Kimi Sulopuisto

LEAN PRE-PRODUCTION FOR INDEPENDENT GAME DEVELOPMENT

Thesis
Kajaani University of Applied Sciences
School of Natural Sciences
Business Information Technology
1.12.2014

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

THESIS ABSTRACT

School School of Natural Sciences	Degree Programme Business Information Technology
Author(s) Kimi Sulopuisto	
Title Lean Pre-production for Independent Game Development	
Optional Professional Studies Game Production and Business	Commissioned by -
Date 1.12.2014	Total Number of Pages and Appendices 49

"Going indie" is a video game industry term for moving from whatever one is doing now to being a full-time independent game developer. With the ease of digital distribution and the availability of game development tools, practically anyone is free to start making games professionally. Developing a successful video game takes a great deal of work, however. Many prospective developers bury themselves in code, burn out, and yet fail to turn a profit.

Independent developers often don't have the resources to take on the development process by sheer force. This makes proper planning indispensable: Making a successful game is much easier if as many informed design and project management decisions as possible are made in advance. Most of the hard choices are made during pre-production, which is the first stage in the process of video game development. The goal of this thesis is to chart the necessary steps to take during the pre-production stage, and to apply them to an ongoing game project.

Lean production practices are used to design a game that is as captivating as it is cheap to manufacture, and plays to the strengths of its creator. Since independent developers rarely have the budget necessary for traditional marketing methods, they often rely on organic growth via community building and word of mouth. Lean business methods are used to incorporate members of the target audience into the development process. This allows the developer to gradually build interest in the project, as well as to regularly confirm whether or not the project is going in the right direction.

Language of Thesis	English
Keywords	Video game, game design, pre-production, indie games, lean, start-up
Deposited at	☒ Electronic library Theseus ☐ Library of Kajaani University of Applied Sciences

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ
TIIVISTELMÄ

Koulutusala Luonnontieteiden ala	Koulutusohjelma Tietojenkäsittelyn koulutusohjelma
Tekijä(t) Kimi Sulopuisto	
Työn nimi Indie-pelikehityksen esituotanto	
Vaihtoehtoiset ammattiopinnot Pelituotanto ja -liiketoiminta	Toimeksiantaja -
Aika 1.12.2014	Sivumäärä ja liitteet 49

Indie-pelikehitys on omatoimista, itsenäistä videopelikehitystä jonka tarkoituksena on tehdä tuottoa omakustanteisella peliprojektilla. Nykyaikana kynnys ryhtyä omatoimiseksi pelikehittäjäksi on erittäin pieni: sähköiset jakelukanavat ja pelkistetyt kehitystyökalut mahdollistavat sen, että kuka tahansa voi alkaa kehittää videopelejä. Menestyksekkään pelin tekeminen on kuitenkin erittäin haastavaa. Moni innokas pelikehittäjä lannistuu, sillä jo peliprojektin loppuun vieminen on äärimmäisen työlästä – puhumattakaan liikevoiton tavoittelusta.

Omatoimisilla kehittäjillä ei yleensä ole voimavaroja selviytyä tuotantovaiheesta omin voimin. Tästä syystä huolellinen valmistautuminen on äärimmäisen tärkeää. Kehitysprosessi on huomattavasti helpommin hallittavissa, mikäli pelikehittäjä tekee oikeat toimenpiteet ennen tuotantovaiheen alkua. Monta tärkeää päätöstä pystytään tehdä esituotantovaiheessa ennen kuin riviäkään koodia on kirjoitettu. Tämä opinnäyte käsittelee esituotantoa ja sen sisältämiä indie-pelikehittäjälle elintärkeitä vaiheita.

Opinnäyte esittelee lean-menetelmiä pelisuunnitteluun ja projektin hallintaan. Pelistä pyritään tekemään kevytrakenteinen ja helppo valmistaa, mutta silti mahdollisimman laadukas. Markkinointipuolella lean-ajattelun avulla tunnistetaan asiakasryhmä ja otetaan siihen tuntumaa; tarkoituksena on liittää asiakkaat osaksi pelikehitysprosessia. Näin pelikehittäjä saa jatkuvaa palautetta työstään ja vähitellen tuotua peliprojektia ihmisten tietoisuuteen.

Kieli	Englanti
Asiasanat	Videopeli, pelisuunnittelu, esituotanto, peliala, start-up, indie
Säilytyspaikka	☒ Verkkokirjasto Theseus ☐ Kajaanin ammattikorkeakoulun kirjasto

CONTENTS

1 INTRODUCTION 1

1.1 Starting point 2

1.2 Aim of the thesis 3

2 LEAN DEVELOPMENT PRACTICES 5

2.1 Grip & minimum viable product 6

2.2 Maximizing potential 7

2.3 The hook 7

2.4 Lean design 8

2.4.1 Simplicity vs. complexity 9

2.4.2 Literal vs. conceptual 9

2.4.3 Lego-block design & emergence 11

2.4.4 Discrete vs. continuous mechanics 12

2.4.5 Player choice 13

2.4.6 Focus on the core 14

3 RESEARCH & CONCEPT 16

3.1 Idea, title & hook 17

3.2 Scope 19

3.3 Goal 20

3.4 Platform 21

3.5 Genre & mode 22

3.6 Monetization model 24

3.7 Audience 25

3.8 Technology 27

4 EXECUTION 29

4.1 Game design document 30

4.2 Art & audio 31

4.3 Marketing, advertising & PR 34

4.4 Prototyping & testing 37

4.5 Funding 39

5 CONCLUSION 42

SYMBOL LIST

AAA	In the video game industry, AAA or triple-A is a term used for games and studios with the highest budgets and levels of promotion. In many ways, this is the opposite of independent development.
AI	Artificial intelligence: Used to generate intelligent behavior in non-player characters, often with the purpose of simulating opponent gameplay and strategy.
Game jam	A gathering of game developers for the purpose of planning, designing, and developing a game within a very short span of time.
Indie	Independent; in video game development, this signifies that the game is made by a small development team without the financial support of a video game publisher or other outside source. In recent years, indie game development has become a huge movement within the video game industry.
Lean	Lean manufacturing, lean production or simply "lean" is a customer-centric philosophy that considers the expenditure of resources in any aspect other than the direct creation of value for the end customer to be wasteful.
Power-up	Power-ups are in-game objects that instantly benefit the game character. This is in contrast to items, which can be used at a time chosen by the player.
Platform	Electronic systems that are used to play video games. These include personal computers, game consoles and mobile devices.
Playtesting	The process of exposing a game in development to its intended audience in order to identify potential design flaws and gather feedback.
Prototype	A prototype is an early version of a product built to test a concept or process in practice.
Startup	A business that is generally newly created and in a phase of early development.

1 INTRODUCTION

It's 2014: I'm about to finish my studies, and I'm really aching to make games. My intention is to move from a student and hobbyist to a full-time professional. But I'm not aching to make just any game – I'm aching to make my game. "Going indie" is a video game industry term for moving from whatever one is doing now to being a full-time independent game developer, and that's what I want to do.

Why? Because the job prospects for a game designer are scarce. Because I don't want to waste my time as a lowly game tester when I know I can do better. I don't want to make someone else's coffee and pray for a chance to rise up the ladder. Instead, I want to actively pursue my fortune. I find corporate life unfulfilling: I don't enjoy being told what to do, when to do it, and how to work. Finally, I think many of the big companies' games are dull. I'd rather put my time and energy into bringing my own game ideas to life.

The goal of an independent game developer is to work as lean as possible while generating experiences that people will want to pay for. At the same time, I see aspiring developers slaving over their projects yet receive nothing in return. With the advent of digital distribution and simplified tools, anyone can start making games right now. This brings a temptation to run headlong into development. But that is often a road to frustration and failure. Working this way, it's all too easy to spend every available minute on the game project, burn out, and never turn a profit.

Some look at local trailblazers, such as Supercell, and conclude that funds and resources are what separate success from failure. Others say it took Rovio over 50 games to break through – that the ordeal is impossible without decades of experience. Some still vouch that you need unreal luck to make it on your own. They consider independent development just a big gamble.

But I disagree. Success is naturally a combination of a hard work, skilled craftsmanship, and good timing. In addition, millions in the bank as well as lifelong experience do make a difference. Without these, you can't afford to push through the production by force. But you don't have to. If you make the right design, marketing and project management decisions – before typing a single line of code – I believe it's possible for even an inexperienced developer to make a successful game.

1.1 Starting point

You don't need qualifications or AAA-studio experience to be a game developer. You also don't need a college degree any more than you need to have published game titles on your résumé. Anyone can become a video game developer right now, and starting to make games on your own is in fact a very efficient way to become good at it. There are many cliché expressions for this: Practice makes perfect; the 10,000-hour rule; 95% sweat, 5% brilliance. (Schweer 2012.) But it's a simple truth: You will become a better developer the more time you spend making games. The grunt work of video game production will greatly improve your skill in programming, micro-design, game art and more. However, the process of making and releasing a video game usually features a long list of crafts, such as:

- Game design (macro-design)
- Programming
- Level, character and feature design, and writing (micro-design)
- Art and animation
- Audio and music
- Marketing and business (Taylor 2012.)

All of these are required in one form or another to create a polished video game product that people will pay to experience. Each field is also challenging to a degree where one can spend a lifetime mastering just one of them, and an independent developer will have to cover all the disciplines himself. (Taylor 2012.) With my basic skills in programming and art, I would be working till the end of time and burn myself out, let alone being able to compete in quality in the increasingly tough indie game market.

This is why an independent developer will always want to design within restraints and let limited resources, skills and availability challenge his creativity. Don't try to make the game of your dreams while desperately trying to find the right resources for it. Rather, design a game around your current skillset and circumstances. (Schweer 2011.) At the same time, don't compromise your ideals: Make something you want to make and are proud of. There are aspects I personally value in games, and these will act as a framework for my project:

- I want to make games that are smart enough to generate fresh and interesting experiences.

- I want to make games with simple rules but complex results. The permutations and outcomes should occasionally surprise even myself, the designer.

- I want to make games that let the player be genuinely creative.

- I want to make games that are fun to learn, where the game has minimal explanation and where playing the game is the most intuitive way to learn.

