Waltteri Paulaharju

ENGINEERING AN EXPERIENCE: PRE-PRODUCTION CHALLENGES IN VIDEO GAME DEVELOPMENT

JYVÄSKYLÄN YLIOPISTO
INFORMAATIOTEKNOLOGIAN TIEDEKUNTA
2020

ABSTRACT

Paulaharju, Waltteri
Engineering an experience: Pre-production challenges in video game development
Jyväskylä: University of Jyväskylä, 2020, 43 pp.
Information Systems, Bachelor's Thesis
Supervisor: Seppänen, Ville

This is a bachelor's thesis report in the University of Jyväskylä faculty of information technology. The topic of this thesis is pre-production challenges and development methods in the video game industry, and the study is conducted through a review of real-life video game development project postmortems. This thesis has two main goals: First, to analyse the postmortem material for common and significant challenges reported in the early stages of the video game development process. Second, the same material as well as previous scientific literature is used in an effort to identify practices that could be deployed in video game projects to minimize risks stemming from pre-production. This study identifies three broad categories of issues faced during and resulting from the activities of pre-production: (1) schedule, (2) specification and (3) scope. A common cause for these challenges seems to be the lack of pre-production efforts and rushing into implementation with insufficient planning, design and validation. Even though it remains debatable what kinds of development methods best mitigate pre-production issues, the data suggests that conducting more careful pre-production activities could benefit video game projects. Such activities are, among others, early prototyping and testing for idea validation as well as considering which kind of documentation would best assist in the production phase. Agile development methods could also aid in validating early ideas quickly and improve communication within the project organization.

Keywords: video game development, pre-production, issues, development methods, project management

TIIVISTELMÄ

Paulaharju, Waltteri
Kokemuksen rakentaminen: Esituotantovaiheen haasteet videopelikehityksessä
Jyväskylä: Jyväskylän yliopisto, 2020, 43 s.
Tietojärjestelmätiede, kandidaatintyö
Ohjaaja: Seppänen, Ville

Tämä raportti on Jyväskylän yliopiston Informaatioteknologian laitoksen kandidaatintutkielma. Tutkielman aiheena on esituotantovaiheen ongelmat sekä kehitysmenetelmät videopeliteollisuudessa, ja tutkimus suoritettiin analysoimalla toteutuneiden videopeliprojektien postmortem-kirjoituksia. Tutkielmalla on kaksi pääasiallista tavoitetta: Ensimmäisenä pyritään tunnistamaan yleisiä ja merkittäviä haasteita videopelikehityksen varhaisessa vaiheessa. Toiseksi samasta aineistosta sekä aiemmasta tutkimuksesta pyrittiin tunnistamaan kehitysmenetelmiä ja käytänteitä, joilla on onnistuttu vähentämään esituotantovaiheen riskejä. Tutkielmassa tunnistettiin kolme korkean tason haastetta, joita kohdataan esituotantovaiheessa tai esituotantovaiheen aikaisista toimista johtuen: (1) aikataulu, (2) määrittely sekä (3) laajuus. Tavanomainen syy näille haasteille vaikuttaisi olevan esituotantovaiheen puutteellisuus sekä kiire siirtyä nopeasti tuotantovaiheeseen huolimatta puutteellisesta suunnittelusta sekä ideoiden validoinnista. Vaikkakaan ei voida tyhjentävästi sanoa, mitkä kehitysmenetelmät johtavat parhaaseen lopputulokseen esituotantovaiheessa, tutkimusaineiston perusteella voidaan todeta, että huolellinen esituotantovaiheen aktiviteettien suorittaminen voisi vähentää esituotantovaiheesta tai sen puutteesta aiheutuvia riskejä. Tällaisia aktiviteetteja ovat muun muassa aikainen prototypointi ja testaus sekä hyödyllisestä ja kattavasta dokumentoinnista huolehtiminen. Ketterät kehitysmetodologiat saattavat osaltaan parantaa ideoiden validointia aikaisessa vaiheessa sekä tuotanto-organisaation keskistä viestintää.

Asiasanat: videopelit, videopelikehitys, esituotantovaihe, haasteet, kehitysmenetelmät, projektinhallinta

FIGURES

TABLES

TABLE OF CONTENTS

1 INTRODUCTION

This bachelor's thesis studies common challenges faced in video game development primarily during the early stages of the development cycle. Although video game development comprises of complex and difficult technical activities, this thesis focuses mainly on the project management aspects of the development. The research data is collected by analysing postmortem articles written about completed video game development projects. The postmortems provide firsthand information on the common challenges as well as successes in video game projects, and the collected data is used as the basis for this thesis.

In the past decades, video games have become an established part of entertainment culture in many parts of the world. Video games are no longer played only on bulky desktop computers or game consoles, such as PlayStation or Xbox, for we carry them in our pockets wherever we go. As various digital devices have become more ubiquitous, so have also video games become more ingrained as a choice of pastime. According to a Finnish player barometer (Kinnunen, Lilja & Mäyrä, 2018), 60.5 % of the Finnish population actively play video games, when *active* is defined as someone who plays digital games at least once a month. At the same time, the video game industry has become a considerable global market, and a report delivered by Newzoo Game Market (2020) estimates its size in 2020 to amount to almost $160 billion.

As digital technology advances and becomes more powerful and game developers are able to create more elaborate and grandiose video game environments, so also the demand for more spectacular experiences on the part of the players grows. Nowadays, ambitious video game development budgets are comparable to those of Hollywood blockbusters, reaching tens and even hundreds of millions of dollars. With bigger budgets the stakes are higher: failure will cost more and years of work could go to waste. For this reason, video game developers should seriously consider their development practices and methods to make sure that their projects will not fail on mistakes that could easily be avoided. By considering the potential risks associated with video game development, developers can make informed decisions concerning the project. This is not to say that merely reading about others' projects can enable one to make accurate estimates and predictions without experience. It can be

argued, however, that valuable insights can be gathered from the clues left behind by others who have gone through the process of creating video games.

Even though video games are works of software in the same way as other types of digital systems, they do differ in some basic elements. First, video games generally are not developed to solve a practical problem. Second, they traditionally contain a considerable number of artistic elements, such as graphics, music and story. The purpose of video games, or of any entertainment for that matter, is to induce certain kinds of experiences in the user. In the case of video games, this experience is usually called *fun* (Koster & Wright, 2004). Fun is the primary requirement of a video game, and because it is a subjective experience, it cannot be planned into a game with mere technical expertise. While video game development requires serious software engineering expertise, it is also a creative, artistic process. This means that collaboration between technical and artistic professionals is often needed to bring about a commercially successful video game. Creating a mutual understanding of the project becomes all the more vital, especially as the size of the project organization grows and the length of development increases. Effective communication is required to keep the members of the project working towards the same end.

In the context of this thesis two research questions are presented:

1) *What kind of challenges video game development organizations face during pre-production?*
2) *What kind of development practices can be used to address those issues and reduce the risks caused by early mistakes?*

The focus of this thesis is on pre-production, which, in this thesis, is defined as that phase of development during which ideas are generated, initial project plans and game designs are created, and perhaps early prototypes are constructed and feedback gathered to guide the direction of the project. Traditionally, a central game design documentation is produced during pre-production and specifications based on that documentation are created to assist in the production phase. Depending on the methodologies used by the development organisation, the decisions made during pre-production can be treated as fixed and only limited alterations are allowed in later stages, or, on the other extreme, every plan and decision can be considered as temporary and even radically alterable if testing and feedback propose a change of course.

The data gathered through postmortem analysis reveals some common themes of issues faced during early stages of production. It also gives clues about some of the practices that are used to avoid the risks of mistakes in pre-production. Considerable challenges arise due to difficulties in managing the scope of the project. As the designs of the game often go through significant changes during the production cycle, it is difficult to estimate the total amount of work that will take place over the production cycle. This is partly due to the difficulty of specifying what types of elements are required to produce the kind of experiences the designers intend. Scope and specification emerge as the two most frequently reported challenges stemming from activities in pre-production,

with the third being schedule, which is often directly affected by problems with the first two. The data in conjunction with scientific literature suggest that it is more often the lack of pre-production efforts than an excess of it that causes problems later on. In the analysed postmortems little evidence is found that for a successful project, plans that undergo no changes during development can be created. The more significant factors appear to be the extent of changes as well as the stage of development at which they take place. It is a common sentiment in project management literature that the later changes happen, the more costly they are (e.g. Boehm & Basili, 2001). In video game development, however, a drastic late change in a gameplay element can be a matter of life and death when it comes to the commercial success of the game.

The next section describes the research method of this thesis. A definition is given to postmortem articles that were used for data analysis. Also, the data extraction and categorisation process is presented. Section 3 presents the research results and in section 4 the research results are analysed and assessed in relation to existing research literature. Section 5 presents implications of this study as well as limitations of the study and further research topics are proposed. The thesis ends with a concluding summary.

2 RESEARCH METHODS

To acquire information on the challenges faced in the development of video games, an analysis and data extraction was performed on Gamasutra.com[1] postmortem articles about the successes and challenges of particular game development projects. A total of 32 articles were read and both the positive issues (PI) as well as the negative issues (NI) were recorded for data analysis. On this thesis, the word *issue* is used for both the successes and missteps of video game projects, discerned from one another by either the prefix *positive* or *negative*. A broad categorisation was created for the issues based on the data found in the articles: (1) *Pre-production*, (2) *Production*, (3) *Post-production*, (4) *Technology*, (5) *Game element*s and (6) *People*. Each individual issue was then identified more accurately and categorised under one of the six super categories. The acquired data revealed several recurring themes that game developers considered significant and worth mentioning. It points to some of the challenges faced in video game development and works as a basis for this bachelor's thesis.

2.1 Postmortem Articles

Postmortems are originally a knowledge management tool that capture a completed project's description and main problems as well as its successes (Birk, Dingsøyr & Stalhane, 2002). The Gamasutra postmortem articles are typically written by a key member or members of the video game development team in question; usually the project manager, a lead programmer or designer, who participated in the project from the beginning till the end and thus managed to form a comprehensive overview of the entire process. The articles generally follow the form outlined in Figure 1.

The articles start with a brief introduction of the produced video game and background of the project, and in some cases information on what the development company situation was like before the project started. After the introduction follows a section where (typically) five elements that the writer(s)

[1] Gamasutra.com is a website focused on video game development.

considers key successes or right choices that took place during the development project are listed under the title “What went right”. These five positive issues (PIs) can be anything from company culture to the choice of graphics engine or from scheduling to artificial intelligence programming. Next, in the section titled “What went wrong”, similarly, five negative issues (NIs) are listed that were considered failures or bad choices that had a particularly strong effect on the project. Finally, a conclusion follows, in some cases with a simple chart about the video game with data of, for instance, the duration of development, the size of the development team, budget and so on.

