis of how the organizations functioned gave us more insight into the existence and applicability of RE in the game industry. In the following we will introduce the categorized observations (summarized in Table 3), and then discuss the main findings. After this we introduce two stereotypical process models used in the studied organizations.

4.1 Categories

The category *Design objective* summarizes the objective that the organization has on the first design phase of a new product. We included this category to the analysis to understand the types of requirements needed to achieve these design objectives. *Marketable demo* means that the organization aims to design a version of the product, which can be used as marketing material for publishers or financiers. *Proof-of-concept* means that the organization designs and develops the first version which tests that the core features of the new product work as intended and that the concept is sound from the technological point of view.

The category *Design method* indicates the way the organization does the design work. *Vision* indicates that the organization has one or few people, who design the first version based on their initial idea. In these cases, the role of requirements engineering is limited to the vision of experts who decide what should be done. *Idea pitching* means that the organization has separate design and idea pitching events, from which the most promising candidates are examined further. *Prototypes* means that the organization starts with a very simple idea such as a theme and a genre, and examines with prototypes what sorts of functionality and content would work. *Brainstorming* indicates that the organization has design discussions in a group, trying to come up with new concepts for game products. Finally, *Drawings* means that the designers work by drawing out their ideas and by creating mock screenshots, concept art and such to give an idea on how the new product should look like. In all cases requirements were collected internally or externally through initial prototypes and collecting the feedback from their use.

The category *Changes between the first and published version* indicates roughly the amount of changes that typical game products go through from the first functional version to the final published product. *Large* indicates that there may be shifts in game genre, theme or that several core features might be added, dropped or changed during the development. *Small* indicates that the published games are mostly similar to the design version, with minimum changes on features, themes or game rules.

The category *Level of details in the first design* indicates the amount of details the organizations bring to their initial game design. *Functional prototype* indicates that the organizations design and build entirely functional proof-of-concept prototype with all the core features before starting to develop the actual product. The category *Basic gameplay elements* indicates that the organizations design most of the game content, but may not commit resources to develop anything functional such as a prototype. *Core features and concept art* means that the design consists only of core features, some early forms of rule sets and a decision on the artistic style of the game visuals.

The category *Testing on design* implies the amount of influence the testing activities in the organization have on the product design. *Large* implies that testing results may warrant large changes to the game, even dropping core features or major content or change in the genre or theme of the game. *Medium* indicates that the testing work can cause large changes to the game content and features, such as dropping content or making changes to the story, but that the main features more or less stay the same. *Small* indicates that the testing activities are mostly used to balance rules or game mechanics, and do not affect features or content to a large degree.

Similarly, *Marketing on design* indicates the power the marketing has on the product design. *High* indicates that the marketing team has the ability to dictate what sort of features the products need, change features of existing products and based on the feedback from publishers, what sort of games should be developed in the future. *Medium* indicates that the marketing team can dictate themes and core features of the games or, for example, affect the theme of the game or its visual style. *Low* means that the marketing team mostly suggests the changes that have little to no impact on the final product.

The category *Main testing objective in development* summarizes the objectives that the organization has in testing activities. *User experience* means that the organization tests the usability aspects and how "fun" the game product is to use. In terms of RE, it can be considered as collecting non-functional requirements from a target audience. As part of *usability experience, game mechanics* indicates that the testing is used to balance the internal game rules so that there are no always winning strategies. *Technical aspects* indicate that the company focuses on ensuring that everything functions technically correctly, models load correctly, effects are displayed, and that the game is stable. Minor and major issues identified during testing can lead to new functional requirements in the game engine and/or platform. In organizations where several goals are listed the goals are in the order of priority.

4.2 Findings

Finding 1: Game developers need to manage plans and product requirements, as the product may vary greatly between the first design and release.

In game development the first design may not be close to the final product. Cases A, C and F reported that there is usually a big difference in the product between the first design and the final product. In the other organizations the first design was more simplified and covered only the core features and concept art, which in many cases stayed relatively stable.