- I want to make games that value the player's time, never forcing the player to repeat tedious functions or perform boring busy work.

The trick is turning these statements into efficient design that creates value for myself and for my customers, and to find theoretical background for making a game that is fun to play yet cheap to produce. It's no use to design something one can't make, so the question is: Is it possible to make a game that is deep and creative, with simple rules, with very little money? In addition, what kind of a game would that be? Which methods are the most efficient?

1.2 Aim of the thesis

Independent game developers cannot use the traditional waterfall development model that big corporations use. This model is sometimes derogatively called "build it, ship it and pray", and for good reason: Building a product in strict vacuum leaves much to chance. On the other hand, budding developers can't simply figure things out as they go along. All-out freedom leads to irresolution and inefficiency. Instead, each stage of game development must be treated as a distinct project of its own, where time and effort are to be focused on a single stage at a time. Any failures are suffered as early as possible, and the eventual success of the current stage is used to carry the next. (Schweer 2011.)

The lifespan of a video game's development process is usually divided into three high-level stages:

1) Pre-production: The game idea is formulated, shaped into a documented game concept, prototyped, and possibly pitched to a publisher or investor.

2) Production: A product backlog is made according to the game design document and based on the prototype, and the majority of the code and assets are created.

3) Post-production: The game is published and pushed to the market, and supported post-release via patches, additional content and community interaction. Success is valuated with metrics, and future plans are made accordingly. (Obenchain 2013.)

Success in indie development is very unlikely if the first steps of the process are poorly executed. To emphasize: Any indie game project should be rethought if the pre-production is not at first a resounding success. This is why the aim of this thesis is to build a cohesive roadmap into laying the foundations for indie development success, as well as to follow those steps in my ongoing game project. The focus will be solely on the pre-production stage.

The goal of pre-production is to take a game idea and transform it into a full-fledged game concept that thoroughly conveys what the game is about, what it looks like, and who it's for. While game designs shift and change during the production stage, often as a result of prototyping and playtesting revealing features that must be improved upon, there are aspects of design that cannot be altered without undoing plenty of work. These critical elements must be set in stone as soon as possible. (Adams 2009, p. 44-46.) Pre-production also entails the bulk of "macro-design" – conscious, informed decisions based on theoretical argument. These things are much easier to read and write about, as opposed to micro-decisions that rely on instinct and tacit knowledge derived from long experience. (Ismail 2014.)

For the purposes of this thesis, I will divide pre-production into two distinct parts: The concept stage and the execution stage. In the concept stage of game design, you make decisions regarding your game that you live with for the life of the project. It's like pouring the foundations for a building: You can revise the decoration and interior when required, but you can't decide that you really wanted a different type of building halfway into construction. The execution stage of pre-production is where the plans from the concept stage are put into motion via art, prototyping, marketing, and more; funding is the final piece and bridges the gap between pre-production and production.

2 LEAN DEVELOPMENT PRACTICES

"Obscurity is a far greater threat than piracy." - Tim O'Reilly (Taylor 2014.)

"Lean" is the most important word borrowed from the vocabulary of startup companies. Many startups begin with an idea for a product but too little research. They might spend months, sometimes years, perfecting the product without ever showing it to the prospective customer. When the company inevitably fails to turn a profit, it is simply because they never communicated with the customer. They never determined whether or not the product was interesting, or if there was a market for the product to begin with. At the end, the customers show complete indifference for the product, and the project fails. Any money and time will have gone down the drain. (Ries 2011, p. 13-14.)

Lean development is all about being agile and customer-centric. Even a game of high technical or artistic quality will fall into obscurity if it doesn't find an audience, or if it doesn't meet the specific expectations of its target market. The goal of an indie game developer is to work as efficiently as possible while developing games that people will pay for. With limited sources of income, counting precious pennies, you want absolute confidence in the work you're doing – to know that what you're creating will be successful. (Schweer 2011.)

The most valuable aspect of the lean method is that it incorporates customers into your game development process, so you know that you're building a game that people will love way before you launch – even before you invest much time or money in the project at all. This is vital because independent developers don't have the money in the bank to gamble on a dream. (York 2012) Meanwhile, creating a minimum viable product ensures you're able to present your game to people outside the project as soon as possible. This is a great way to gather criticism, discover design flaws, and rectify mistakes early in the process, when as few assets as possible have to be thrown to the trash. (Rogers 2010.)

Lean isn't used just for manufacturing and production, however. Lean applies in every business and design decision. It is not a cost reduction program, but a way of thinking and acting. (Ries 2011, p. 13-14.) There are many game design strategies that fit in the lean development category; these are parallels to their traditional industry counterparts, cutting corners whenever available while improving the product with the customer in mind. To paraphrase, lean game

design forgoes allocating production resources in any other aspect than that which creates value for the end customer.

2.1 Grip & minimum viable product

Doing business is deceptively simple, following a sequence of:

1) Find out what they want (research & planning)

2) Go get it (production)

3) Give it to them (delivery) (Tristem 2014.)

See how these parts match up to the three stages of game development: Pre-production, production, post-production? Pre-production is all about research and planning, and many people downplay or completely miss this first step. A financially unstable indie developer must test any assumptions by trying to build grip as soon as possible, where grip can be defined as such: You have grip when people consistently part with their time, information or money in response to your offer. (Tristem 2014.)

What offer? The minimum viable product. In lean manufacturing, a minimum viable product is the bare minimum that you need to create in order to test the assumptions of your business: To prove that your game has an audience that finds your game compelling. (Oedekoven 2011.) In the case of an independent game developer, the minimum viable product could be a game prototype or the game concept, or perhaps even just the idea paired with a mock-up. It may seem ludicrous asking people to back something that's little more than an idea on a piece of paper, but it taps into a positive feedback loop. Either, people aren't interested and you "fail fast" – another start-up term – and move on to the next idea. Or the people love it and you keep going. (Tristem 2014.)

2.2 Maximizing potential

The first rule of lean indie development is to make something you want to play. That way you have at least one happy player, and there's no waiting for feedback before changes or trying new ideas. It's also easier to work on something when the main drive is that you're eager for it to be done so that you can have it. If you were just a fan of the game you're making, you'd wish there was something you could do to get it done sooner and better. If you're a fan and the developer, then you've no-one else to fault. (DeLeon 2010.)

You want to maximize your working potential. Personally, I'm a poor programmer and have almost zero experience in audio production, so I will seek to partner up with one or two people with talent in those areas; that leaves me to focus on the design and art disciplines. Multiple people also have multiple minds to apply to strategic issues, as well as likely very different personalities to evaluate the product. This helps immensely, and for this reason any lone developer should consider bringing in at least one other person eventually. Very few people can realize their full potential working on their own forever: It's simply too convenient to have that second opinion. (Taylor 2012.)

Being your own boss can be deceptively tough. An independent developer will have to make his own schedule yet keep up a hearty pace. You will have to be brutally honest with yourself, and this is made easier if you use awards, competitions and other events – even semesters – as real deadlines. Events such as Independent Games Festival have submission deadlines that force you to make fast decisions and commit to a schedule. The benefits of such competitions are multifaceted: The awards and deadlines are a great motivator, and the event is a great way to connect with like-minded people and start building grip. (Yu 2010.)

2.3 The hook

To attract interest in your game as soon as possible, the game needs a hook. A hook is an angle that makes your game interesting and sets it apart from everything else. Imagine a news story about your game – what would the headline be? Would it be intriguing? (Rose 2013.)

Since most indie developers don't have the budget necessary for traditional marketing methods such as television ads or street banners, they often need to rely on guerilla methods to

foster organic promotion. (Larochelle 2014.) One of the worst things for an indie game developer to hear is that they have a "nice little game." Though often a genuine compliment, it also means that the game isn't perceived as "hot". To properly build grip and awareness, the game concept needs to appeal to the indie demographic: These are the people that are genuinely interested in games and are compelled to search for and spread information about them. If the game doesn't pique this group's interest, the game will not attract attention on the indie-friendly channels that are vital for successful lean marketing strategies. (Uribe 2014.)

A perfect example of a hot indie concept is a game aptly named Superhot (fig 1). Superhot is the result of a game jam where a student team built a browser-based prototype featuring a simple, unique mechanic: It is a first-person shooter where time moves only when you do. Think about it: A shooter where time moves only when you do. That is a fascinating concept no matter how you say it. (Sarkar 2014.)

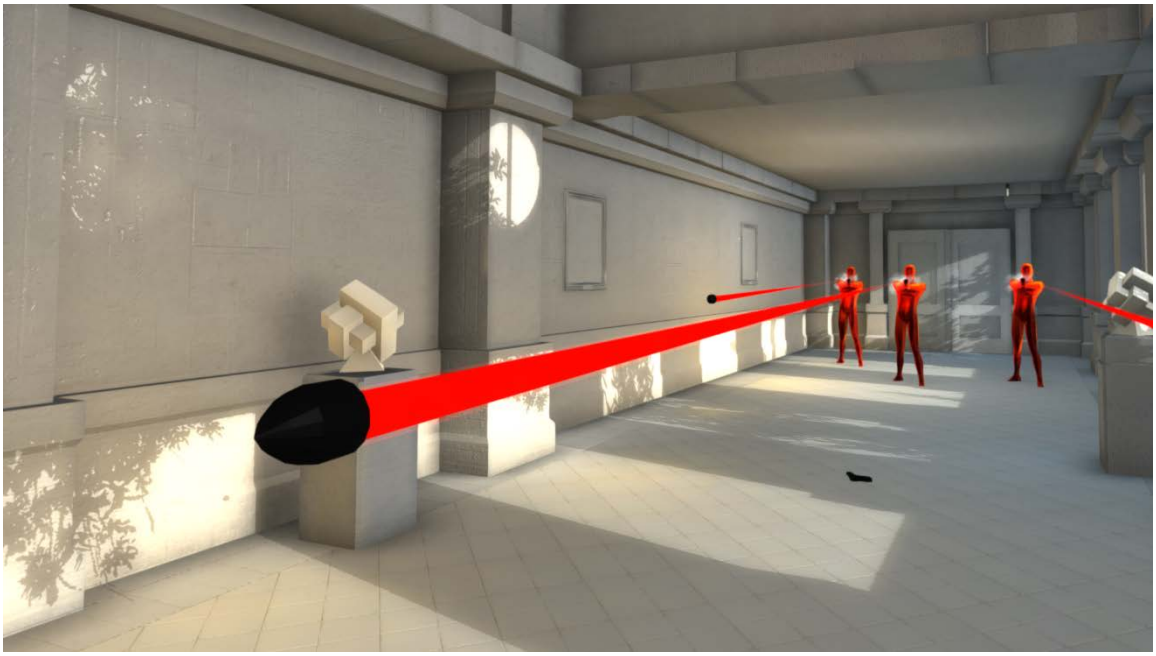

Figure 1. In Superhot, the player can literally dodge bullets. (Superhotgame 2014.)