It is important to note that all of the selected postmortem articles were written about video game development projects that resulted in a finished product which was eventually brought to market. In addition, many of the games in the data set were developed with comparatively large budgets and published by major publishing companies. Most games in the data were (and many still are) highly popular in their genre and are generally well rated both among critics and gamers alike.

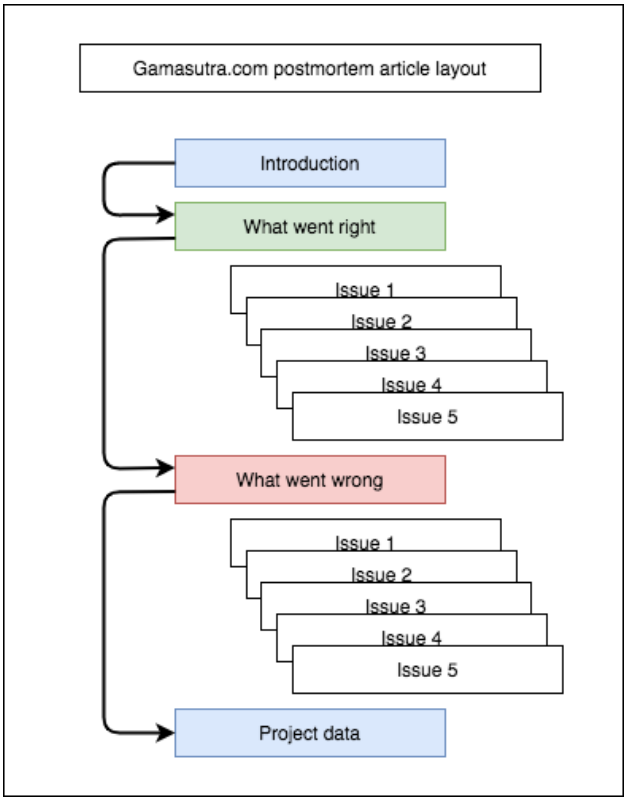


FIGURE 1 A simplified representation of the Gamasutra.com postmortem article layout

2.2 Article Analysis & Data Extraction

A search with the keywords *postmortem* and *post mortem* was conducted on the Gamasutra website. Only articles that were written in a way which clearly iden-

tifies both PIs and NIs were chosen for further investigation. From the initial 35 articles 3 were discarded due to article structures that did not clearly differentiate positive and negative issues. After the potential articles were identified, a total of 32 articles were chosen and meticulously studied. The key information concerning both the successes and the challenges of the project was saved on a text file for further study and data extraction. In most cases, a total of five successes and five failures were recorded from each article. Although in many of the cases the title of the particular issue explicitly revealed the type of the issue, further reading sometimes revealed that the actual issue was something quite different than what the title indicated. This was also the case in situations where the title of the issue was too vague to be interpreted to indicate towards a specific issue. It is also worth noting that in a few cases a single title resulted in more than one issue recorded. In unclear situations only issues that were explicitly reported under a single title were recorded.

When the issues of each article were recorded, a broad categorisation was created for the issues. These individual categories were then subjected under six super categories: *Pre-production*, *production*, *post-production*, *technology*, *game elements* and *people*. The entire category structure is presented in Table 1. After each issue under both "What went right" and "What went wrong" had been identified, a table was created for statistical analysis and interpretation.

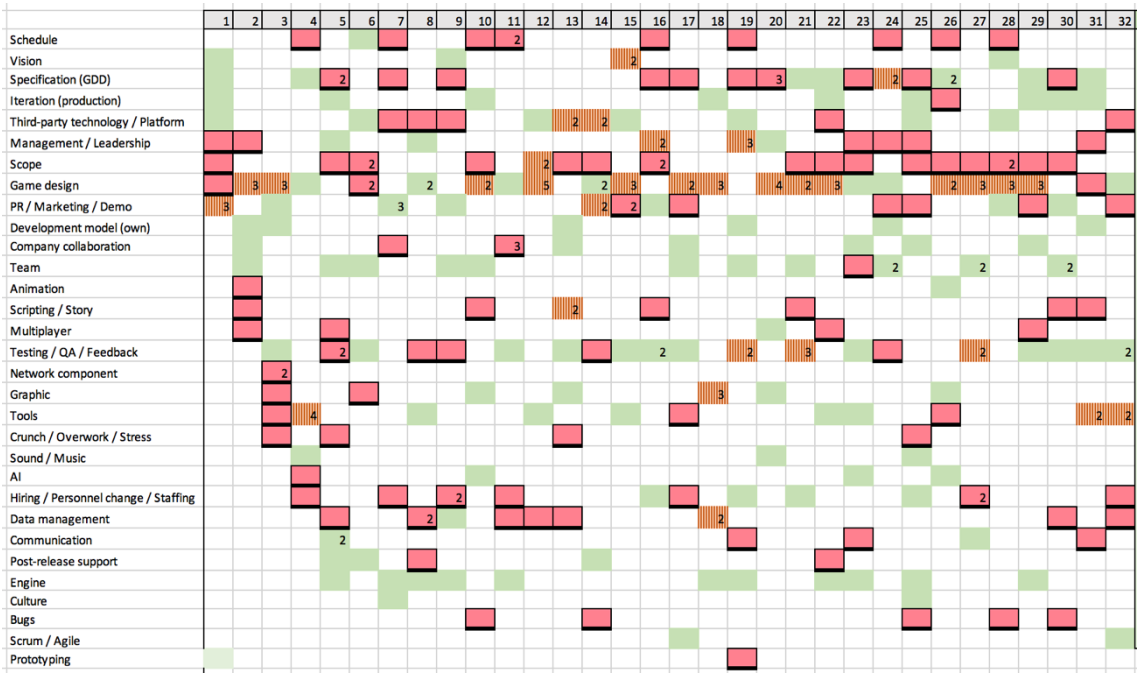
TABLE 1 Issue categorisation

Pre-production	**Production**	**Post-production**
Schedule	Iteration	PR, Marketing, Demo
Vision	Development model (own)	Post-release support
Specification	Company collaboration	Bugs
Scope	Testing, QA, Feedback	
Prototyping	Overwork, Stress	
	Data management	
	Agile development	
Technology	**Game elements**	**People**
Third-party technology	Game design	Management
Network component	Animation	Team
Tools	Scripting, Story	Staffing
Engine	Multiplayer	Communication
	Graphics	Culture
	Sound / Music	
	Artificial intelligence	

The individual, project-specific data points were recorded on a spreadsheet for easy visualisation, as presented in Table 2. The spreadsheet was constructed in tandem with the reading process of the postmortem material. On the top row the number of the postmortem article is presented, and the left column indicates

the particular issue topic. Topics were added as they gradually arose in the research material.

TABLE 2 Data records on a spreadsheet



As can be seen, a wide array of topics arose in the postmortem material. This was simultaneously a benefit as well as a challenge in creating the super-categorisation. Certain compromises had to be made in specifying which particular topics belong to which super category, so as to keep the data manageable within the context of this study. On the spreadsheet, green cells (normal fill) represent reported positive issues, red cells (borders) represent negative issues and the number on some cells indicates situations in which the particular topic is mentioned more than once within a single article. Orange cells (vertical lines) are topics that are mentioned as both a positive and a negative issue within a single postmortem article.

3 RESEARCH RESULTS

In this chapter an analysis on the research data is performed. First, an overview of the results is presented. Second, the negative issues during pre-production reported in the research data are addressed. Finally, those practices reported by game developers that are considered beneficial are analysed.

3.1 Overview of Research Results

A total of 341 issues were identified from the collected data, out of which 174 were positive issues and 167 negative issues. In the latter category, a total of 143 unique negative issues (UNI) were found, after removing those individual negative issues that were mentioned more than once within one article. If, for instance, it was stated twice but with different words within an article that there were problems with staffing, the duplicate issue was disregarded when calculating UNIs. The UNI distribution within the six super categories was following: *Pre-production* 25.9%, *production* 16%, *post-production* 10.5%, *technology* 9.8%, *people* 13.3% and *game elements* 24.5%. The distribution is visualised in figure 2.

The *pre-production* category was the most represented of the categories. The second most represented was the game elements category, which includes issue types such as game design, animation, sound and music and artificial intelligence. Naturally, as developers struggle to create a gaming experience that will entertain the audience, individual design choices can have a significant effect on how the final product turns out. But generally, game design choices cannot be unequivocally directed to either pre-production, production or post-production, even if they were clearly made during a particular phase of development, and for that reason it was decided to create a distinct category for that group of issues.

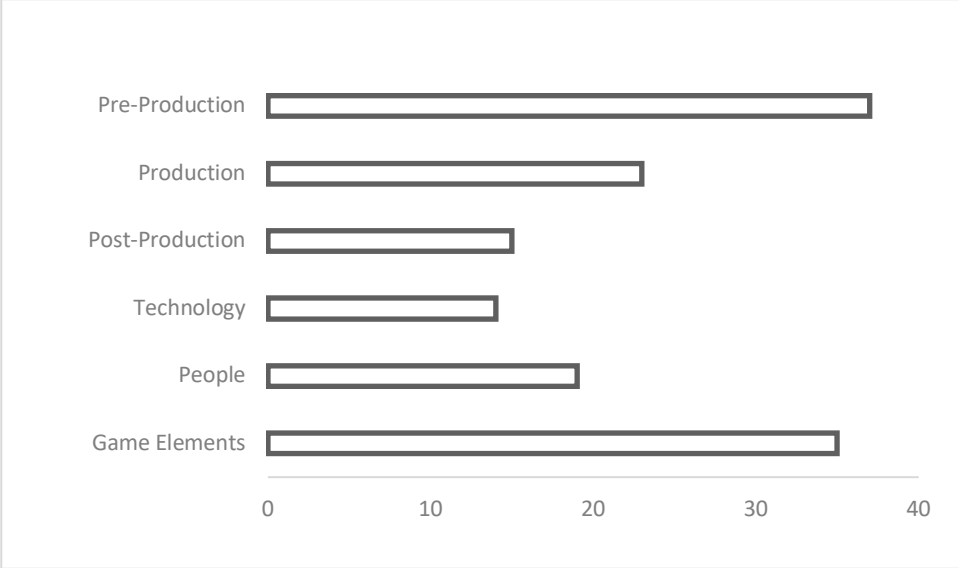


FIGURE 2 Distribution of unique negative issues

Together, the issues in *production* and *post-production* amounted to roughly that of *pre-production* alone. This might indicate that ill-advised decisions and lax preparation in the early stages of the development cycle often cause considerable obstacles later on.