"Putting the core ideas in - that does not take that many months, but the final version always seems to take time. We have to change stuff, take things away, put new stuff in and keep doing so until everything works." – Case F, Upper management

Table 3. Categorized observations

	Case A	**B**	**C**	**D**	**E**	**F**	**G**
Design objective	Marketable demo	Proof-of-concept	Proof-of-concept	Marketable demo	Proof-of-concept	Marketable demo	Proof-of-concept
Design method	Idea pitching, prototypes, brainstorming	Vision, brain-storming	Vision, Idea pitch-ing, proto-types	Vision, drawings	Brainstorming, prototypes, drawings	Prototyping, Vision	Vision
Changes between the first and published design	Large	Small	Large	Small	Small	Large	Small
Level of details in the first design	Functional prototype	Basic gameplay elements	Functional prototype	Core features, concept art	Basic game-play elements	Core fea-tures, con-cept art	Basic gameplay elements
Testing on design	Large, able to affect fea-tures	Medium, able to affect features	Large, may cause major changes	Small, some changes possible	Large, may cause major changes	Large, may cause major changes	Large, able to affect features
Marketing on design	Low	High	Low	High	High	Medium	Medium
Main testing objective in development	User experi-ence, tech-nical aspects	User experience, technical aspects	Technical aspects, game mechanics, user expe-rience	Game mechanics, user expe-rience, technical aspects	Technical aspects, user experience	Technical aspects, Game mechanics	Game mechanics, technical aspects

"I'd say that they [first and published version] differ to some degree, but the basic idea stays the same, and the core design, is still the same." – Case B, Designer

"Our first functional prototype was quite close to the designs we had. Of course we had to make some changes during development, mostly from new ideas emerging during development." -Case E, Designer

In the game industry it is not common to collect requirements in advance. Instead, the approach of *test and tune* is widely used. The initial idea is generated and developed inside the game company and then it is tested on the target audience to identify what will be liked and what will not. However, the feedback from users is rarely systematically documented. Brainstorming, pitching, and drawing were the main instruments to get new insights on how the game should be further developed in the studied companies.

Finding 2: Game products can be changed significantly based on the feedback from marketing and testing.

In all case organizations, marketing and testing teams affected the design process to a large degree. In all organizations except in Case D, the testing phase had the possibility to affect the product design and change features in the product.

"No [the testing work] does not affect that much"…"Mostly the bigger things are decided and thought out on the early stages of development." – Case D, Manager

"You can design all for all the things you want, but the only way to know for sure [if the design works] is to test things with users." – Case E, Designer

"It [testing results] does affect and it should affect."…"Even in late stages, if we find out that something is [expletive], we do it again and again until it works." – Case C, Designer

However, the organizations which were doing large productions (Cases A and C) were also the only organizations to say that marketing has only small impact on their products. They were also the only organizations in which the design work was done to the degree of a marketable prototype, meaning that when the product is sold to the financiers or publishers, it is already relatively mature, fully designed package. In other organizations, the marketing had quite a large impact on the product design.

"We try to understand the pros and cons of our design, and assess the design from the financial point of view. After that concept design we make a proof-of-concept prototype to get the overall design"…"after the proof-of-concept prototype comes actual demo. What separates the demo and actual product is the amount of content." – Case C, Designer

"And then there is the business aspect. Obviously, [our games] have to make money." – Case B, Upper management

"The crude fact is that you have to make what sells, not what you necessarily personally like." – Case D, Upper management

"Even if our idea in business is to make great games, we still have to have enough financial perspective to get food on the table." – Case G, Designer

Finding 3: Requirement analysis is conducted mostly with user tests and usability testing.

In all examined cases, the organizations reported that testing had at least some effect on product features. In addition, all organizations except Cases F and G, one of the main test objectives was the user experience.

"Our testing is more like finding out if, for example, the controls feel appropriate. It is more like reacting to feedback [from user testing] than hunting down bugs." – Case E, Project manager

Based on these observations, organizations do requirements analysis in form of usability testing and assessment of their product features. Since the features of game products are usually associated with the user experience – "the fun factor"- analysis of the requirements is usually done with usability testing or user tests since the objective is to understand what the target demographic may want from the product.

"We sometimes make drastic changes because [the result is not considered fun]". –Case F, Designer

Finding 4: Game developers try to minimize the amount of functional requirements that should be implemented.

In Cases A, B, C, D and F the organization was using third party game engines instead of designing and implementing their own. In these organizations the decision was usually made to cut complexity and, from the viewpoint of game development, unnecessary work which could be outsourced. The technical solutions to problems such as physics modeling or 3d object manipulations were left to the third party. In this regard, these companies attempted to minimize efforts in managing functional requirements by using 3rd party components.