2.4 Lean design

"I'd say the best design is where the game becomes better from the developers being lazy." - Olle Håkansson (Håkansson 2014.)

There is constant tension between simplicity versus complexity, and shallowness versus depth. There are countless of simple games that are shallow, and complex games that are deep. But making a game that is both simple and deep is a rare achievement. Good design is all about how integrated the mechanics are: Tightly-woven games will play differently every time, and make new situations emerge even with the smallest changes in their configurations. The ultimate lean design challenge is to design the deepest game possible with the simplest possible system. (Venturelli 2014.)

2.4.1 Simplicity vs. complexity

The 80's and 90's were a time when computer game design was dominated by a kind of bottom-up thinking: Levels were made out of grid tile arrangements, the player character was often non-humanoid (e.g. a space ship), and enemy AI was nothing but simple patterns on repeat. This was a result of technology's computational limits. Modern computers are now powerful enough to render complex 3D models and calculate adaptive AI. However, making these things takes plenty of time and requires very specific expertise, and usually leads to what is only one particular type out of many possible aesthetics. (DeLeon 2010.)

Making a single room for a complex AAA-game may take more hours than designing an entire level in a simple indie game. If you're not a talented 3D-modeler as well as an expert in lighting and audio, your room in the AAA-game would look rough no matter how much time and effort you put into it. By comparison, even a fledgling developer could build a decent level if the game is simple enough. All that, and many complex games are worse than many simple games. There are simply far more ways to fumble designing or creating content for a complex game as opposed to a simple game. Even worse, the hard work gets burnt on trying to force what's ultimately as artificial as a cartoon to look as photorealistic as live film. Would photorealism improve Disney products? Probably not. (DeLeon 2010.)

2.4.2 Literal vs. conceptual

Simple is conceptual, complex is literal (fig 2). This means that simple things often afford a vastly larger range of perspectives and ideas to explore when compared to the literal, complex representation of how things look or function in real life. (DeLeon 2010.)

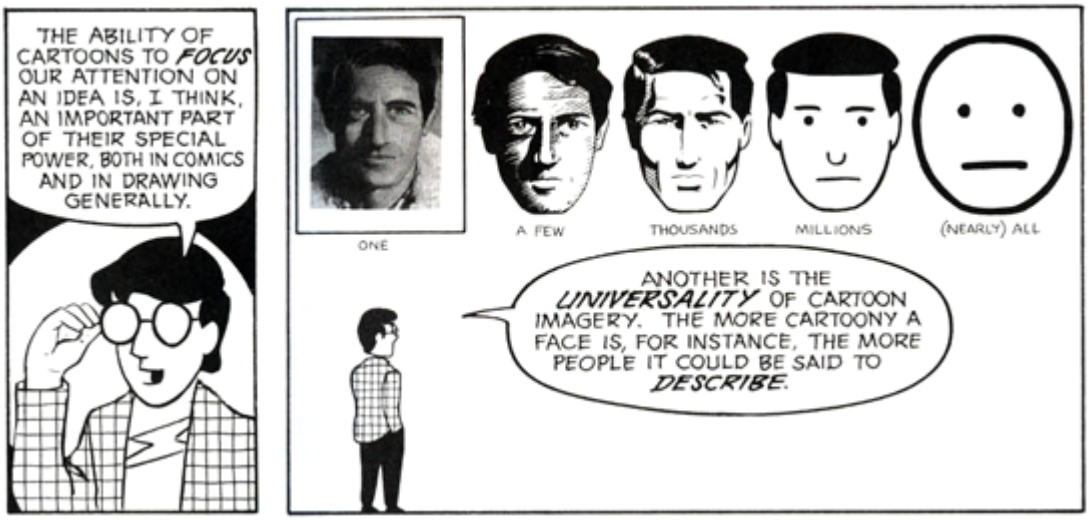

Figure 2. (DeLeon 2010.)

The game called Minecraft (fig 3) is a prime example: The game features plenty of resource gathering, an instance of which involves the player punching a tree to extract wood. The visuals are detailed enough that the player can understand this activity as woodcutting, and use some real-world knowledge to apply in the game world. But proper abstraction ensures the player doesn't stop to question how absurd the activity really is. In this vague presentation, punching trees for wood makes sense. The level of visual abstraction in Minecraft matches the abstraction of the game mechanics; increasing the level of graphical fidelity would destroy this balance and instantly break immersion. (Gile 2014.)

Figure 3. Woodcutting in Minecraft. (Gile 2014.)

It wasn't laziness or the lack of ability or resources to strive for realism. Instead, the developer of Minecraft used lean design to simplify the game and accelerate production. It cannot be emphasized enough: This kind of lean design was decades ago required for video games to function. But even today, those methods can save a tremendous amount of time and energy while improving the end product. (DeLeon 2010.)

2.4.3 Lego-block design & emergence

It is vital to learn to recognize patterns in your ideas for your game and to convert them into generalized system rules rather than trying to implement dozens of individual cases. This is similar to building with Lego-blocks, where the blocks are simple but the resulting contraption (fig 4) can be very complex in comparison. By designing general patterns rather than individual cases, you can more easily understand how your game will function, and you will also make it easier to program. (Adams 2009, p. 311-312.)

Figure 4. (A Lego a Day 2013.)

There is a saying, "The whole is greater than the sum of its parts." This is especially true for games. Chess is famous for generating enormous depth of play with relatively simple elements and rules. Something similar can be said of computer games such as Tetris. Both of these games consist of simple parts, yet the number of strategies and approaches that they allow is

enormous; no two sessions will feel the same. This is an attribute called emergence. (Adams 2012, p. 25-26.)

The emergent quality of the gameplay comes not from the complexity of individual parts but from the complexity that is the result of the many interactions among the parts. The advantage of this approach is that these kind games are efficient to build but hard to design. (Adams 2012, p. 26.) This works greatly in favor of lean design practices, because design is the primary arbiter of the scope of the game. It is in an independent developer's best interest to create design-heavy, production-light games. (Powers 2014) "Rocket jumping" is a famous example of unintended gameplay: It is the act of assisting a character's jump with the blast wave of a nearby explosion, and it is as unrealistic as it is enjoyable. In early first person shooters, the technique was emergently possible by explosion force being a generalized rule. This also illustrates that in games it is more important to be consistent than realistic. (Adams 2012, p. 44.)

With that said, you should still create special cases for variety or when circumstances require it. For example, "boss" enemies often behave distinctly rather than as a recombination of shared components. This gives them unique character, and makes the gameplay experience feel less formulaic. Every video game player, whether experienced or not, has many preconceived notions. Good design often involves being aware of those notions, and only violating them if there is a compelling reason to do so. (DeLeon 2010.)

2.4.4 Discrete vs. continuous mechanics

Many modern games simulate game physics with extreme precision: A game object might be positioned half a pixel more to the left or right, and this can have a huge effect on, for example, the result of the player character jumping. This creates a smooth, continuous flow of play, which is the main characteristic of continuous game mechanics. In contrast, the rules of an internal economy, such as health or damage, tend to be represented with whole-number values. In a game economy, elements and actions often belong to a finite set that does not allow any gradual transitions. For instance, you cannot pick up half a power-up. These are discrete game mechanics. (Adams 2012, p. 9.)

This difference between game physics and game economies affects the nature of player interaction as well as the designer's opportunities for innovation. It's important for a designer to know which type of mechanic his game will use; some games use both. Innovation is one of

the most desired perks of independent games, and discrete mechanics often leave more room for creativity. While most of the time there is little point in completely changing the physics of continuous games, many such games do include a unique system of power-ups or skills, or an economy of items to collect and consume, to make their gameplay stand out. Physics are comparatively easy to design because of the clarity of Newton's laws and the increasing computing power available to simulate them. The laws of economics are more difficult to design, but complex design is also less expensive than heavy-handed code or graphics. (Adams 2012, p. 11-12.) Discrete mechanics therefore tend to be the leaner choice in game mechanics.

2.4.5 Player choice

"I've always felt the best choice is one that leaves you wishing you could choose all the options." - Maria Jayne (Nutt 2014.)

There's two overarching design principles almost every game should fulfill: To get the player making interesting, meaningful decisions as quickly as possible, and to minimize the amount of time the player has to do boring things. Player engagement is all about giving the player interesting choices. (Taylor 2012) But if you swarm a player with too many choices, they will either pick randomly, or stick with what they have tried before. Neither case is considered interesting play, resulting in "analysis paralysis" and player boredom. (Venturelli 2014.)

XCOM: Enemy Unknown is a game that bears an excellent instance of player choice. Although the gameplay consists of purely discrete mechanics, soft limits are used to gracefully constrain player choice. The player commands soldiers, where each is allowed to move anywhere in the game field the player wants. But that's not how player perceives the situation, because each location has one of three hard attributes: No Cover, Half Cover or Full Cover (fig 5). Leaving the soldiers out in the open usually results in their demise, so players will almost always opt to move their units towards Full Cover or Half Cover locations. This creates an elegant constraining of the player's space of possibility when positioning the soldiers: You always have just a handful of real choices to make, even though you are free to stray off the safe path if the situation calls for it. While there are many other considerations to make when moving, such as flanking, line of sight and range, ultimately the player is first filtering these options through the lens of available cover. (Venturelli 2014.)

Figure 5. An XCOM soldier in Half Cover behind a waist-high crate, designated by a half-full shield icon. (PSNation 2012.)