Technology as a category of issues concerns the tools and technologies used in video game production, such as network components, graphics creation tools and the game engine. Of these, tools and third-party technology appeared with most UNIs: 6 and 7 respectively. The most common challenges were lacking or insufficient tools, either developed by the game development studio themselves or bought or licensed from a third party.

The *people* category includes issues concerning people-to-people interaction: management and leadership, culture, staffing and the likes. Although the amount of workforce (i.e., staffing) a development team has for a specific project is closely connected to the issue of scope, issues of hiring and staffing were directed under the category of *people* for the purpose of clarity. As is the matter with game design related issues, also issues of people-to-people interaction most often span through the entire project and are thus difficult to pinpoint to specific phases of development. The most frequent issues in this category were management and staffing, with 8 and 7 UNIs respectively. While issues concerning management and leadership were varied, with challenges ranging from unclear responsibilities to lack of leadership, issues concerning staffing and personnel were most commonly about a mismatch between the project's ambitious scope and available workforce.

3.2 Challenges in Pre-production

As the focus of this thesis is on the challenges and practices during pre-production, a more detailed analysis on the research data concerning early-stage issues follows. In the research data, a total of 167 negative issues were recorded, out of which 140 were unique negative issues (UNIs). In pre-production, a total of 37 UNIs were identified in the following categories: *schedule* (9), *specification* (12), *scope* (16). In categories *vision* and *prototyping* no UNIs were identified, and thus they are omitted from Figure 3. Categories *schedule* and *scope* were both identified with one (1) unique positive issue, whereas *specification* identified 10, *vision* 3 and *prototyping* 2 UPIs. A comparison of the three major issue categories in pre-production is presented in Figure 3. As is seen, *schedule* and *scope* were mostly considered affecting the project negatively. This could be explained by a common stance whereby project scope and schedule are considered noteworthy only when things go wrong. In a normal situation, when the project progresses as planned, scheduling and scope are taken for granted. At the same time, they are closely interlinked. Problems with scope can easily affect schedules and vice versa. *Specification* received almost equal positive and negative attention in the postmortem data. In the context of this study, issues concerning early-stage design and engineering decisions were attributed to the *specification* category.

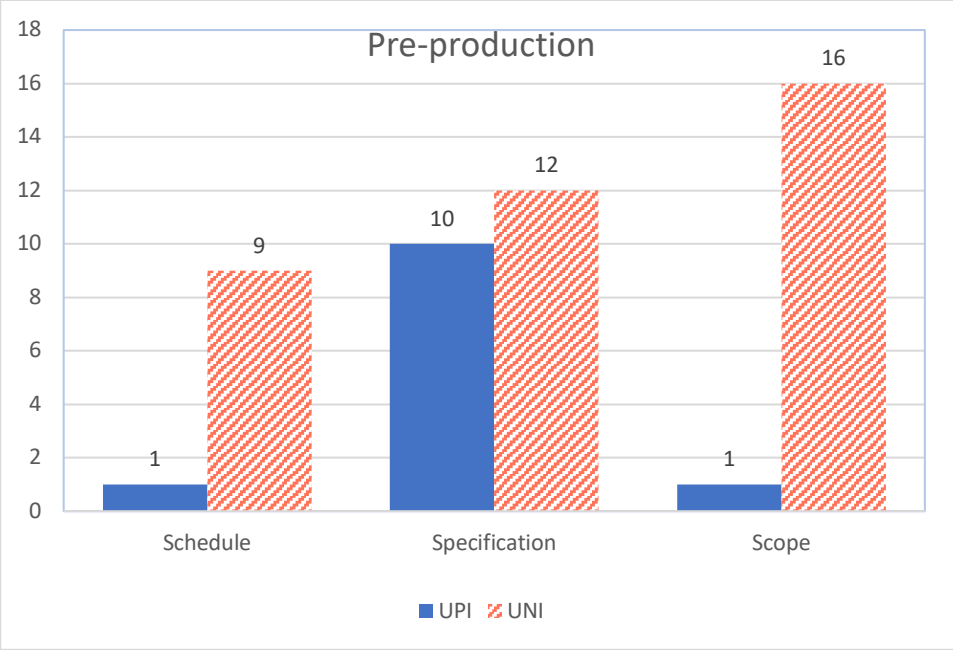


FIGURE 3 Distribution of unique positive issues (UPIs) and unique negative issues (UNIs) in pre-production

3.2.1 Schedule

In the researched postmortems, 9 video game projects reported challenges with scheduling. The problems with schedules were often due to inadequate early planning and specification in the pre-production phase and the difficulty to assess resources required in different stages of development. Also, unexpected changes in plans and designs posed, in some instances, threats to project schedules. Improper and unrealistic scheduling resulted in rushing through certain parts of development, such as testing and prototyping, as was the case with Thief (Leonard, 1998):

> […] we failed to reassess core scheduling assumptions carefully once the schedule began to slip. Captives of a series of unrealistic schedules, we didn't leave enough time for the sort of experimentation, dialogue, and prototyping a project like Thief needs. (PM4)

In video game development, the development studios are often imposed constraints on the part of the publishing company with budgets and deadlines. These cross-organisational relationships can impose also other types of constraints to the development team, as was the case with Ratchet & Clank (McCabe & Dezern, 2016), where a movie was being produced in parallel with the video game, and the changing schedules of the movie affected the game studio:

> We soon learned, however, that film release dates are far more fluid than those of games. Release windows mean everything to a film's success. […] For us, this meant more stops and starts than we're used to. We'd be geared up for a strong finish only to learn that the finish line had been moved back. […] we also learned that extra time can be as emotionally draining as not enough time. (PM11)

Scheduling can be also used as a strategic tool by planning particular release dates that could provide a competitive edge in the market. Delivering a video game just in time for the holiday season can be a valid goal for many publishers. Also, if a video game studio is able to keep track of potential competition and upcoming releases, they can strategically aim to publish before competing products. This can, however, create time pressures and thus affect production schedules. Brandon Reinhart of Unreal Tournament (1999) recounts their scheduling pressures competing with Quake 3, which was considered a major competition of their game:

> Unfortunately, the game hit shelves in November, pushing us very close to Quake 3's release date. While Unreal Tournament often performed better than Quake 3 in reviews, we believe that sales would have been much higher still had we released in October. (PM7)

For the most parts, scheduling difficulties appear to be concerned with the challenge to make accurate estimates of resource needs and resulting too tight deadlines. This is often caused by the changing specifications and unexpected events during development. Such difficulties were reported by Molyneux (2001) of

Black & White (2001), McCabe and Dezern (2016) of Ratchet & Clank (2016), Wade and Sonny (2013) of God of War: Ascension (2013), Rieke and Boon (2008) of Call of Duty: Modern Warfare (2007) and Lemarchand and Druckmann (2008) of Uncharted: Drake's Fortune (2007).

3.2.2 Specification

Specification in the context of this thesis refers to those activities and issues during the early stages of development that in one way or another specifies the video game to be produced. These specifications can concern the various designs of the game or, for example, the technical specifications used by engineers in production. A total of 12 video game projects in the research data reported negative issues with specification during development.

Specifications inform the development team about what it is that is going to be produced. The very first idea of a game can be considered a rudimentary form of specification. It usually already contains some elements of what the game will be like and how it is played, and these elements can be used to derive the technical specifications of how to create the game, whether a video game, a board game or some other type of game. As the idea is developed, more game elements are gradually added and previous ones modified or discarded, and again more detailed technical specifications can be created. All video game studios and teams probably have their unique ways of going about the specification process throughout the development cycle, but traditionally some sort of design documentation is created before implementation begins.

Challenges with specification can affect both the scope and schedule of the project, leading to the production of elements and assets that will later be discarded as redundant and unusable. A common challenge identified in the research data was inadequate or inaccurate early specification that eventually led to either an unrealistic estimate of resource needs and schedule issues or costly production of assets that eventually did not make into the final product. The balance between early planning and documentation and possibility for late-stage changes is a disputed topic in software engineering and also in video game development, and, for example, Brandon Reinhart (2000) of Unreal Tournament (1999) discusses the lack in their early design documents:

> While I am a big supporter of open, cabal-style design, I have to stop and wonder how Unreal Tournament would have turned out had we a strong initial design. It's quite possible that the game's weaker elements would have been much stronger if we had put together some concept art and focus material. (PM7)

Brian Upton (1999) of Rainbow Six (1998) recounts their experience and bemoans the lack of a game design document:

> We never had a proper design document, which meant that we generated a lot of code and art that we later had to scrap. What's worse, because we didn't have a detailed outline of what we were trying to build, we had no way to measure our progress (or lack thereof) accurately. (PM9)

While it is one thing to create plans, designs and specification documents, it is quite another thing to evaluate the resources needed to realise those plans. In software engineering and especially in video game development where every product is unique and often very much different than the one before, it is a real challenge to make accurate predictions of time and resource needs for specific tasks and systems. Whitney Wade and Chacko Sonny (2013) of God of War: Ascension (2013) write of this difficulty in their postmortem:

> What we didn't know was that every idea would feel too small upon the start. What we didn't know was how much work it would take to make the game we envisioned, and how many people would be required to successfully realize that vision. On top of that, we all didn't agree on what the game should be. (PM16)

There seems to be a tendency in video game development to rush through the pre-production phase and begin implementation of rough concepts and ideas with limited specifications. This can, however, often lead to much extra work and wasted resources. Because video games are often complex systems with countless interconnected relationships and dependencies, many of these aspects can be left unnoticed without careful planning. Also, implementing incompletely understood and defined elements can result in wasted energy and financial resources. Mike Fridley (2013) of Kingdoms of Amalur: Reckoning (2012) states that they could have benefited from a more thorough pre-production phase:

> […] our preproduction time for Reckoning itself was entirely too short. […] What we should have done was make sure we had defined everything that needed defining. We had a basic scope of content, but we hadn't done much to understand the feature set or the game's major hook. (PM17)