[Using engine] helps a lot; a game engine is a huge piece of software."…"People from our company may not have an answer, but somebody from another company may have come across the same problem, and can give the answer [via support service provided by the engine provider]". – Case F, Developer

"*If I already have things available in the engine format, I just include them from our repository. Using existing resources leaves people free [to do other things]." –* Case A, Developer

However, most of these organizations still had to do technical development. Cases E and G still have their own game engines, and Case B was recently using an own solution. In addition, Case organizations A, C, and F reported that they sometimes do extensive modifications to their third party game engines. In addition, all organizations reported that they test their product for technical aspects.

"I think that most important is that the game functions without crashing. Of course, the game content is also balanced…" Case C, Project manager

"Our leading principle is that nothing leaves the office unless QA lead has accepted it, whatever the reason." – Case A, Project manager

Based on these observations, we conclude that game companies try to minimize technical requirements in favor to non-functional ones, which are mostly related to usability and user experience. However, functional requirements cannot be fully avoided in the development of new games.

4.3 Process models

The organizations were also asked to describe how their game development processes were organized. We used the models drawn according to these descriptions to assess how requirements engineering could be applied systematically in the game development context.

Based on the models made according to the descriptions, we divided the organizations into two groups. We call these stereotypical process models as development pipelines (cases A, B, C and D, Fig. 1A) and iterative models (cases E, F and G, Fig. 1B). The division between the two models is on the expected amount and role of iteration; in pipelines the expected model is that the product matures from one main phase to another with minimal iterations, whereas in the iterative model the development is expected to return to design and requirements gathering, and the development work is interlaced between multiple phases of design and testing.

The pipeline model is a straightforward waterfall-type approach to game development. The reason why organizations applied this method was that the design was developed to a functional prototype before the contract to develop a full game was sold (Cases A and C), or that the testing work had only a low influence on the product (Case D) or that the organization applied strict phases and deadlines in their development process (Case B). The common denominator in all these organizations was that the process minimized the need for testing requirements or changing features when

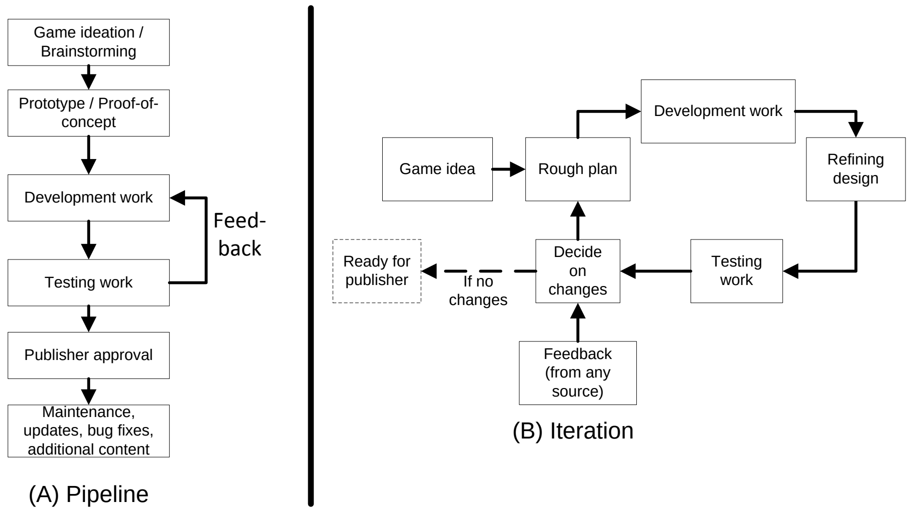

Fig. 1. Different development models adopted by organizations: pipeline approach (cases A, B, C and D) and iteration approach (cases E, F and G).

the organization committed to the development work. In most cases this behavior was related to the business aspects: "real" development started only when there was a clear demand, at least partial return of investment was ensured and the amount of required effort was relatively well-known. Interestingly, all the pipeline processes, except for Case B, considered themselves to be agile organizations, applying variations of Scrum or a similar method in daily work, but in the bigger picture doing development work with a rather plan-driven approach.

The iterative model used by case organizations E, F and G starts with a less-than-complete plan, where existing features are tested and assessed periodically and if needs arise, are changed in the next iteration. These organizations apply collected feedback extensively during development and are willing to make major changes to their product. The decision to apply iterative approach can be explained with organizational experience: Cases F and G were developing their first commercial product and this approach allowed more control to steer the product towards the intended objectives. In Case E, the organization had good experiences with "user testing"-driven development; one of their earlier products underwent drastic changes after user tests discovered a new, well-received unintended feature from the product.