If XCOM: Enemy Unknown did not have these cover rules, or if they were not as important to keep your units alive as they are in the current design, players would have to consider more options. Suddenly, looking two or three turns ahead would be an insurmountable task; the game wouldn't feel as fluid, with turns lasting significantly longer. On the other hand, what if you could only move your units to cover spots? In this case, the game would lose much of its emergent potential. There are occasions where a soldier can be made to sprint out in the open and save the day, and these situations would not be possible with this constraint. (Venturelli 2014.)

2.4.6 Focus on the core

"The action that your player performs most frequently should feel like fun all by itself." – Anand Ramachandran (Ramachandran 2014.)

To elaborate the quote, the core gameplay of most games comprises the player performing one or two actions very often. Although these actions are often used to carry out complex maneuvers, the core actions should be at least decently entertaining on their own. They should

feel satisfying enough to perform hundreds of times even without any additional context. (Ramachandran 2014.)

Nintendo's Super Mario -games are famous for the solid physics governing the controls of the game's titular character. Even if one were to remove every obstacle and enemy Mario faces, simply making the character jump or run would be entertaining for a while. The same can be said for Blizzard's hack and slash game, Diablo. The core chunk of Diablo involves only clicking on monsters to attack them, but there's a very visceral feeling to this: The crushing blows are accompanied by grisly sounds and brutal death animations. This means that the player could click on the monsters all day, making it engaging even without the complementary gameplay elements. (Ramachandran 2014.)

When it comes to building a strong core for the game, there's something called the 80/20-rule. To illustrate, 80% of the user's enjoyment comes from 20% of the work that you produce; how responsive or visceral the game feels is often within that 20%. If the core is not satisfying, nothing else about the game is going to make up for it. You should strive to get that fraction the best it can be. Whatever comprises the other 80% should only be good enough to not distract the player from the main 20%. Otherwise too much time is being spent on things other than the core. (DeLeon 2010.)

3 RESEARCH & CONCEPT

Video game design is the process of imagining a game, defining the way it works and describing its elements. Designing a game begins with an idea, but ideas alone are worth very little because everyone has them. (Adams 2009, p. 29-30.) That said, not all ideas are created equal. This is why it's vital for any game designer to have a backlog of ideas to draw upon. When starting a project, always choose an idea you have already incubated for a time, and one you believe you have the resources to create. (Yotes 2013.)

It's easy to think that creativity is a passive force. But the best ideas never appear like magic. In truth, game ideas can be found almost anywhere but only if you're looking for them. If you're actively working towards getting an idea, it will eventually take form. (Adams 2009, p. 64.) After choosing the idea to work on, the first course of action is to translate the raw idea into a concept document. The concept is meant to describe the general specifications of the game in sufficient detail for future design work. It's the first treatment of the game, and the very first step on a long journey. (Rossi 2013.)

When it comes to project management, no change goes without rippling effects to the rest of the design because each decision affects the project as a whole. The game concept lets you look at the big picture and contrast each decision in relation to every other choice you've made, while upholding a vision of what the game will eventually become. (DeLeon 2009.) Documentation also eliminates hype by forcing you to define the substantial elements of the game and scale back any far-reaching pipe dreams to something feasible. (Ryan 1999.)

In this first part of pre-production, you make decisions that you live for the entirety of the project's duration. Such fundamental elements of the game are established that changing them later would require throwing away a great deal of work. For example, it's not acceptable to change the game genre or target audience halfway through production. (Adams 2009, p. 45-46.) While game designs live as the development progresses, redoing plenty of production work is far too expensive for an independent developer. To ensure this doesn't happen, many of the most important decisions must be made during the concept stage. (Rogers 2010, p. 80.)

3.1 Idea, title & hook

Many people who spend plenty of time playing video games want to design them as well. This is natural: When you play games, you develop a sense of their inner workings as well as the strengths and weaknesses of various gameplay mechanics. Playing games is an invaluable experience for any game designer; it allows you to compare different features of different games and consider why one works while the other doesn't. Moreover, many new game ideas are born from the desire to improve an existing game. It's likely you've looked at a game and thought, "If I had made this game, I would have…" (Adams 2009, p. 66-67.)

But it's not enough to simply play games as a pastime. To learn from other games, you have to pay attention as you play. Look at each game with a critical eye, and deliberate how and why the mechanics work. Take note of anything you like or don't like, and of features that contribute to those feelings. Also think of how others might feel about the game and its mechanics: Are some aspects of the game universally lauded? If so, why? Moreover, playing games is a great way to get a broad sense of the market, best practices, and potential competitors. (Adams 2009, p. 66-67.)

There are three rough categories or types of games that should pique the interest of a game designer: Games you want to make, games you want to have made, and games you're good at making (fig 6). When coming up with a game idea, the one with the most potential is likely to fall into all three of these categories. This way you have an idea for a game that you're excited to bring to life, is similar to games you're probably used to playing, and is of a genre you understand the best. (Yu 2010.)

Figure 6. (Yu 2010.)

There's really no right or wrong way to document game ideas. I always write new ideas on paper first. The reason for this is twofold: Paper is the fastest medium for scribbling and sketching, and I get to double-check the idea when I eventually type it down on the computer. This gives me a chance to proofread what I've written down. Many ideas that sound good in the heat of the moment are in fact terrible.

The next step is to let the idea sit. I liken it to cooking over a low flame for a long period of time. I let the idea mature in my head while making small revisions over time, and I often keep game ideas in this incubation stage for very long periods of time. Many of the ideas in my head are ones I don't plan to use for years; some I discard, others I merge into one. At the start of the project, I always pick one of these well-cooked ideas to turn into a game concept. The choice is based on my current skills and circumstances. If I don't completely believe I could make the game I imagined in the notes, I'll let it sit longer and choose something more feasible. After all, there's no sense designing something I can't build. Ideas alone are worthless.

My project begins with the idea for a game called Starkillers. It's a sci-fi strategy game similar to FTL: Faster than light (fig 7), only with one-on-one turn-based multiplayer and progression à la Hearthstone. "FTL meets Hearthstone." I introduce the idea to some of my friends and peers in this early stage to test for grip. The idea is well-received. This means very little yet surprisingly much; after all, it is the first green light in the development process.

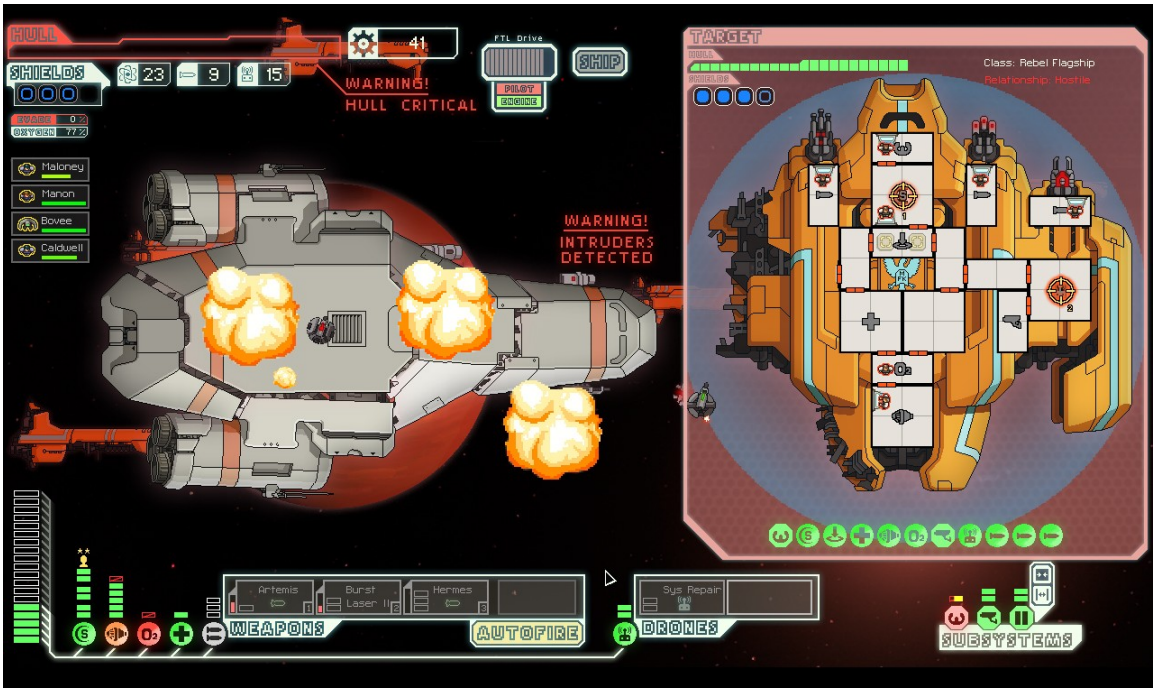

Figure 7. FTL: Faster Than Light, a simple-but-deep strategy game of high critical acclaim. (PC Gamer 2012.)

3.2 Scope

Game ideas are like blocks of stone: They need to be carefully chiseled into shape. Once a game idea is established, the first thing to do is start trimming it down. Turning an idea into a game concept is almost always a subtractive process. Where ideas are fleeting dreams and imagination, a game concept needs to be unambiguous enough to seriously discuss the game as a product an audience would like to buy. The next step is to set boundaries that help shape and funnel the design. (Schell 2008, p. 75-77.)

The first thing to consider is the scope of the project: how big of a task are you undertaking when you set out to make your game? It's useless to spend time on something you cannot realistically hope to achieve. What about resources: How big is your team? What are their talents? The baseline is that you should always design within restraints and let the limits in resources and schedule drive your creativity, not the other way around. (Schweer 2011.) To paraphrase: Create as fun a game as you can with the resources you have. It is infinitely better to create a weaker product and finish it than it is to cancel the project because you designed bigger than you could create. Programming and art are the heavy lifting of production work but design is the arbiter of scope: It defines how much grunt work is required to cross the

finishing line. If you design too big, you run out of time or money; if you design too small, your game is not exciting enough to turn a profit. (Powers 2014.)