As video game development is often a collaboration of professionals from various fields, successful specification requires effective communication between different members and teams in the development organisation. As the development progresses into implementation after initial plans and designs are done, professionals of different backgrounds need different kinds of specification documents to be able to successfully complete their work. Artists, for instance, may need very different kind of documents than programmers do. At the same time there must be a mutual understanding of and agreement on the big picture. The work of the artist affects the work of the programmer and vice versa. For this reason, attention should be paid to communication and mutual understanding when specifications are being made. Without proper communication and guidelines for specification, development can become lopsided and resources can be allocated to less important elements. Jan Klose and Thorsten Lange (2015) of Lords of the Fallen (2014) give an example in their postmortem of unclear priorities in specification:

> […] art design decisions were made by top management without considering the whole pipeline, giving tech and game design personnel a hard time. […] The art-driven design approach resulted in being over-budget with our assets, and with the stuff being rendered on screen at once. It again meant that we could not fulfill our

> production sprints because nobody could actually integrate everything that they would have needed to. (PM19)

The complex nature of video games and their purpose of entertaining the player and inducing experiences of 'fun' makes accurate plans and estimates difficult. Countless unexpected issues can arise, and original plans can turn out to be no fun at all, which can mean the end of the entire project or a significant change of plans. Thomas Mahler (2015) writes in their postmortem of Ori and the Blind Forest (2015) about the difficulty of making plans and predicting risks and challenges:

> The problem the industry faces is that game development is almost impossible to properly plan. There is just no way of knowing exactly which issues you'll run into, how many resources exactly will be needed to finish the production, etc., especially if the production time is a rather lengthy one. (PM25)

3.2.3 Scope

Scope is closely tied to the scheduling and specification of a video game project. It is the size or magnitude of the project, which directly affects the resource requirements that are needed to complete the project within planned confines. The bigger the scope, the more there is to do and the more resources are needed, be it people working on the project, money or time. If we are, for simplicity's sake, to think of scope and specification in the context of a ship's journey from port A to port B across the sea, then scope is the length of the route and specification is the direction of the ship. These two elements influence one another. If an unnecessarily long route (scope) is chosen, the journey will cost more and take longer than needed. If, on the other hand, the captain navigates the ship even slightly off course (specification), the ship might arrive in an altogether wrong port, even in the wrong continent. This may result in considerable financial losses and other harm. Or it may turn out to be a great blessing in disguise. And the same can be true for video game development.

In the research data, a total of 16 projects reported problems with the project scope. These problems in one way or another generally have to do with questions "How much?", "How many?" and "How big?". A common problem in software projects is to decide upon how much actually needs to be done and how much this requires resources. Also, changes, big or small, throughout the development cycle all affect the project scope, usually adding to the burden of work that needs to be carried out. It may also be that original estimates can have been dramatically inaccurate, which means that work needed to complete the project is much more than initially predicted. Issues with scope can mean that much work must be abandoned to stay on target and complete the project within schedule and budget. In this regard, scope is closely related to specification: focusing on wrong or unimportant activities can consume resources from the activities that are essential to the success of the project.

A common problem, according to the research data, seems to be the mismatch between the developers' ambitions and production budget. As video

games are generally not produced to solve practical problems, they are often treated more as creative and artistic creations without strict limitations on creativity and imagination. Hence, the designers' visions can often swell and become overly ambitious. If the plans and designs are not realistic in regard to the development budget, it can lead to futile work and schedule strains. As commercial video game developers also aim to make a profit, an effort must be made to try to match the plans with the available resources, even with the risk that many dear elements must be cut along the way. Too much ambition can, therefore, hurt the development, as is stated by Michael de Plater (2015) of Middle-earth: Shadow of Mordor (2014):

> During pre-production, we didn't have good metrics on our production capacity. This led to the specification of the game being over-ambitious, which as we started to get a clearer picture of reality required us to make some pretty big and painful cuts. (PM23)

Along the same lines writes Alyssa Finley (2008) of BioShock (2007):

> Our goals and vision pretty consistently overreached our production capacity. […] Although we were able to add resources regularly and make some cuts late in the game, our ability to plan for how much work it would take to bring any single idea or space from concept to completion was poor. (PM30)

In project work, relative freedom with little constraints can blur the line between what is realistic and what is not. Without time constraints, for example, the development team can spend considerable time and financial resources on polishing details that may be relatively insignificant in regard to the player experience or keep adding content that are beyond the scope of the design plans. Warren Spector (2000) of Deus Ex (2000) writes in their postmortem of the trap of relative freedom in the development of video games:

> We started out thinking very big […] we were unrealistic, blinded by promises of complete creative freedom, and by assurances that we would be left alone to make the game of our dreams. A really big budget, no external time constraints, and a marketing budget bigger than any of us had ever had before made us soft. (PM1)

Another example of the failure to prioritise and impose constraints is provided by Ray Muzyka (2001) of Baldur's Gate II (2000), where relative freedom and lack of oversight resulted in uncontrolled feature creep:

> One of the dangers of development is that game developers have a tendency to always add content if they are given time. They don't naturally spend time limiting and polishing content; instead, more time means more stuff. It's wise to use that prioritized feature list to hone the work (of course ours was informal, which made it a little difficult). (PM5)

Estimations of the project's scope must be made early on in order to be able to gather enough human resources into the project. Assessing the project scope assists in the process of finding and hiring professionals into the team. Failure

to accurately estimate scope can lead to stress and overwork in cases where human resources do not match with the project's scope, as is reported by Jason Rieger (1998) of Myth: The Fallen Lords (1997): "[…] it became clear very early in the project that we were understaffed for such an ambitious undertaking" (PM19)

In the research data, lack of pre-production efforts and the tendency to rush into implementation was often a key reason for issues later on in development. John Garvin (2012) of Uncharted: Golden Abyss (2012) recounts their insufficient early planning:

> We completely underestimated the amount of additional content required by an Uncharted game. As we finished prototyping and began actual production, we realized this, and started making big cuts in the design and story. […] Given the chance to do it again, we would have spent much more time up front vetting every aspect of the experience, negotiating a target game length. (PM13)

There are many early decisions in video game development that can affect the scope of the project. Naturally, the game designs and specifications affect scope, but also choices concerning technology and tools can have a considerable impact. For instance, a choice regarding the game engine can have a significant impact on how much work there is. Choosing to build the engine from scratch can be a massive undertaking, but the benefit could be that a custom-built engine works best for the kind of video game that is being built. On the other hand, a licensed engine can save work, but it may not work as well with the game. Boian Spasov (2015) writes in the postmortem of Tropico 5 (2014) about the increased risks associated with starting from scratch:

> When you start on a blank sheet, you are more likely to make mistakes. Not the minor mistakes that are part of the daily life of a developer; but big costly mistakes worth months of development time. (PM22)

3.3 Practices Used to Mitigate Pre-production Risks

It is a more interpretive task to analyse the research data for practices during pre-production that could reduce mistakes and risks later on in development. The quantitative data does not directly point to specific practices that are generally and commonly found. Thus, in this section a rough overview of the research data is presented in an attempt to identify pre-production practices that are seen as beneficial by the writers of the postmortem articles.

The first theme that can be identified only by looking at the scale and type of challenges during pre-production is the lack of pre-production efforts in general. The majority of pre-production challenges seem to stem simply from insufficient planning. Because early specification and estimation and risk analysis are not done properly, problems occur with the projects' scope, schedule and direction. This would suggest that better and more careful practices during the early stages of development could benefit the overall project. Warren Spector

(2000) of Deus Ex (2000) merits a thorough pre-production as one of their five key success factors: "We didn't skimp on preproduction. We spent the first six months […] just thinking about how we could turn our high-level goals into a game" (PM1).

A key artifact of pre-production is the documentation created during that phase: the plans, designs and specifications that work as the blueprint for development. A recurring theme in the postmortem data is poor, insufficient or completely lacking documentation to guide development. This suggests that focusing on the quality and sufficiency of design documents early on could reduce the risks of uncontrollable feature creep and spending resources on game elements that are of little or no value. Lauri Hyvärinen and Joel Kinnunen (2010) of Trine (2009) attribute the lack of early documentation as a primary reason for the challenges they had with project scope:

> We neglected proper design documentation, production plans, programming plans and sometimes even art supervision. Everything happened at the last minute. In hindsight it's great to reminiscence how the team pulled together, but it certainly did not feel like the right way to do things at the time. (PM28)

Another frequent topic that arises in the research data is idea and concept validation: prototyping, testing and collecting feedback together received a total of 16 unique positive issues (UPIs) in the postmortems. Where negative issues were identified in these categories, the reason was mostly a lack of them. As Henrik Fåhraeus and Rikard Åslund (2016) of Stellaris (2016) write in their postmortem: "not doing proper prototyping is stupid, plain and simple" (PM20). Whitney Wade and Chacko Sonny (2013) of God of War: Ascension (2013) attribute the success of their game partly to "extensive playtesting, brutal feedback, constant iteration" (PM16). Mike Fridley (2013) of Kingdoms of Amalur: Reckoning (2012) also emphasizes the importance of feedback in their process: "We made sure that getting feedback from real players was high on our priority list from the very beginning" (PM17).

In the research data, only few mentions of comprehensive development methodologies are identified. Those that explicitly mention some development methodology concern agile methods, and more specifically Scrum. Both Mike Fridley (2013) of Kingdoms of Amalur: Reckoning (2012) and Caroline Esmurdoc (2010) of Brütal Legend (2009) attribute Scrum as a key success factor in their projects. Fridley writes of Scrum mostly in a positive light:

> Not only did Scrum allow us to plan better, but it also gave the entire team a lot more ownership and visibility over the game. […] Scrum allows for that day-to-day accountability that was missing for so long in game development. (PM17)

Esmurdoc writes that Scrum helped them to distill the best ideas more quickly:

> The team was practicing Scrum, and the initial payoffs were impressive. […] Scrum allowed the team to quickly test wild theories and not only keep the best ones, but also spit-shine them with continuous iteration over the entire course of development. (PM32)

Based on the data gathered from the postmortem articles, the following themes can be identified in contributing to the success of pre-production in video game development and reducing risks later on: more comprehensive and better pre-production efforts, such as concept creation, ideation, and design documentation; creating early prototypes and collecting feedback by playtesting and using that feedback to iterate the designs; considering the overall development methodologies that are used throughout the development cycle, with special attention to agile practices.

4 DISCUSSION

The results of this bachelor's thesis highlight the types of challenges and risks video game developers should take into account when beginning their development journey. On one hand, the issues identified in this study have much in common with issues in other types of IT projects. On the other hand, since video games differ from most other types of software in the sense that they are generally not developed for a utilitarian purpose, some of these challenges seem to be amplified. The creative and artistic nature of video games may limit the applicability and usefulness of some traditional software engineering practices. This seems to result in a situation where standardized processes are more difficult to be developed and more room for late changes must be allowed. At the same time, it appears that even in the creative field of video game development, doing proper planning, up-front design and early testing of ideas can reduce the risks critical challenges later on in development.