5 Discussion

The in-depth investigation of development processes and discussions with the professional game developers in the companies lead us to the conclusion that the organizations may not strictly apply requirements engineering principles, but they do have process activities which can be characterized as "requirements analysis" and "requirements identification." In most organizations the development process and applied process models were rather informal (only Cases A and C had assigned roles and documented process model), and the need to do these activities originated from the practical needs related to things like testing objectives or design methods.

The development of games often starts with an idea generated through brainstorming or pitching. In this phase, the elicitation of functional and non-functional requirements is rarely done, but the analysis of fun and enjoyment plays an important role [6, 7]. In the following phases, the role of requirements engineering is important for collecting and analyzing the requirements coming from testing the game design. Callele et. al. call these requirements as experience requirements [8]. In this study, we found that the decision about continuing or stopping the development of a game can be made based on the collected experience requirements. Some designs and ideas for products may simply be rejected after proof-of-concept studies. This was also the reason why game companies develop several prototypes that are not very different from each other, and test different approaches to the same problem with focus groups. They do this in order to find the well-received solutions to components such as user interfaces and internal game mechanics. In contrast to the high attention to non-functional requirements, game companies pay less attention to functional requirements and try to minimize them by using 3rd party engines and platforms. This is especially important for small companies like Cases B, D, E, G, which lack resources to develop, maintain, and support their own game platform. For large game companies the development of an own platform or engine is also a resource-consuming activity that has little chance to provide a competitive advantage to the company because most games differentiate horizontally [4] and therefore a new product platform can rarely be fundamentally different from the existing ones.

Game development is an area where engineering meets creativity [8]. In our study, the game companies generated new game ideas mainly in-house. However, as soon as the idea is identified and the game design documents are created, most of the process deals with engineering rather than with creative activities like brainstorming new characters, levels, or story canvases. In this regard, our results are similar to the viewpoint of Kanode and Haddad who said that RE should be used when the game idea is identified and agreed to be implemented [9]. It seems to be difficult to adopt engineering practices like RE at the earlier stage as suggested by [7, 8], because it is common to generate tens of ideas initially but a few of them will be considered for detailed evaluation and implementation. Overall, in this study we did not observe a conflict between creative and engineering activities. Instead, these activities supported each other at different stages of the development cycle.

As a result of our attempt to get a deep understanding of the game development process, we sketched out two stereotypical process models of game development. We called these two models pipeline-type and iterative-type processes. The pipeline-type process in the game industry, or waterfall model [9], was already identified previously. This process is suitable for mature companies that do not heavily rely on testing their prototypes with end users and mainly produce games in-house without constant collaboration and communication with the market they aim at. This process in the game industry is very similar to the waterfall model used for development of other software products and therefore RE practices can be used in the same way, before starting developing a prototype or a proof-of-concept. The iterative-type process model relies on the feedback from customers and the role of the feedback is critical for making decisions about continuing, releasing, or withdrawing a game. In this type of process there are no predetermined requirements in the beginning, but as the feed-back is collected and analyzed, new requirements are introduced into the developed game. However, the feedback is rarely documented systematically and in the form of requirements. More often they are managed informally by changing directly the source code and testing a newer version again. In this regard, we see that there are benefits in a more formal approach to managing requirements in the iterative-type process model as it could help to decrease the number of iterations needed before the game release.

There are several threats to the validity of this study as for any qualitative study [21]. The collected observations, findings, and process models are dynamic rather than static [15] meaning that the collected data could be extended by collecting and analyzing more data. However, already this set of data that includes game companies of different size, working with different game platforms, and aiming at different mar-kets enabled us to identify the variety of approaches to the adoption of RE practices in the game industry. Our purpose was not to describe all possible ways to adopt RE in game companies. Instead, our aim was to understand if RE has its own place in game development or if it is obsolete for the industry that is dominated by insights, ideas, fun, and enjoyment. The study has also a territorial bias as we interviewed game companies in Finland only. However, due to the size of Finnish market, these compa-nies all aim at the international market, which decreases this bias.