But how much does game development actually cost? No project is literally zero-budget. Even if you're working on your own, you need food, electricity, and a place to stay. Each new team member will require additional overhead. A standard metric is to multiply the number of developers by the amount of development time by a fixed salary. Because it's almost impossible to avoid underestimating the time and effort it goes into finishing a video game project, it's typical to compensate for this by overestimating the expenditure. (D'Angelo 2014.)

As a prospective indie developer, I have to consider the business end of the project. I expect to make my game in the span of two years, with two other people besides myself. The formula therefore looks like this:

- Number of developers: 3
- Number of months spent on development: 24
- Expenditure per month: 5,000 €
- Budget = 360,000 €

That's more than quarter a million! Fortunately, a big portion of that can be mitigated by clever bootstrapping methods. For example, I don't need to bring other people on board during pre-production because my development skills are enough to cover the early stages. But the truth is: To fully realize my game project, I will have to seek external funding at one point or another. In order to do that, it's vital to know the ballpark figure and plan accordingly.

3.3 Goal

There are commercially successful indie games about a plethora of subjects: Acrobatic miners in search of an ancient treasure; the commander of a virtual SWAT team; and an explorer stuck in a non-Euclidean labyrinth. These are innovating concepts behind several commercially successful indie games. These games are never going to be as big as Call of Duty or the Sims, but the strength of indie development lies in the fact that they don't need to be. AAA franchises must rake in revenue by the millions because they cost as much to make. For a

small-time game studio, however, commercially successful means only making enough money to continue making games. (Taylor 2012.)

To make a career out of independent game development, you will have to find a niche and stay ahead of what others are doing. Define your goals at the start of the project, rank them in order, and use them as a compass to direct the design and development. The question is: What do you most want to achieve with your game project? Is your goal:

- To get good review scores?
- To make a certain amount of money?
- To get plenty of players and popularity? (Rose & Short 2014.)

For my project, these are in the order of importance: Critical acclaim above all else. I want to make a game that is taken seriously and not looked down upon. I want to build a small but loyal community, to find a niche where I have true competitive advantage, even at the cost of excluding many potential players. I want to build a brand and keep on making more of the same: Hardcore, intelligent games for an audience that dedicates a large amount of time to video games. Some revenue is certainly necessary to keep on going, but building a following is far more important to me at this early point in my career. After all, obscurity is a greater threat than piracy; getting my name out there will make future ventures much easier.

These priorities greatly affect my choice of platform, target audience, and business plan. I also have to keep the goals in mind when I bring aboard any other developers, because their plans have to match mine – at least on a high level.

3.4 Platform

The first technical standpoint to consider in a game concept is the platform: That is, which machines the game will run on. For example, some genres of games are better suited to one kind of platform than another. Each device has multiple features and characteristics to consider, such as screen space, performance, storage space, and peripherals. Like any of the hard design decisions to follow, these aspects have implications on other, future discretions. The user base also wildly differs from platform to platform. It is vital to know the strengths and

weaknesses of the different types of platforms as well as the users' habits and behavior. (Adams 2009, p. 77-78.) From a high-level game design perspective, the most obvious branch is between desktop and mobile: The contrast in specifications is the most pronounced between these two archetypes. (Henshell 2014.)

There's a saying, "Write what you love." That is the single best tip for choosing the platform you will develop for. Too many developers chase trends or spend time pondering what someone else would like them to do. Sometimes that's the right train of thought. After all, you want to make games you can sell; empathy is required on occasions. But in this case, maximizing working potential trumps any other considerations. (Henshell 2014.)

I'm an avid fan of home computers as a gaming platform, and I've been using Windows operating systems since I was a child. A great deal of lean strategic advantage comes from battling on home turf. Working on my own, it would be insane for me to develop for anything else. Moreover, I don't like casual games that the mobile platforms are known for. I don't play them, and they bore me. Write what you love: I greatly enjoy deep games of strategy found almost exclusively on PC. In addition, PC has remained a solid platform for video games for more than 20 years. Chasing trends is best left for the big companies.

Finally, many mobile games have had to endure brutal reviews that clearly have a grudge against mobile gaming, as well as critics dismissing the game out of hand. Even though mobile games comprise a massive market, many game critics or review websites choose not to cover mobile games at all. (Rose & Short 2014.) I have to line up the choice of platform with my goals and expectations for the game: There is some serious elitism among console and PC gamers and journalists, and I plan to make use of that quality instead of being the victim of it.

3.5 Genre & mode

Genre is the term for the category which the work belongs to. When describing movies or books, this categorization is based on the content, such as comedy or drama. In video games, however, genre refers to the types of challenges that the game offers. For example, shooter games are one genre whether they're set in the Wild West or in outer space. If the player's main activity is to fight the enemy by shooting, the game is a shooter game no matter the setting or story. (Adams 2009, p. 70.)

The purpose of any design documentation is to understand your own game and to explain it to others; defining the genre is the fastest way to focus this explanation. It's tempting to start thinking about your game in terms of setting or story as in traditional media. For example: "Wouldn't it be fun to play a game where you're the captain of space ship?" However, this does little to describe the gameplay. It doesn't answer the single most important question you can ask yourself at this stage, "What is the player going to do?" This doesn't mean assigning activities to input keys or deciding which button performs a specific action, but you do have to know the activities and challenges that the game offers. (Adams 2009, p. 68-69.)

For example, my game is a sci-fi game. That tells very little: Popular media the likes of Star Wars has set certain preconceptions when it comes to space and starships. Instead, I know my game belongs in the strategy game genre, because it involves tactical, decision-based combat not unlike submarine warfare. This choice is the result of several considerations:

- The strategy genre is a great fit for design-centric, discrete mechanics as discussed in chapter 2.
- The sci-fi setting allows for stylish, minimal art. I play to my strengths.
- Write what you love. I love intelligent games that require thought.

Another important distinction is the primary gameplay mode of the game. This is the type of activity in which the player spends the majority of his time. Most games features one primary mode: To illustrate, in many strategy games this means commanding units in the battlefield. Secondary modes might include research trees or diplomacy menus, but the player generally spends much less time in these than performing the primary activity. (Adams 2009, p. 39-40.) The mode is a key piece of information: Starkillers features ship-to-ship combat, but the player never even moves the ship! People looking for a new Wing Commander probably wouldn't want to play what amounts to a sophisticated game of Battleship. The correct audience will want to play a game such as this, however, and it's important to communicate these features to them.

3.6 Monetization model

By definition, every professional game developer is trying to make a profitable game. Unfortunately, many developers wait far too long to start thinking of monetization. Many leave it till the final stretches of the production stage. This is a backwards method, because the business model deals with the paying customer and therefore heavily affects many other aspects of the game. As per chapter 2's lean development practices, the correct time to think about monetization is at the very beginning, during pre-production, before a single line of code is written. (Natanson 2013.) Although hybrid models exist, business plans for indie games branch into two rough categories:

1) Premium, where the player will have to buy the game in full before he can play it. Some games offer a free demo version for the players to try the game before buying.

2) Free-to-play, where the game allows the player to download and play without any form of payment required at any point. If the player wants to proceed faster or gain extra customization options, he can do so via in-app purchases. Experience potions, healing salves or visual modifications paid with real money are the staples of the free-to-play monetization. (Hindman 2011.)

The free-to-play business model is very much in vogue at the moment, and this might persuade some to overlook its disadvantages. However, many who do so have ended up wishing they had gone with the more traditional model. Building a successful free-to-play game involves certain inherent challenges that aren't in favor of the shoestring indie. Free-to-play is more creatively restrictive because the monetization model only works in tandem with very specific types of game design, where the monetization has to be woven into the core gameplay mechanics. There are also other considerations, such as server upkeep, as well as the fact that free-to-play has a bad reputation in certain circles for abusive monetization practices. The latter is very important if one plans to target the indie demographic, and it is generally recommended that budding developers start with the premium model. (Taylor 2012.)

I prefer to set a fair price for my game that I feel is appropriate based on the amount of content and depth it has in it. I also make sure not to undervalue what I'm doing: Going indie and targeting a niche audience, I can compete in areas other than price. Once again, I want to align my decision to use premium monetization with my other plans. One of the key goals I

set for my game is to be taken seriously, and I think that's very difficult to achieve if the customer isn't paying for the game.

To reiterate, there's a huge psychological difference between free and not free. Money turns the transaction into a true business relationship. (Lahti 2014) At the same time, I will likely sell my game cheaper than my competitors, because I'm a budding developer and my primary goal is to build a reputation. In this case, over-delivering is a viable strategy; you can never add too much value for your customers. (Lahti 2014)

3.7 Audience

Chapter 2.2 advises to only ever make games you would also like to play. This is to simplify the development process and motivate the developer; that is, to gain strategic advantage over bigger companies where people often have to work on games they might not be so passionate about. Being the first person to like your own game is important, but it's just as important to realize that not everybody enjoys the things you enjoy. Professional game developers make games to entertain an audience. In order to turn a profit, you must think about who your audience is and what they like. (Adams 2009, p. 72.)

In regards to the commercial side of development, some of the first questions are who the ideal customer is, and which problems they need solutions for. A common error for inventors is to come up with a problem and then devise a solution. But there's no point making something that nobody needs. Find and scratch an itch – even if it's just "Many people seem to wish there were more games of X variety." Finally, think how you solve these problems the best – preferably better than anyone else. (Schweer 2012.) Finding and serving a target market is all about trade-offs – to choose what to do and what not to do. There are three rough strategies to follow:

- Serve the few needs of many customers
- Serve the broad needs of few customers
- Serve the broad needs of many customers in a narrow market (Fischer 2014.)

It has to do with the ways your activities interact and reinforce one another. The game Dark Souls is a fine instance: It is the quintessential game that doesn't try to be all things to all

people. Instead, Dark Souls has a very specific audience: Hardcore gamers who want a hardcore experience. To illustrate, Dark Souls has only a single difficulty level. The developer only needs to test and balance for one type of experience, and the customers revel in this no-frills challenge. There's barely any tutorial because the audience doesn't need or want one, and the narrative and music is sparse but of high quality. This saves up on the developer's resources while improving the customer's experience. (Fischer 2014.)