In this section the two research questions are inspected in relation to the research results. This section starts with the first research question:

> *What kind of challenges video game development organizations face during pre-production?*

The findings of different challenges identified in the research are compared to existing literature and theories in video game development and IT project management. Next, the second research question is addressed:

> *What kind of development practices can be used to address those issues and reduce the risks caused by early mistakes?*

Those themes that arise in the research data are inspected in relation to domain literature with the effort to highlight those practices that have been found beneficial in regard to the common mistakes made and challenges faced during pre-production. Finally, this section closes with short introductions to the possible implications of this study, its limitations and suggestions for further research topics.

4.1 Challenges in Video Game Development

Based on the gathered data, difficulties in the pre-production phase of video game development are common and often have significant consequences on the entire project. As circumstances in game development companies are as varied as the sparks of inspiration that eventually turn into video games enjoyed by thousands or even millions of people, so also there is great variety in how development teams go about the pre-production phase of a video game project. Yet, no matter what the size of the company or what type of the project, similar challenges can be recognised across the field. Even though the development of video games is considered in many ways different compared to the development of more traditional software, typical project management problems encumber also video game projects.

In the context of this thesis video game development is divided into three broad phases of development: pre-production, production and post-production. Pre-production, which is the main focus of this work, is the phase of development that precedes the actual implementation and creation of a given system, the production phase. In pre-production various plans and designs are made whose aims are to give a general direction for the entire project and provide a preliminary design, a blueprint to be realised in the production phase. If we consider the traditional waterfall[2] model of software engineering, Sommerville (2011, p. 31) divides the process into five distinct phases: (1) Requirements analysis and definition, (2) System and software design, (3) Implementation and unit testing, (4) Integration and system testing and (5) Operation and maintenance.

Of these, the first two would belong to the domain of pre-production, stages 3 and 4 to production and post-production would contain the period of operation and maintenance. Although many software development practitioners (Dybå & Dingsøyr, 2009) as well as video game developers (Petrillo & Pimenta, 2010; Koutonen & Leppänen, 2013) use methodologies that more resemble agile[3] practices than strictly a waterfall model of development, in most if not all cases some level of planning and design is always done before moving on to production. For this reason – and for clarity – the development lifecycle is divided into the aforementioned three main phases.

In video game development as well as in traditional software development the core goal of the pre-production phase is to specify what it is that is going to be produced. In other words: What do we want and/or need to create? The answer to this question is more clearly dictated by the pertinent problem(s) that needs to be addressed and solved in the case of, say, a customer relation-

[2] The waterfall model depicts the fundamental process phases as distinct activities that are performed in a successive order in relation to one another (Sommerville, 2011, p. 29).

[3] Agile software development is basically a set of principles that aims to improve the process of software engineering (Beck et al., 2001). The core tenets of agile development are, according to Cockburn & Highsmith, to reduce the cost of moving information between people and to reduce the elapsed time between making a decision to seeing the consequences of that decision (2001, p. 131).

ship management (CRM) system or a public transport ticketing software than in the case of a video game. In the case of a video game, the key question could be: How can we create a game that produces fun and exciting experiences to the player (Schell, 2008)? This is essentially the problem that the video game development team is going to try to solve throughout the development process, which begins in the pre-production phase of ideation, planning and design. It is the search for the 'fun factor', as Stacey and Nandhakumar (2008) put it.

The pre-production phase usually involves deciding on a schedule and budget plan as well as defining the scope of the project. These three elements of the project are strongly interlinked: by increasing the scope of a project also the budget will most certainly increase and as a result the development time (in man-months) is longer. Three main elements are generally attributed as the foundation to any project: cost, scope and time (Atkinson, 1999). These elements, that also constitute the project management triangle (Figure 4), provide the basic metrics for the evaluation of a project. In the research data, challenges with project scope were the most frequent problems, with a total of 16 UNIs.

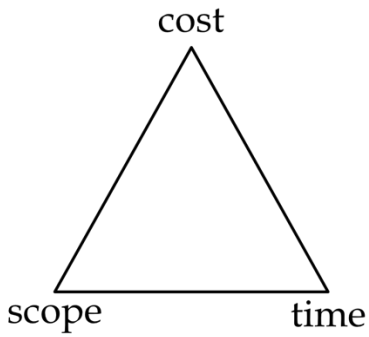


FIGURE 4 The project management triangle, drawing loosely on Atkinson (1999, p. 338)

In the collected data, project scope appeared as the most frequent issue that was mentioned as a challenge during video game development projects. A total of 16 unique negative issues concerning project scope were identified in the data, meaning that 50% of the development teams had problems with defining the scope of the project or adhering to a previously determined scope. These findings are in line with the research results of Petrillo, Pimenta, Trindade and Dietrich (2009). In their study, the two most reported problems during video game development were unrealistic or ambitious scope and feature creep, which is the unplanned addition of features during production. Feature creep often causes the project scope to inflate beyond original plans. In their data, 75% of studied video game development teams reported problems in these areas, both of which are to do with project scope.

In terms of schedule and budget, scope is essential. However, scope must be approached slightly differently when comparing traditional software systems and video games. As said, traditional systems are developed in order to provide a solution to a more or less specific problem, and in that context the

idea of scope revolves tightly around that very problem: the optimal scope is that which results in a reasonable solution to the problem. On the other hand, as the purpose of video games is more abstract, the optimal scope is more difficult to define. In other words, sometimes a simple solution brings more joy, fun and more compelling experiences to the player than a complex one. But a commercial video game studio cannot be satisfied with *just* fun. They must also consider the perspective of the potential customer, the player: Will they actually buy the game? Hence, the problem of defining a video game scope could be: *What is sufficient to produce experiences of fun at a level which gets the customer to buy the game at a price that will cover the costs of development and possibly even make a profit*?

Herein lies the challenge of scope in video game development. The task to define project scope is not relieved by the fact that video game development is a complicated process which requires the expertise of people from various different domains and where highly specified tools, technologies and techniques are used. Game designers, artists and software engineers alike collaborate in the process of creating a video game. The cross-disciplinary nature of video game development makes communication even more vital and more difficult at the same time (Whitson, 2018; Zhu & Wang, 2019). Professionals from different backgrounds may use a highly specialised vocabulary to express their viewpoint during development (O'Donnell, 2014), and this can further impede a mutual comprehension of the project's scope.

The Standish Group (1994; 2014) report considerably high figures of software projects of different scales that either fail or face significant challenges in scope and budget in their Chaos reports. Although the reliability of the figures of the Chaos reports have been strongly criticised (e.g., Jørgensen & Moløkken, 2006; Eveleens & Verhoef, 2010), there is nonetheless no doubt that many IT projects struggle with estimations of scope, budget and schedule. Thus, it is not surprising that similar problems are reported also in video game development. One would argue that the scope of a video game is (and must be allowed to be) more fluid and flexible than that of other types of software. In the search of the ephemeral 'fun factor', initial plans and designs may turn out to be completely lacking the element of fun, and the final design rather emerges gradually during different stages of development as a result of continuous playtesting and iteration (Stacey & Nandhakumar, 2008). Without proper oversight this may, however, lead to excess feature creep, with the result that the scope of the project bloats uncontrollably.

An unreliable and unrealistic estimation of project scope causes not only financial concerns, but it may also psychologically affect the individual members of the development team. A proposal is made (Callele, Neufeld & Schneider, 2005; Salazar, Mitre, Olalde & Sánchez, 2012) that game developers use the game design document produced in the pre-production phase to reduce the risk of unplanned scope expansion. Callele et al. (2005) further argue that video game developers could benefit from applying best practices from the tradition of requirements engineering. At the same time, as agile principles are increasingly being adopted in game development, arguments against such rigid and inflexible documents abound. As in most instances, the quest for the 'fun factor'

results in changes of initial designs, an original design document may quickly become outdated unless constantly revised.

The only way to find out whether a game design induces intended experiences in a player is to play it. But because video games are complex systems that often take years and millions of dollars to develop, it is impossible to comprehensively test every individual idea in a design. Here we come to the issue of specification, which is closely related to project scope, but can nonetheless be identified as a distinct problem.

For the reason that it is often rather difficult to predict which kinds of game designs induce the best experiences, specification can prove to be challenging and require considerable resources during development. A video game studio may work for months on a project which will eventually turn out to lack the degree of fun that would make a customer want to buy it. If the entire project is discarded, considerable financial losses can ensue. In addition, a publishing company whose millions were burned in a project that crash-landed might end the relationship with the studio and consequently the existence of the studio might be at risk. In the wake of such events, individual employees might experience stress and anxiety about their own employment. Lack of specification early on and the resulting delays can allow competing games to enter the market before release and thus hamper sales.

In the domain of more traditional types of software, Boehm and Basili (2001) argue that the further the development process has advanced, the more is the cost of fixing issues that have resulted from poor requirements specification. And they are not speaking of marginal numbers: They state that after release, the cost of such issues could be a hundred times more compared to fixing them at earlier stages of production. But as said, it is difficult to *plan* a blockbuster game. That is what makes specification in video game development such a problematic topic. Although most projects eventually end up changing original plans to at least a certain degree, the drawbacks of lack of early specification are obvious.

In the data set for this thesis, a total of 12 unique negative issues concerning specification were identified, which was the third biggest individual issue category after project scope and game design. Even though game design choices are at the same time specification issues, a distinction was made between issues that were clearly reported to stem from a lack of early planning and pre-production efforts. Game design issues that were not directly attributed to the pre-production phase were designated into a separate category. The data would suggest that more than third (37,5%) of video game projects face difficulties that arise due to neglected early specification and requirements elicitation. This may, in fact, be in part due to a prevalent sentiment concerning game development that 'fun' emerges organically as a result of continuous implementation, playtesting and iteration. While there may be truth in that sentiment, anyone who has ever tried to create something a little more complex without drawing even the roughest outlines on paper knows that rushing straight into implementation often results in negligence of the big picture.

As popular video games have become increasingly complex, with budgets in the tens of millions and countless interconnected parts often using novel

tools and technologies, one could argue that early specification and planning become all the more critical. Professionals with highly specialised skills are needed in the creation of video games (Callele et al., 2005; Stacey & Nandhakumar, 2008), which puts even more weight on specification: without a clear plan on what is being built, it is difficult to allocate the right amount and kind of human resources to the project.