6 Conclusion

Game developing organizations do not significantly differ from other organizations that develop software products. However, the focus on non-functional requirements and quality assurance with user testing has an impact on how game organizations operate: the game developers apply mostly informal processes, with two stereotypical approaches which in this paper were identified as waterfall-like pipeline model and iteration approach which basically is a prototype-driven incremental development model. As for findings, the game developers do need requirements management, as the products may have significant differences between the first prototypes and the final product, based on customer and market feedback. Nevertheless, RE practices are not widely adopted by game organizations. In addition, many game developers try to minimize the amount of implemented functional requirements simply by insourcing the technically challenging parts from the development process. This little focus on managing non-functional requirements and outsourcing of functional requirements lead to the situation when RE practices have not been adopted by game organizations consciously but they do apply concepts from requirements engineering, especially requirements identification and requirement management and manage risks caused by the non-functional requirements with constant prototyping and user testing. Overall with this evidence it can be argued that requirements engineering needs to be adopted by game organizations more systematically because RE methods do not contradict with creativity, but support it and provide solutions to dealing with game development in the turbulent market environment.

Acknowledgement

This study was supported by the European Union Regional Development Fund projects A31814, “Kaakon Peliklusteri” and A32139 “Game Cluster”. A part of the funding came also from the Academy of Finland grant #259454. We would also like to thank InnoVire research group at LUT Kouvola for their cooperation.

References

1. Tschang, T.: When Does an Idea Become an Innovation? The Role of Individual and Group Creativity in Videogame Design. Presented at the DRUID Conference (2003).
2. Hirsch, P.M.: Processing Fads and Fashions: An Organization-Set Analysis of Cultural Industry Systems. American Journal of Sociology. 77, 639–659 (1972).
3. Lampel, J., Shamsie, J.: Critical Push: Strategies for Creating Momentum in the Motion Picture Industry. Journal of Management. 26, 233–257 (2000).
4. Lampel, J., Lant, T., Shamsie, J.: Balancing Act: Learning from Organizing Practices in Cultural Industries. Organization Science. 11, 263–269 (2000).
5. Callele, D., Neufeld, E., Schneider, K.: Requirements engineering and the creative process in the video game industry. 13th IEEE International Conference on Requirements Engineering. pp. 240–250 (2005).
6. Draper, S.W.: Analysing fun as a candidate software requirement. Personal Technologies. 3, 117–122 (1999).
7. Callele, D., Neufeld, E., Schneider, K.: Emotional Requirements in Video Games. 14th IEEE International Conference on Requirements Engineering. pp. 299–302. IEEE (2006).
8. Callele, D., Neufeld, E., Schneider, K.: An Introduction to Experience Requirements. 18th IEEE International Conference on Requirements Engineering. pp. 395–396 (2010).
9. Kanode, C.M., Haddad, H.M.: Software Engineering Challenges in Game Development. 6th International Conference on Information Technology: New Generations. pp. 260–265. IEEE Computer Society, Washington, DC, USA (2009).
10. Feiler, P.H., Humphrey, W.S.: Software process development and enactment: concepts and definitions. 2nd International Conference on the Continuous Software Process Improvement. pp. 28 –40 (1993).
11. Curtis, B., Kellner, M.I., Over, J.: Process modeling. Communications of the ACM. 35, 75–90 (1992).
12. Harjani, D.-R., Queille, J.-P.: A process model for the maintenance of large space systems software. Presented at the The International Conference on Software Maintenance (1992).
13. Charmaz, K.: Constructing Grounded Theory: A Practical Guide through Qualitative Analysis. Sage Publications (2010).
14. Suddaby, R.: From the Editors: What Grounded Theory is Not. Academy of Management Journal. 49, 633–642 (2006).
15. Corbin, J., Strauss, A.: Grounded Theory Research: Procedures, Canons, and Evaluative Criteria. Qualitative Sociology. 13, 3–21 (1990).
16. Strauss, A., Corbin, J.: Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory. SAGE Publications (2008).

17. Paré, G., Elam, J.J.: Using case study research to build theories of IT implementation. Presented at the International conference on Information systems and qualitative research , London, UK (1997).
18. Kasurinen, J., Taipale, O., Vanhanen, J., Smolander, K.: Exploring perceived quality in software organizations. 5th International Conference on Research Challenges in Information Science (RCIS). pp. 1–12 (2011).
19. The commission of the European Communities: Commission recommendation concerning the definition of micro, small and medium-sized enterprises, (2003).
20. Robson, C.: Real World Research - A Resource for Social Scientists and Practitioner-Researchers. Blackwell Publishing, Malden (2002).
21. Onwuegbuzie, A.J., Leech, N.L.: Validity and Qualitative Research: An Oxymoron? Quality & Quantity. 41, 233–249 (2006).