To rephrase, these trade-offs serve the audience better, and are complementary and reinforcing, and make it harder for competitors to go head to head with the developers. It's not always about using fewer resources, but using resources efficiently and to better serve the customer. (Fischer 2014.) Not unlike the developers of Dark Souls, I am making a hardcore game for desktop systems. I want serious, critical acclaim, and I have certain values that I'd rather not compromise. I'm also on a tight budget. These things considered, I am fortunate that there's a place where:

- The cost of making games has dropped significantly.
- The average game sells for 15€ and people are content to pay that much.
- The developer can get in front of his target audience for free through popular review sites.
- The customers regularly search out new games online.
- The customers have fierce loyalty and follow the developer with genuine interest. (Henshell 2014.)

That place is called the indie scene for desktop (PC, Mac and Linux) game development, and members of that audience are often called gamers. They wear game logos on their apparel, brandish concept art on their computer desktop and Facebook profile, and share development news throughout their social network. Games are part of their lifestyle, and they are passionate about them. (Henshell 2014.)

In comparison, the media expenditure in the mobile scene keeps on climbing because the trend is for the cost of customer acquisition to grow ever higher. Casual players don't read review sites or developers blogs, and have almost zero brand loyalty. They might enjoy a game but care very little for the developer. Most of all, casual players don't love games; they're

looking for distraction. (Henshell 2014.) I don't want to put my heart and soul into making something that could be replaced by reading celebrity gossip.

3.8 Technology

The last but not least of the considerations in the game concept is two-fold: Will the game use 2D or 3D technology, and which game engine or tool is used to build the game?

Many independent developers shy away from 3D-technology – and for good reason. 3D games are often more complex than 2D games, and chapter 2.4 explains in detail why simplicity is one of the core tenets of efficient lean development. That doesn't mean making 3D games as an independent developer is always the wrong choice. Instead, 2D acts as the standard which should be deviated from only when there's a strong and compelling reason to do so. (Barnson 2013.)

To reiterate, only make a 3D game if the gameplay mechanics call for it. Further considerations include:

- 3D technology requires a greater variety of specialized skillsets.
- 2D games are generally faster to develop and deploy.
- Write what you love: Many independent developers grew up with the 2D games of 90's. As mentioned in chapter 3.4, always consider home turf advantage.
- Big studios almost never make 2D games, which leaves you to compete with other independent developers.
- Most game designs work perfectly fine in 2D. The extra graphical dimension might look great, but going indie is all about working lean. (Barnson 2013.)

The chosen rendering technology naturally affects the choice in development tools. An independent studio will almost never want to build their own game engine. (Yu 2010.) Instead, think of the cheapest and cleanest way to start producing your minimum viable product. Your users will help you decide how to extend the product – right now you need to get it in front of them as soon as possible. (Schweer 2012.) To speed this process, it is wise to use as simple

tools as possible, even if you don't intend to use them during the production stage. (Adams 2012, p. 16-17.)

Use a familiar tool and write what you love; the bottom line is to be efficient. Some developers seem to think that simplified tools such as GameMaker or Construct are somehow illegitimate. But the customer will care little for which program you use. Even if they care enough to ask, they won't care enough for it to steer their purchasing decision. Use whatever means necessary. (Yu 2010.)

In my case, I have the easy choice of using the familiar Construct 2 game engine. It is an HTML5-based multi-platform editor that speeds up development via an intuitive drag-and-drop interface. Construct 2 offers the perfect 2D environment to build Starkillers: The desired features include deployment for Windows, Mac and Linux, as well as rapid prototyping via web browser. This means I must forgo 3D technology, but that is a trade-off I am more than willing to make. Best of all, Construct 2 has a very streamlined coding environment, which makes up for my lack of programming experience.

4 EXECUTION

Chapter 3 described a game concept as a general idea of how one intends to entertain a specific audience through gameplay. That description was accurate enough for an overview, but a more thorough explanation is necessary for future production. Once you have made the fundamental decisions about your game in the concept stage, it's time to move into the execution stage. In this latter half of pre-production, any design work will begin to move from the general to the specific and from theory to practice. (Adams 2009, p. 67.)

Rough gameplay statements, such as "Ammunition can cause critical hits," are detailed enough for the game concept. But moving towards production, the game design document would likely read along the lines of this: "Critical hits: The first hit can never be a critical hit (0% chance). Each subsequent hit to the same system has an increased 20% chance to inflict a critical hit. The effect of a critical hit varies between fire (laser), breach (explosive) and lockdown (ion) according to the respective ammo type." See the difference? This is what design is all about – digging into the details till there is little room for misconception. (Adams 2009, p. 54.)

When transitioning from concept to execution, the plans laid out in the game concept are expanded upon but the overarching decisions do not change. This is important: Some designers might be reluctant to make such steadfast decisions and abide by them, too. They might argue that they're keeping their options open. But this is a sign of a lack of vision and confidence, and the consequences are usually disastrous. The execution stage must have a solid foundation to build upon. In this stage, the developer writes the game design document, creates concept art, prototypes core gameplay features, launches the marketing efforts, and begins to secure funding for production. These tasks are iterative and often require multiple attempts to get right, and hence contrast with the rigidity of the concept stage. If something doesn't seem to work, don't be afraid to fall back; try a different approach as long as it's within the boundaries established in the game concept. (Adams 2009, p. 46-48.)

4.1 Game design document

In the broadest sense, the purpose of design documentation is to communicate the designer's vision in sufficient detail to implement during production. This way any programmers or artists don't have to constantly ask what they should be doing, or work in a box with no knowledge how their efforts integrate with the big picture. Documentation doesn't replace the need for discussion, because gathering into a room and sharing opinions and ideas is a much faster way to reach consensus. Rather, documentation is a viable second step: It is used to flesh out the minute details of a meeting and translate them into a plan. This is a great way to build consensus and eliminate the vagueness common to verbal ideas. Documents can also be easily revised and iterated upon via further discussion. (Ryan 1999.)

The purpose of a game design document in specific is to present the game and its mechanics, as well as a plan for implementation. But design documentation means different things to different professions: To a designer it is a way to commit to ideas and uphold the vision, while to a programmer or artist a game design document is a point of reference. (Ryan 1999.) In fact, the game design document should never be likened to a novel that you're supposed to read from cover to cover. Instead, a proper design document explains each step of the development process akin to a construction manual. (Patel 2014.)

Early in the project, the game design document is often born from the concept document and elaborated from there. In fact, design documents are often called living documents because they grow as the game grows. (Adams 2012, p. 14.) As development progresses, more and more details are added to the document. The items, objects and characters introduced in the game might be listed first. Additional details are added later: What the components do, what they affect, and how they interact with each other. All relevant roles and behaviors should be documented in an orderly fashion. (Obenchain 2013.) The game design document will also expand with any concept art as well as empirical knowledge gained from the prototype.

4.2 Art & audio

No-one wants to read your design. This is not a pleasant thing for a designer to discover, but it is something very important to be aware of. Pictures simply convey ideas much faster than plain text, and even rudimentary sketches make the design much easier to mentally conceive. This applies as much to yourself, your team as well as any people outside the project. (Rogers 2010, p. 18.) Video games are an art form, so aesthetic considerations are also part of game design itself. This doesn't mean a game has to feature beautiful art. Rather, the game must be designed with a sense of style. (Adams 2009, p. 21.) It is the game designer's responsibility to express the vision of the game, .and aesthetics are an integral part of that vision. This element isn't detached from the game mechanics, but is instead used in conjunction with in-game functionality to emphasize various aspects and to generate emotions. (Rossi 2013.)

Needless to say, a game with poor artistic qualities – sloppy artwork, clumsy animation or even a muddy soundtrack – will disappoint players even if the gameplay is superb. There's more to aesthetic considerations than beauty, however. The interface graphics, such as buttons and fonts, must complement the game's setting and theme to create a consistent experience. (Adams 2009, p. 21.) This is why the game concept must be tied to a coherent art style. For example, Frozen Synapse (fig 8) isn't just any strategy game: It presents everything about its future SWAT setting in a clean and distinct manner. (Taylor 2009.)

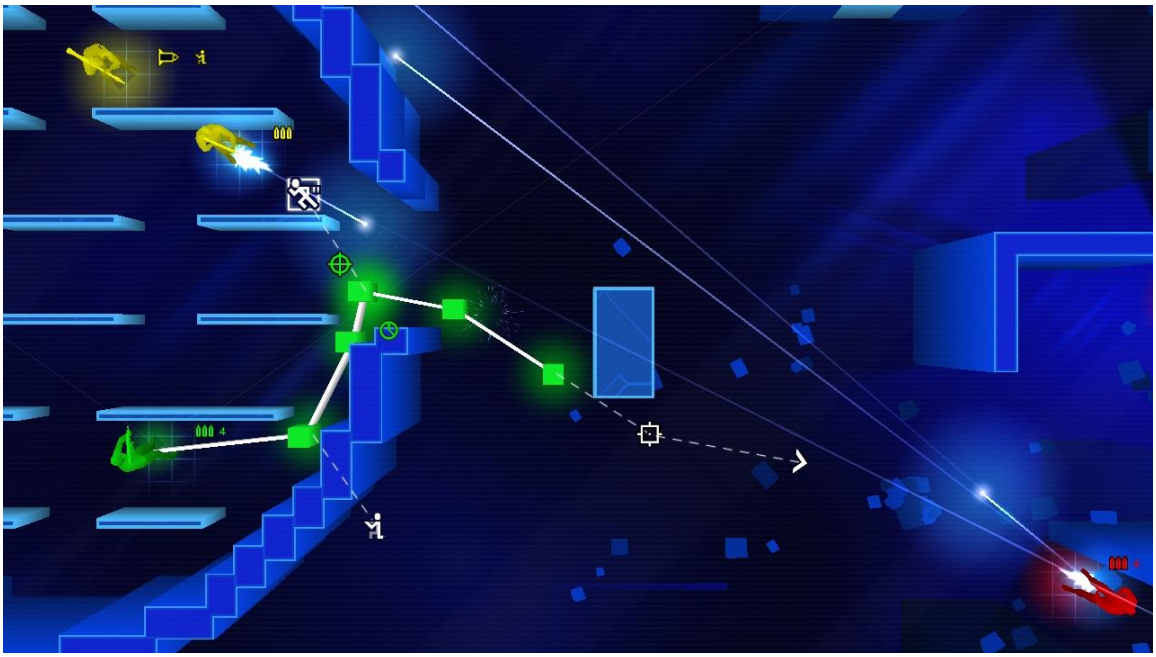

Figure 8. Futuristic squad tactics in Frozen Synapse. (Destructoid 2011.)