Callele et al. (2005) concluded in their study of video game development practices that a special difficulty with specification is translating game design documentation into a form that can effectively be used by artists and software engineers during the production phase. This transformation becomes problematic because, according to Callele et al. (2005), an early game design document contains a great amount of implicit information, and it requires experience and expertise to identify and specify this information. Callele et al. (2005) describe the document transformation as a four-step process. First a game designer writes a storyline. Then a gameplay procession in the form of actions the player takes is written to match the story. Next, the in-game requirements, such as non-player characters and required items, are specified. Finally, these requirements are used to create specification documentation that can be used in technical implementation. As the documentation becomes more detailed at each step, the amount of required documentation also grows continuously. This, according to Callele et al. (2005), may lead to an unwillingness to begin the process in the first place. It also requires skill to manage this process in a way that benefits rather than hinders the project.

In addition to design choices, the video game development team must also deal with countless technical choices. In the case of console games, for instance, an individual game is often made available to several platforms simultaneously, and in the case of mobile games, to dozens of different devices and carriers. With PC games, the games must be able to run satisfactorily on many variations of components. When games are marketed globally, language and culture must also be taken into consideration early on (Alves, Ramalho & Damasceno, 2007). There are also the questions of tools and third-party technologies and game engines that need to be taken into account. These are only some of the technical factors that need to be taken into consideration at some point in development. One would argue, the earlier the better.

4.2 Development Methods to Avoid Pre-production Mistakes

Even though pre-production practices are receiving increasing amounts of attention from the scientific community (Ampatzoglou & Stamelos, 2010), there appears nonetheless to be lack of academic consensus on the best practices during the early stages of development. However, one thing became apparent from the data analysed for this study: A *lack* rather than an excess of pre-production effort is more often considered the problem. In the collected data, there is not a single mention of too much time spent in pre-production, specification and planning. As one of the most common reported scenarios

concerning pre-production problems was a flawed estimation of schedule, budget and resource requirements and the resulting expansion of scope, feature creep and lack of focus, an argument can be made that effort during pre-production should be given due value.

The arising issues in pre-production are traditional project management challenges: defining the scope, content and schedule of the project. The research data would suggest that focusing more on the preliminary planning and design during pre-production could reduce the risk of costly surprises later on. Applying traditional project and risk management practices could prove to provide the needed structural constraints to video game development.

A common theme that surfaces frequently in the postmortem articles is the quest for balance between game design documentation and the freedom to by-pass these early design specifications. Prototyping and playtesting during development can provide data that would suggest radical changes in the game design, and the development team must decide whether the benefits of the change outweigh the cost of doing the changes.

Prototyping, according to the definition by Beaudouin-Lafon and Mackay (2009, p. 1007), is creating a "concrete representation of part or all of an interactive system." They elaborate that "a prototype is a tangible artifact, not an abstract description that requires interpretation." As an example, a prototype of an envisioned video game environment could be crafted out of paper, cardboard, clay, gypsum or other convenient materials that can be easily and relatively cheaply manipulated. As another example, a non-functional user interface mockup could be created using digital visual design tools. Beaudouin-Lafon and Mackay (2009) argue that the three essential properties of prototypes, especially of the so-called offline prototypes that are not made with computers, are: (1) They are inexpensive and quick to make, (2) they are less likely (than those made with specific programming languages) to constrain a designer's thinking and (3) creating such prototypes does not require special expertise. In essence, prototypes can provide a foretaste of what the final creation could feel like with minimal cost and effort. In an industry such as video game development where the goal is to invoke certain experiences, feelings and emotions, creating prototypes early on could allow the designers to quickly and cheaply test and adjust their ideas.

Protypes help designers test their assumptions about how particular aspects and elements of the game work and feel. They allow the designer not only to see their ideas as something more tangible than mere descriptions on paper, but also to gather feedback from others, perhaps potential users, who can reveal viewpoints the designer him or herself didn't notice. It has been shown that user involvement and feedback can benefit the game design process (Kujala, Kauppinen, Lehtola & Kojo, 2005). Testing prototypes and early game designs already during pre-production can save a game studio significant financial resources compared to if defects and poor game designs are discovered when the designs have already been implemented.

A general definition of software testing (e.g., Basili & Selby, 1987) is the effort to verify whether the product is built according to the design plan and that all requirements are fulfilled. Testing should also reveal faults in the

system. Video games, however, differ somewhat from many other types of software products in that they often comprise of various artistic elements that are rarely present to the same degree in other types of software (Blow, 2004). This is emphasised by the finding in a study of seven video game companies (Kasurinen & Smolander, 2014a) where it became evident that game development is often considered, by the developers themselves, primarily as an artistic endeavor rather than a pure software engineering undertaking. In their study, Kasurinen and Smolander found evidence to the common notion that in game development, testing and the resulting changes are encouraged, even when test results propose radical changes to core gameplay elements. This goes against the attitude which is sometimes present in traditional software engineering, i.e. that late changes are a threat to the development and should be avoided due to increasing costs (e.g. Slaughter, Harter and Krishnan, 1998; Boehm & Basili, 2001).

It is not, however, enough merely to create prototypes and perform testing: The results of testing are what really matter. Developers must listen to the feedback given by testers and consider whether there is a need for changes in the game design. This listening becomes more important in video game development because, as pointed out by Kasurinen and Smolander (2014a), the most rigorously tested features in video games are difficult to quantify. Because user experience, game mechanics and usability are at the heart of video game design, proper attention must be paid to gathering and utilising feedback to find areas of improvement.

Pre-production in video game development is the stage which precedes production, where designs are turned into working code and assets. In other words, the foundation for production is created by pre-production efforts. It is also the quality of pre-production that determines whether the transition to production succeeds with ease. Callele et al. (2005) pointed out that the transition between pre-production and production often causes critical challenges for game companies. This predicament is partly caused by the failure to use the information provided by the game design document (GDD) in a way that effectively guides developers in the stage of production (Callele, Neufeld & Schneider, 2006).

The game design document is traditionally considered as the central artifact created during pre-production, which, according to Aleem, Capretz and Ahmed (2016, p. 19), "consists of a coherent description of the basic components, their interrelationships, directions, and a shared vocabulary for efficient development." The basic game design elements, according to Salazar et al. (2012), of a GDD are (1) overview, (2) mechanics, (3) dynamics, (4) aesthetics, (5) experience and (6) assumptions and constraints. Callele et al. (2005) state that Bethke (2003), on the other hand, lists story, characters, character dialogue, gameplay, aesthetics, missions, puzzles, cutscenes and animations as well as special effects as some of the essential elements of a GDD. The game design document not only captures the intended specifications and requirements of the game, but it also informs developers in implementation (Callele et al., 2006). Thus it at least implicitly defines the scope of the project and provides developers with specifications for the production phase.

While it is certainly a step forward to create game design documentation in the first place, Callele et al. (2005) state that special attention must be paid to translating the design documentation into a usable software engineering requirements specification. Translating designers' narrative and gameplay elements into requirements and specification requires special expertise and domain knowledge. For this reason, Callele et al. (2005) suggest that game development studios expend ample resources into this process and make sure that it is performed by professionals with the necessary technical and communications skills.

Kasurinen, Maglyas and Smolander (2014b) studied the use of requirements engineering processes in seven video game companies, setting out to find out whether video game companies are able to use traditional requirements engineering methods in video game production. Requirements engineering is a methodological framework that attempts to discover all explicit and implicit stakeholder needs and goals concerning a system and translate those needs and goals into a set of system requirements, both functional and non-functional (Nuseibeh & Easterbrook, 2000). Requirements engineering process generally starts with gathering information of the problem to be solved and analysing and validating this information. The bases for this information are the stakeholders of the project and their particular goals, that may or may not coincide. Numerous techniques exist that attempt to assist in this process.

Another fundamental goal of requirements engineering is to clearly document and communicate the recognised requirements and derived specifications (Nuseibeh & Easterbrook, 2000). As video games are generally 'just for fun', the stakeholder environment in video game projects is somewhat different than is the case with more traditional software projects. It could be argued that most if not all players of video games do it for the purpose of enjoyment, even though the source and sort of enjoyment within the game experience may vary greatly.

Kasurinen et al. (2014b) found support in their study to the conception that in many video game projects the difference between the original game designs and the final game are significant. A working prototype is often used as a testable artifact that is used to collect feedback. The study found that prototyping and testing are in most cases a continuous process whereby problems and new requirements, primarily non-functional concerning the gaming experience, are identified. They concluded that little evidence of systematic requirements engineering processes is found in video game development and suggest that adopting such practices could potentially benefit video game projects (Kasurinen et al., 2014b). The question from the point of view of video game companies is, whether requirements engineering practices could be used in a way that shortens development time and reduces resource requirements without undermining the quality of the games. As has been stated, there may be a considerable difference between the original design and the final video game. Video game companies often accept that even nearing release date, big changes may be necessary and even if they threaten the planned schedule and budget, these changes are implemented. Could rigorous requirements engineering methods early on enable game designers identify more of the

issues that would otherwise surface later during production, when it is often more expensive to fix them?

Whether or not traditional requirements engineering practices can ever fit the highly creative domain of video game production, it seems to be clear that capturing, communicating and managing requirements and specifications in game design is a challenge in game projects. It is often the case that video game designs are produced by professionals who do not possess enough software engineering skills to turn those designs into working code. Thus, the design team and implementation team are usually comprised of different people. Communication between designers and engineers can prove to be problematic if the designer does not fully understand the technical domain and the capabilities of the technology, and the engineer may, on the other hand, misunderstand the designer's ideas. There are several new methodologies and paradigms attempting to ease the translation of design into requirements and further into working code, such as model driven game development (Zhu & Wang, 2019) and software product lines (Furtado, Santos, Ramalho & de Almeida, 2011). While these methodologies appear still to require highly specialised expertise, another trend surfaces in the research data as well as in scientific literature: adopting agile practices in video game development.

The agile manifesto (Beck et al., 2001) lists 12 general principles that were, at the time, an attempt to work as guidance for better software development, not to say that they were any less relevant today. The main themes of these principles are (1) individuals and interactions, (2) working software, (3) customer collaboration and (4) responding to change. The agile principles could be stated as the effort to have the best possible individuals in a team interacting together and with the customer, iteratively producing working software and thus being able to respond to change at any stage of development. The people-centered approach of agile development is clearly expressed by Cockburn and Highsmith (2001, p. 131) as an effort to (1) reduce the cost of moving information between people, and (2) reduce the elapsed time between making a decision to seeing the consequences of that decision. These two goals are achieved by, for instance, having people working in the same project sit physically closer, talking person to person rather than trying to communicate through excess documentation, having highly skilled and motivated people on the team and working in increments.