Consider the following aspects:

- Keep it simple: Simple art often works really well for strategic PC games – look at FTL: Faster Than Light (fig 7) or Frozen Synapse (fig 8).

- Focus on the core: When a player scores a kill in Frozen Synapse, they're rewarded with a striking death animation. This is integral to the core 20% of the game as explained in section 2.4.6.

- First impressions matter: There are many games with poor splash images and menu screens, but those are the first things that the player sees and some of the easiest parts to get right; they're often just static images. (Taylor 2012.)

Art is one of the first things I do after writing up the game concept. It helps infinitely to have even just a doodle of what everything looks like ahead of time; having sketches (fig 9) of game screens also helps with object placement when it's time to make a prototype.

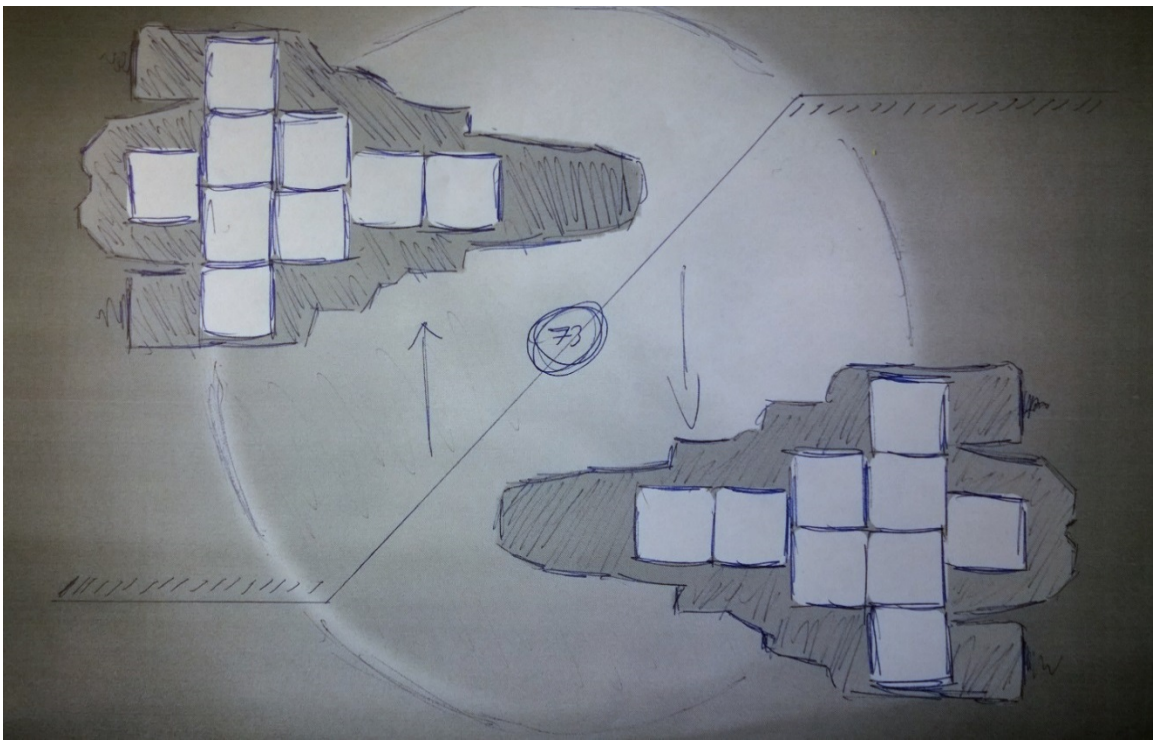

Figure 9. Rough sketch of the ship vs. ship combat of Starkillers.

Having distinctive art is also a big factor in attracting grip, and getting funded at a later time (Dube 2014). For this I'm going to draw on my web design expertise by employing flat design, an interface art style popularized by Windows 8. Flat design removes stylistic characters such

as drop shadows, gradients, textures, and any other type of design that is meant to make the element feel three-dimensional (fig 10). It is a very economical choice:

- Flat design feels crisp and modern, and allows one to focus on what is the most important: The content and gameplay.

- Flat design makes things more efficient and cuts out the fluff. Flat design is essentially the lean method of art.

- By stripping down design styles, flat design is a way to future-proof the art so it stays relevant for longer periods of time. (Turner 2014.)

Figure 10. Rymdkapsel makes good use of flat design: The game is as stylish as it is minimal. (Google Play 2013.)

Last but not least of the artistic considerations is the sound and music of the game. However, graphics are much more important than the audio aspect.

"Audio is the least important of the creative disciplines in indie games: I can think of many, many successful games which have utterly terrible sound and music." - Paul Taylor (Taylor 2012.)

It's a brutal thing to say, but for indie games audio is simply icing on the cake. As such, audio is the one discipline I'm willingly choosing to neglect. That does not mean not having a strategy: Lean choices in other categories help immensely to skimp on sound. For instance, the

sci-fi theme and minimal art style work well with a sparse soundscape, befitting the strategic gameplay. But the baseline is to not spend any remarkable time or money on audio.

4.3 Marketing, advertising & PR

Today, the indie game scene is very much mainstream. Plenty of games are released every single day. This means that even an amazing game will fail to turn a profit without appropriate promotional efforts. Since traditional methods such as advertisements or street banners are relatively expensive, this leaves the average indie developer in a tight spot. The zero-budget solution is to use organic methods to drive word of mouth, but this approach takes plenty of work to get right. (Larochelle 2014.) Building awareness for your game generally involves:

- Marketing: Generally Internet-based material that will get information about your game out. Concept art, screenshots, prototypes, articles, blogs – almost anything.

- PR: Any type of social interaction where you end up discussing something that might be related to your game. This could be discussing the game with your peers, networking at a game conference, or debating mechanics on a forum. (Toresson 2013.)

- Branding: Maintaining a consistent and professional company and product image (fig 11). Strive for a distinct style with everything visual: Game art and company logo alike. This way people will recognize you from the sea of games in the market and perhaps remember you at a later time. (Heikkinen 2014.)

Figure 11. Branding creates a uniform look for everything the company does. (Vetica 2014.)

As established in chapter 2, many indie developers skew their efforts towards production, diving headfirst into code, while others leave marketing for the last, developing in secrecy till the final stages. The lack of marketing is one of the biggest reasons why many independent studios fail to generate revenue and stay commercially viable despite having a high-quality product. Building awareness takes time and effort, and it is an ongoing responsibility that requires constant attention. (Taylor 2009.) Some concrete ways to start building awareness are having a:

- Homepage that's easy to navigate and has a presskit and blog as well as information, concept art, screenshots and video footage of your game. Use Google Analytics to keep track of visitors.

- Facebook page that's active and social. Encourage interaction by interacting with people yourself, and answer people directly. Post images and videos; they get the most attention and interaction. Debate and discuss news in games on appropriate Facebook pages you yourself enjoy; the visibility is beneficial for your developer or company account.

- YouTube account that has videos of gameplay as well as possible events, team introductions and other things of interest. Link to the channel in all your videos so that any viewers easily find more of what you've got, and be active in the comments section the same as you would on Facebook.

- Twitter account that follows people that you would want to read more from. The key here is to socialize: Don't be afraid to engage with people if you have an opinion that contributes to a discussion. This way you are being part of the community discussing the topic. (Toresson 2013.)

To provide content that attracts attention, it is essential to know what kind of people are reading about your game and what content they are looking for. Different people will be interested in your game at different times, and for varying reasons. To build a strong community throughout the development of your game, it is important to identify the three major groups that follow these kinds of things. These groups are:

1) Fellow developers: Your peers are the first group to approach. Even if you only have an idea or a mockup, fellow developers can still be excited about your game because they can recognize potential. Share ideas and development methods for quality feedback and early grip!

2) Gamers: As discussed in chapter 2.1, connect with your target audience via the minimum viable product. Enthusiastic early adopters will often remain a key asset in gathering feedback and promoting the game. Concept art and a functional prototype are essential in attracting more people from this group.

3) The press: Websites that write about video games will want to write a good article, which means that unusual game features or interesting development stories are the best approach. If members of the press like your game, they can dramatically increase its awareness. However, a gameplay video of tremendously high quality is often required to gain the attention of big publications. (Larochelle 2014.)

This manner of community building is a lean and efficient way to gain organic awareness. Talk to developers and launch your minimum viable product to attract early adopters; invite these people to try and criticize the product. Start a positive feedback loop by getting your community involved in the development process: Ask for feedback and follow the advice to improve the end-product. (Schweer 2012.)

4.4 Prototyping & testing

Imagine pitching today's smartphones to a 90's audience. Cellphones decades ago were very sturdy; the batteries lasted a very long time, and each handset had a physical keyboard. The idea of a modern phone – complete with a short battery life and fragile casing – would probably not be well received, because people simply wouldn't be able to imagine the features they're missing. Yet Apple confidently built their iPhone and took the cellphone market by a storm. That speaks of the importance of prototyping; sometimes, in order to sell something, you've got to build it first. (Tristem 2014.)

This is especially true for video games. Prototypes enable early testing and direct player observation. This is more efficient than verbal feedback because the customer won't know if they'll want the game made, even if it could be perfectly described in advance. (DeLeon 2011.) Prototypes lack the polish of a final product, and this is a great asset because the main attraction of video games is gameplay. You have to make sure that the core mechanics are fun and entertaining without additional context, and there's no better way to confirm this than via a prototype. (Adams 2012, p. 15.) Ultimately, you want a strong core to drive development and act as the center of product. When moving to production work, this will make it easier when you have to cut or refine features – you'll always have something to fall back on. It's also possible to discover new and fun mechanics while prototyping. Sometimes these features are even better than the original core; in this case, consider replacing the core or saving the ideas for the next project. (Yu 2010.)