Koutonen and Leppänen (2013) surveyed Finnish video game companies to find out about the development methods used in their projects. As a reference point, they quote the web-survey of 20 game studios made by Musil, Schweda, Winkler and Biffl (2010), where it was found that 61.5 % of the respondents use Scrum [4] as their principal development methodology. Koutonen and Leppänen (2013) found similar results, as more than 50 % of the surveyed game companies claimed to adhere to Scrum throughout the development cycle. Scrum's development partition loosely resembles the

[4] Scrum is an agile methodology based on flexible development sprints. The basic premise in Scrum is that software system development environment changes rapidly and the development organization must be flexible enough to adapt to the unpredictable and changing circumstances (Schwaber, 1997).

general division of the video game development process, as its three main phases, as described by Schwaber (1997), are (1) *Pregame*, which is the planning and system architecture design stage, (2) *Game*, which consists of an unpredetermined number of development sprints, and (3) *Postgame*, which is the preparation stage for release.

The study by Koutonen and Leppänen (2013) found that video game companies report several benefits of using agile methods. On the development side, the most significant benefits were better communication between professionals, the ability to more quickly find the 'fun factor' and implementable features, and the overall improved quality of the final product. On the management side, companies reported easier project management due to agile principles and techniques and some benefits were reported in scope management and schedule and budget estimation. This data seems to suggest that agile development methods introduce improvements not only to the production phase, but also to pre-production in the form of more accurate estimates and more effective specification.

In the postmortem research data, agile development methods, such as Scrum, were explicitly mentioned only twice. Both mentions regarded the agile method (Scrum) as a positive factor in game development. The reason for such scarce occurrence in the data may be partly due to the fact that video game development has already for a relatively long time utilised agile-like techniques and methods. In addition, as some studies (e.g. Stacey & Nandhakumar, 2008; Petrillo et al., 2009) conclude, it is often the case that video game companies adopt specific agile principles and techniques rather than adhering fully to a complete framework.

Based on the research data, it appears that Scrum's focus on people and the team results in increased motivation to take responsibility of one's work during development. The frequent meetings and emphasis on face-to-face communication allows developers and designers better keep track on what is going on in the project, compared to trying to keep comprehensive documents up to date and searching for information in them. The iterative nature of Scrum and other agile methodologies seem to hasten the process of finding those elements in the game design that are inducive to an enjoyable experience. Creating models and prototypes and testing them in each sprint could enable video game companies specify the final game elements earlier than if adhering to a waterfall model, thus helping to avoid big and costly late-stage changes in already-implemented game elements (although the agile principle accepts and even encourages late changes, if better solutions are discovered).

4.3 Practical and Theoretical Implications

This study shows that directing resources to proper pre-production can help video game development teams better stay within pre-determined schedules, budget and scope. Game developers should focus on testing their ideas and suppositions early on to find out which elements work and which do not in re-

gard to the intended player experience. Creating prototypes and actually letting potential users test them and provide feedback can significantly improve the end result and reduce the risk of spending great amounts of resources making something that is not fun or fixing bad choices late in development. As designs and other documentation are being created in preparation for implementation, special attention should be paid to the usefulness of these documents. Professionals with different backgrounds and skills work together in video game projects, and thus it is necessary that there is mutual understanding and that documentations can actually be used when production begins. To improve communication and management, it may be beneficial to consider development frameworks that have been proved useful in this regard. Agile methodologies, for instance, are reported to improve timely information flow and effective communication within the organization.

4.4 Limitations of the Study

One of the main limitations of this study is the fact that the research data consists only of video game projects that resulted in a finished project. And not only that: many of those games were not only published but became some of the most successful video games in their genre. This means that a large group of video game projects was left without representation in this study. Video games that were published but managed to gain only mediocre publicity and projects that were started but were never finished are not represented in the data. From the perspective of an aspiring video game developer, however, an alleviating factor in this study's limitations is the presumption that most such developers want to create games that do become successful. In this regard, it might make sense to study those projects that succeeded rather than those that failed, although a very different type of issue palette might concern the more unsuccessful projects.

This study also focused on pre-production challenges from a relatively broad perspective. The issue categories of specification and scope, although closely related, can contain within them a broad array of minor issues that were left unidentified in this study. For example, issues in specification can arise from various different problems, such as the failure to communicate design plans to the programming team or the lack of a common understanding of how the game should be. For this reason, more research could be conducted, especially concerning pre-production challenges in more detail.

4.5 Topics for Further Research

Further research on prototyping early on could provide interesting insight into how quickly successful video game projects decide upon the elements that are found in the final product. Also, the transition between pre-production and

production and the relevant documentation deserves further inspection, especially because the amount and form of documentation seems to be under debate in the video game industry. It may be, however, that the most successful game companies are not willing to disclose the details of their particular practices in order to maintain a competitive advantage.

5 CONCLUSION

This thesis set out to study common challenges faced and beneficial development methods used in video game development, with a focus on activities in the early stages of development, or pre-production. The research questions of this thesis were:

1) *What kind of challenges video game development organizations face during pre-production?*
2) *What kind of development practices can be used to address those issues and reduce the risks caused by early mistakes?*

Pre-production was defined within this thesis as that period in the development process where first ideas are turned into concepts and designs and where planning is conducted in preparation for implementation. Three broad categories of pre-production issues were identified in the study: (1) schedule, (2) specification and (3) scope. These are traditional project management challenges, and the complex field of video game development often makes dealing with them all the more difficult. Because video game development generally focuses on the search of the 'fun factor' and creating certain kinds of experiences to the players, it is often impossible to make unchanging, comprehensive plans and designs that require no changes during later stages of development. It was often this difficulty to plan ahead that caused a great deal of the challenges in the three categories.

There are, however, ways that video game developers can reduce the risks of uncontrollably expanding scope, failing schedule or insufficient specification. One of these is to spend more resources for pre-production activities, such as concept creation, design, prototyping, testing and making useful documents to aid implementation. Also, development methodologies that support quick testing and feedback were found useful in the research data as well as in previous literature.

REFERENCES

Aleem, S., Capretz, L. F., & Ahmed, F. (2016). Game development software engineering process life cycle: a systematic review. *Journal of Software Engineering Research and Development*, *4*(1), 6.

Alves, C., Ramalho, G., & Damasceno, A. (2007). Challenges in requirements engineering for mobile games development: The meantime case study. In *15th IEEE International Requirements Engineering Conference (RE 2007)*(pp. 275-280).

Ampatzoglou, A., & Stamelos, I. (2010). Software engineering research for computer games: A systematic review. *Information and Software Technology*, *52*(9), 888-901.

Atkinson, R. (1999). Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria. *International journal of project management*, *17*(6), 337-342.

Basili, V. R., & Selby, R. W. (1987). Comparing the effectiveness of software testing strategies. *IEEE transactions on software engineering*, (12), 1278-1296.

Beaudouin-Lafon, M., & Mackay, W. E. (2009). Prototyping tools and techniques. In *Human-Computer Interaction* (pp. 137-160). CRC Press.

Beck K., Beedle M., Van Bennekum A., Cockburn A., Cunningham W., Fowler M., Grenning J., Highsmith J., Hunt A., Jeffries R., Kern J. (2001). *The agile manifesto*.

Bethke, E. (2003). *Game Development and Production*. Wordware Publishing, Inc.

Birk, A., Dingsøyr, T., & Stalhane, T. (2002). Postmortem: Never leave a project without it. *IEEE software*, 19(3), 43-45.

Blow, J. (2004). Game Development: Harder Than You Think, *Queue*, vol. 1, nro. 10, ss. 28–37.

Boehm, B., & Basili, V. R. (2001). Software defect reduction Top 10 list. *Computer*, vol. 34, no. 1, pp. 135-137, doi: 10.1109/2.962984.

Callele, D., Neufeld, E., & Schneider, K. (2005). Requirements engineering and the creative process in the video game industry, in Requirements Engineering, 2005. *Proceedings. 13th IEEE International Conference on Requirements Engineering*, pp. 240–250.

Callele, D., Neufeld, E., & Schneider, K. (2006). Emotional requirements in video games. In *14th IEEE International Requirements Engineering Conference (RE'06)* (pp. 299-302).

Cockburn, A., & Highsmith, J. (2001). Agile software development, the people factor. *Computer*, *34*(11), 131-133.

Dybå, T. & Dingsøyr, T. (2009). What Do We Know about Agile Software Development?,"in *IEEE Software,* vol. 26, no. 5, pp. 6-9, Sept.-Oct. 2009, doi: 10.1109/MS.2009.145.

Eveleens, J. L., & Verhoef, C. (2010). The rise and fall of the chaos report figures. *IEEE software*, *27*(1), 30.

Furtado, A. W., Santos, A. L., Ramalho, G. L., & de Almeida, E. S. (2011). Improving digital game development with software product lines. *IEEE software*, *28*(5), 30-37.

Jørgensen, M., & Moløkken, K. (2006). How large are software cost overruns? A Review of the 1994 Chaos report. *Information and Software Technology*, 48:297–301.

Kasurinen, J., Maglyas, A., & Smolander, K. (2014b). Is requirements engineering useless in game development?. In *International Working Conference on Requirements Engineering: Foundation for Software Quality* (pp. 1-16). Springer, Cham.

Kasurinen, J., & Smolander, K. (2014a). What do game developers test in their products?. In *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement* (pp. 1-10)

Koster, R., & Wright, W. (2004). *A Theory of Fun for Game Design*. Paraglyph Press.

Koutonen, J., & Leppänen, M. (2013). How are agile methods and practices deployed in video game development? A survey into Finnish game studios. In *International Conference on Agile Software Development* (pp. 135-149). Springer, Berlin, Heidelberg.

Kinnunen, J., Lilja, P., & Mäyrä, F. (2018). Pelaajabarometri 2018: Monimuotoistuva mobiilipelaaminen.

Kujala, S, Kauppinen, M., Lehtola, L. & Kojo, T. (2005). "The Role of User Involvement in Requirements Quality and Project Success". *13th IEEE International Conference on Requirements Engineering*.

Musil, J., Schweda, A., Winkler, D. & Biffl, S. (2010). A survey on a state of the practice in video game development, Report IFS-QSE 10/04. *Institute of Software Technology and Interactive Systems*, Vienna.

Newzoo Game Market Research (2020). Newzoo Global Games Market Report 2020 Light Version. Retrieved 18.9.2020 at: *https://newzoo.com/insights/trend-reports/newzoo-global-games-market-report-2020-light-version/.*

Nuseibeh, B., and Easterbrook, S. (2000). Requirements Engineering: a Roadmap, *Proceedings of International Conference on Software Engineering,* 35-46.

O'Donnell, C. (2014). *Developer's dilemma: The secret world of videogame creators*. MIT press.

Petrillo, F., Pimenta, M., Trindade, F., & Dietrich, C. (2009). What went wrong? A survey of problems in game development. *Computers in Entertainment (CIE)*, *7*(1), 1-22.

Petrillo, F., & Pimenta, M. (2010). Is agility out there? Agile practices in game development. In *Proceedings of the 28th ACM International Conference on Design of Communication* (pp. 9-15).

Salazar, M. G., Mitre, H. A., Olalde, C. L., & Sánchez, J. L. G. (2012). Proposal of Game Design Document from software engineering requirements perspective. In *2012 17th International Conference on Computer Games (CGAMES)*(pp. 81-85).

Schell, J. (2008). *The Art of Game Design: A book of lenses*. CRC press.

Schwaber, K. (1997). Scrum development process. In *Business object design and implementation* (pp. 117-134). Springer, London.

Slaughter, S.A., Harter, D.E. and Krishnan, M.S. (1998). Evaluating the cost of software quality, *Communications of the ACM*, Vol. 41, Issue 8.

Sommerville, I. (2011). *Software engineering 9th Edition*. *ISBN-10*, *137035152*, 18.

Stacey, P., & Nandhakumar, J. (2008). Opening up to agile games development. *Communications of the ACM*, *51*(12), 143-146.

The Standish Group International Inc. *Chaos. Technical report*, The Standish Group International Inc., 1994.

The Standish Group International Inc. *The Chaos Report: 21st anniversary edition*, The Standish Group International Inc., 2014.

Whitson, J. R. (2018). Voodoo software and boundary objects in game development: How developers collaborate and conflict with game engines and art tools. *New media & society*, *20*(7), 2315-2332.

Zhu, M., & Wang, A. I. (2019). Model-driven Game Development: A Literature Review. *ACM Computing Surveys (CSUR)*, *52*(6), 1-32.

APPENDIX 1 POSTMORTEM DATA

	Game	Developer	Publisher(s)	Release
1	Deus Ex	Ion Storm	Eidos Interactive	2000
2	System Shock 2	Irrational Games	Electronic Arts	1999
3	Diablo 2	Blizzard	Blizzard Entertainment	2000
4	Thief: The Dark Project	Looking Glass	Eidos Interactive	1998
5	Baldur's Gate 2	BioWare	Black Isle Studios, Interplay Entertainment	2000
6	Myth: The Fallen Lords	Bungie	Bungie, Eidos Interactive	1997
7	Unreal Tournament	Epic Games	GT Interactive	1999
8	Age of Empires 2	Ensemble Studio	Microsoft, Konami	1999
9	Rainbow Six	Red Storm Ent.	Ubisoft	1998
10	Black & White	Lionhead Studios	Electronic Arts	2001
11	Ratchet & Clank	Insomniac Games	Sony Computer Entertainment	2016
12	SpaceChem	Zachtronic Industries	Zachtronic Industries	2011
13	Uncharted: Golden Abyss	Sony Bend Studio	Sony Computer Entertainment	2012
14	The Binding of Isaac	McMillen and Himsl	Edmund McMillen	2011
15	Pixeljunk 4am	Q-Games	Q-Games, Sony Computer Entertainment	2012
16	God of War: Ascension	Sony Santa Monica	Sony Computer Entertainment	2013
17	Kingdoms of Amalur: Reck.	Big Huge Games	38 Studios, Electronic Arts, THQ Nordic	2012
18	Resident Evil 4	Capcom	Capcom	2005
19	Lords of the Fallen	Deck13	Bandai Namco Games	2014
20	Stellaris	Paradox Studio	Paradox Interactive	2016
21	Persona 4	Atlus	Atlus, Square Enix, Ubisoft	2008
22	Tropico 5	Haemimont Games	Kalypso Media, Square Enix	2014
23	Middle-earth: Shadow of Mordor	Monolith Prod.	Warren Bros. Interactive Entertainment	2014
24	CoD 4: Modern Warfare	Infinity Ward	Activision	2007
25	Ori and the Blind Forest	Moon Studios	Microsoft Studios	2015
26	Uncharted: Drake's Fortune	Naughty Dog	Sony Computer Entertainment	2007
27	Civilization V	Firaxis	2K Games, Aspyr	2010
28	Trine	Frozenbyte	Nobilis	2009
29	Bulletstorm	People Can Fly	Electronic Arts	2011
30	BioShock	Irrational Games / 2K	2K Games	2007
31	Age of Mythology	Ensemble Studio	Microsoft Game Studios	2002
32	Brutal Legend	Double Fine	Electronic Arts, Double Fine Prod	2009

APPENDIX 2 POSTMORTEM AUTHORS

	Game	Postmortem Author(s)
1	Deus Ex	Warren Spector
2	System Shock 2	Jonathan They
3	Diablo 2	Erich Schaefer
4	Thief: The Dark Project	Tom Leonard
5	Baldur's Gate 2	Ray Muzyka
6	Myth: The Fallen Lords	Jason Regier
7	Unreal Tournament	Brandon Reinhart
8	Age of Empires 2	Matthew Pritchard
9	Rainbow Six	Brian Upton
10	Black & White	Peter Molyneux
11	Ratchet & Clank	Shaun McCabe, Chad Dezern
12	SpaceChem	Zach Barth
13	Uncharted: Golden Abyss	John Garvin, Jeff Ross, Francois Gilbert, Chris Reese
14	The Binding of Isaac	Edmund McMillen
15	Pixeljunk 4am	Rowan Parker
16	God of War: Ascension	Whitney Wade, Chacko Sonny
17	Kingdoms of Amalur: Reck.	Mike Fridley
18	Resident Evil 4	Yoshiaki Hirabayashi
19	Lords of the Fallen	Jan Klose, Thorsten Lange
20	Stellaris	Henrik Fåhraeus, Rikard Åslund
21	Persona 4	Persona 4 Team
22	Tropico 5	Boian Spasov
23	Middle-earth: Shadow of Mordor	Michael de Plater
24	CoD 4: Modern Warfare	Zied Rieke, Michael Boon
25	Ori and the Blind Forest	Thomas Mahler
26	Uncharted: Drake's Fortune	Richard Lemarchand, Neil Druckmann
27	Civilization V	Dennis Shirk
28	Trine	Lauri Hyvärinen, Joel Kinnunen
29	Bulletstorm	Adrian Chmielarz
30	BioShock	Alyssa Finley
31	Age of Mythology	Ian Fischer, Greg Street
32	Brutal Legend	Caroline Esmurdoc

APPENDIX 3 POSTMORTEM SOURCE

	Postmortem web address
1	https://www.gamasutra.com/view/feature/131523/postmortem_ion_storms_deus_ex.php
2	https://www.gamasutra.com/view/feature/131813/postmortem_irrational_games_.php
3	https://www.gamasutra.com/view/feature/131533/postmortem_blizzards_diablo_ii.php
4	https://www.gamasutra.com/view/feature/131762/postmortem_thief_the_dark_project.php
5	https://www.gamasutra.com/view/feature/131493/baldurs_gate_ii_the_anatomy_of_a_.php
6	https://www.gamasutra.com/view/feature/131691/postmortem_bungies_myth_the_.php
7	https://www.gamasutra.com/view/feature/131569/postmortem_epic_games_unreal_.php?page=1
8	https://www.gamasutra.com/view/feature/131844/postmortem_ensemble_studios_age_.php
9	https://www.gamasutra.com/view/feature/131829/postmortem_redstorms_rainbow_six.php
10	https://www.gamasutra.com/view/feature/131476/postmortem_lionhead_studios_.php
11	https://gamasutra.com/view/news/272694/Ratchet__Clank_2016_postmortem.php
12	https://www.gamasutra.com/view/feature/172250/postmortem_zachtronics_.php
13	https://www.gamasutra.com/view/feature/181082/postmortem_sony_bend_studios_.php
14	https://www.gamasutra.com/view/feature/182380/postmortem_mcmillen_and_himsls_.php
15	https://www.gamasutra.com/view/feature/189359/postmortem_qgames_pixeljunk_4am.php
16	https://www.gamasutra.com/view/feature/193695/postmortem__sony_santa_monicas_.php?page=1
17	https://www.gamasutra.com/view/feature/197269/postmortem_kingdoms_of_amalur_.php
18	https://www.gamasutra.com/view/feature/195143/postmortem_resident_evil_4.php?page=1
19	https://www.gamasutra.com/view/news/236275/Postmortem_Deck13_Interactives_Lords_of_the_Fallen.php
20	https://gamasutra.com/view/news/274018/Postmortem_Paradox_Development_Studios_Stellaris.php
21	https://www.gamasutra.com/view/news/116413/Exclusive_Behind_The_Scenes_Of_Atlus_Persona_4.php
22	https://www.gamasutra.com/view/news/237667/ Starting_From_Scratch_Haemimont_Games_Tropico_5_postmortem.php
23	https://www.gamasutra.com/view/news/234421/ Postmortem_Monolith_Productions_Middleearth_Shadow_of_Mordor.php
24	https://www.gamasutra.com/view/news/258315/The_making_of_Call_of_Duty_4_Modern_Warfare.php
25	https://www.gamasutra.com/view/news/242530/ Postmortem_Moon_Studios_heartfelt_Ori_and_the_Blind_Forest.php
26	https://www.gamasutra.com/view/feature/132203/postmortem_naughty_dogs_.php
27	https://www.gamasutra.com/view/news/306040/Classic_Postmortem_Firaxis_Civilization_V.php
28	https://www.gamasutra.com/view/feature/134198/postmortem_frozenbytes_trine.php
29	https://www.gamasutra.com/view/news/266261/Classic_Postmortem_People_Can_Flys_Bulletstorm.php
30	https://www.gamasutra.com/view/feature/132168/postmortem_2k_boston2k_.php
31	https://www.gamasutra.com/view/news/308555/ Classic_Postmortem_Ensemble_Studios_classic_RTS_Age_of_Mythology.php
32	https://www.gamasutra.com/view/feature/132696/postmortem_double_fines_brutal_.php