At this point in development it's important not to focus too much on code quality, because prototype code is meant to be thrown away. Writing code of too high a quality will make the process slower for too little gain. As with many other aspects of lean development, cut corners whenever you can. Only develop laborious features, such as online play, if it is absolutely necessary to do so; such components can often be left for the production stage. (Heinäpurola 2014.)

There are two different methods of creating a prototype: They are called the horizontal and vertical slice (fig 11). Building a horizontal slice means creating a low-fidelity version where the game is playable and includes the core gameplay mechanics, but is very sparse in presentation. At first, the horizontal slice only includes the most rudimentary graphics and sound assets. The slice is then widened over time by building new content and adjusting features

according to player feedback. The vertical slice does the opposite: It is a very small part of the game that looks and acts like it were the final product. But it's only a small portion intended to prove mass market interest before going into production. (York 2012.)

Games with discrete mechanics were discussed in chapter 2.4.4, and one of the strengths of this approach reveals itself at this point: Although most game prototypes are digital, such software is relatively slow to create. A faster alternative is to build a paper prototype. Because discrete game mechanics often do not require precise timing or physics calculations, it means the core gameplay can be constructed as a board game. (Adams 2012, p. 17-18.) This is exactly what I have done (fig 12).

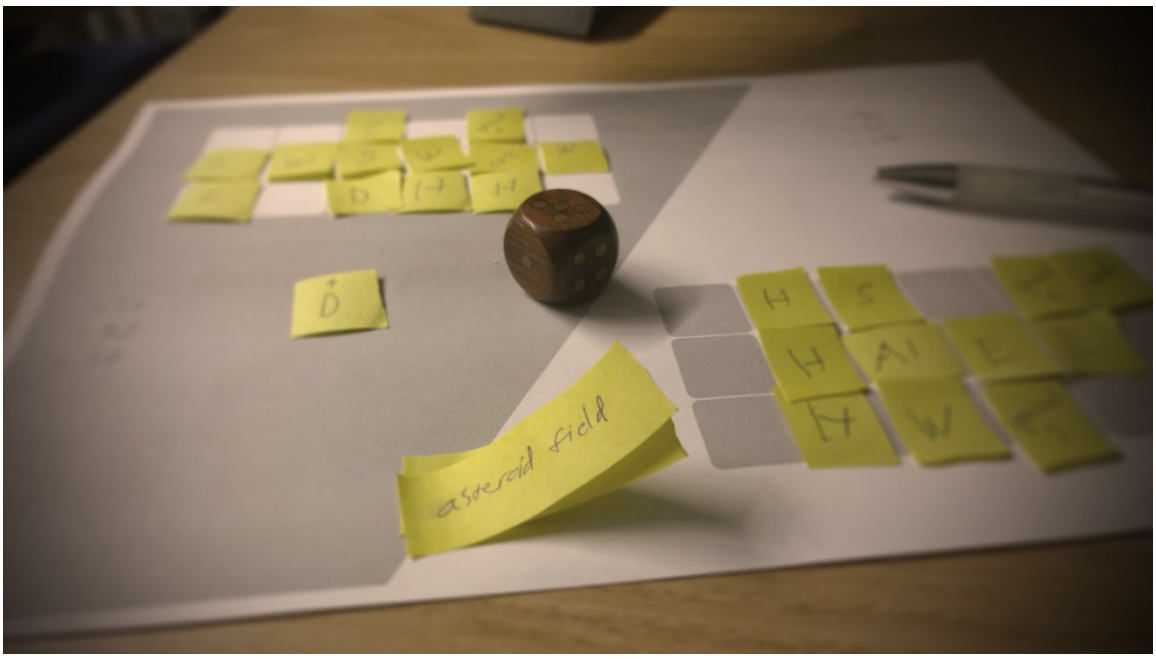

Figure 12. The paper prototype of Starkillers.

It takes some imagination to test game mechanics in this manner, but throughout this thesis I have repeated the mantra: Use whatever means necessary. The physical prototype isn't meant to replace the need for a digital version. However, paper prototypes are blinding fast to make, and they help affirm some of the broadest design decisions before bringing the game to motion on the computer screen. This is what iteration is all about. The previous chapter established the three phases of marketing iteration: From peers to people to press. I want to line up my content creation in a similar manner. The paper prototype combined with the early sketches aren't much to look at, but they're enough to discuss and test the game rules with friends and other developers. Painstakingly crafted concept art is next; this is used to approach

the target market. If the reception is good, the digital prototype will follow. If not, I will go back to the drawing board.

Engaging the end user as early as possible is a foundational tenet of the lean process. Over a duration, it is possible to create a kind of a tribe around the game. People will want to play the game, share information, and see more of it. (Schweer 2011.) This is essential: It is nearly impossible for an independent game developer to succeed without this loop. Seek criticism way before you're comfortable showing your game to anyone. (Rose & Short 2014.)

4.5 Funding

By now it has been well established that game development is a huge ordeal. Making a successful game often takes years of work. However, this also means it will take all that time before seeing any kind of significant financial return from the project. That is often far too long to work solely on money saved upfront, which means you will need to find external funding before going all-out. (Taylor 2012.)

That said, the best time to take someone else's money is never. No matter how good a deal, there are always strings attached (Della Rocca 2013). When financing a project, you're essentially trading future sales in exchange for getting plenty of money today – often at a deficit (Paffrath 2013). Instead, many would-be indies moonlight in big game companies or in an entirely different industry; some are students and live on benefits. The point is that the regular paychecks fund the independent project in its current state, and this way there is practically zero financial risk involved. You should do this as far into pre-production as possible. (Della Rocca 2013.) On the other hand, external funding allows the developer to focus all of his efforts on the game project, and this will become increasingly important as the project moves into the production stage. You don't want to work part-time forever, or else the game will take too long to develop; you risk losing the window of opportunity. (Paffrath 2013.)

Being a professional indie game developer – even just a one-man-band – means that you are essentially running a small business. However, it's important not to rush into founding a start-up prematurely. Without a company, you don't need to pay company taxes, have an accountant, or spend time on paperwork. Focus on building something first. Go through the motions of pre-production: Develop a minimum viable product, build a community, and partner up

with people. Only open a company once you've gained significant traction for your project. The goal is to mitigate risk till the zero hour. (Paffrath 2013.)

Among other things, you can't raise funds until you've registered a company, so it has to be done eventually. But the transition from self-sufficient to externally-funded should be considerate. The second best time to take someone else's money is when you don't need it; rushing only leads to desperation and bad deals. There are multiple funding options that should be evaluated well in advance. Again, no decision is isolated from the rest. Consider the following:

- Publishers: The original piggy bank for game developers, and still a viable option for funding production. However, pitching is difficult and requires a very precise strategy.

- Incubators and accelerators: These programs exchange a small chunk of funding, mentorship and coaching in exchange for some equity in your company.

- Investors: Plenty of money, fast, but they usually take the most equity and often require a board seat, and apply the most pressure to grow your business.

- Crowdfunding: With the rise of Kickstarter and Indiegogo, going directly to your fans for funding is ever more viable. However, creating a successful campaign is a huge effort. (Della Rocca 2013.)

Publishers have always been the go-to option for video game funding, but publishers are notorious for chasing trends. In 2014, there is one thing that all publishers will ask: "Is this a free-to-play game?" A game will be rejected the instant it is discovered that it is not free-to-play. (Fassihi 2014.) This clashes with practices established in chapter 3.6 as well as the design decisions made thereafter. As for incubators, the requirements are less strict and the deals are often favorable, but the money is small-time. This alternative is more of a pit stop than a finishing line. Investors on the other hand want a hefty share of equity; they often require a board seat, and provide the most pressure for growth. It is a high-stakes road to take, and requires keen business sense not common among game developers. (Della Rocca 2013.)

With the ever increasing popularity of the indie scene, going directly to your fans for funding is perhaps the most favorable option (Della Rocca 2013). It sounds great on paper: Money and awareness for no equity at all. However, crowdfunding isn't the free money well it used to be. Today, you have to look at crowdfunding services, such as Kickstarter, as a traditional publisher-developer relationship. Just like pitching your game to a publisher, you need to sell

your game to your prospective customers. You need to answer the question, "Why should people support my project?" (Wester 2013.) The same as you can't pitch a bare-bones idea to a publisher, you can't run headlong into a Kickstarter campaign and expect success. It simply isn't realistic to ask for money in exchange for something you haven't proven to be worth it. (Sarkar 2014.)

For this reason, funding is the final part of pre-production. You need to have established a presence on social media months before you launch on Kickstarter, You have to have a following; to attract a following, you have to have striking art and an exciting prototype. It's a virtuous loop that needs time and dedication to get moving. Know your audience, look ahead, and make sure the way you're taking the project makes sense. For instance, almost all of the games on Kickstarter are premium games for PC. Casual, mobile and free-to-play games do not fare well in crowdfunding. (Deszczulka 2014.)

Notice that it's no coincidence that I plan to bring a desktop game to this indie-dominated channel. There's no serendipity here; just decisions based on careful analysis. Independent developers can't afford to depend on luck, and there's no reason to do so when there's a wealth of information out there. Finally, funding a project is a full-time job. Such a campaign is an all-out marketing and community management effort. For this period of time, you can forget about development. (Dube 2014.)

5 CONCLUSION

You don't need qualifications, AAA-experience or a college degree to be a game developer. "Going indie" is a video game industry term for moving from whatever one is doing now to being a full-time independent game developer, and there are no artificial barriers to stop anyone from becoming a professional game developer. However, the process of making and releasing a video game features an intimidating list of crafts. Worse yet, each of these is highly challenging to master. An independent developer certainly has his work cut out for him, and at first glance there seems to be more work than one can ever hope to finish.

This is the reason many aspiring developers can spend incredibly long hours on their project yet see very little progress, with no end in sight. The freedom to jump straight into code is the undoing of many newcomers: Budding developers simply don't have the money or experience to push through production by force. But if the right decisions are made in the early stages of the project, it is well possible for a small-timer to find indie development success. Most of the hard choices are made during the first part of game development – pre-production – and hence that was the focus of this thesis. Pre-production is further divided in two distinct sections: In the concept part of pre-production, such long-term decisions are made that they must be adhered to for the rest of the project's duration. These are project-defining aspects: Genre, platform, target audience, and so forth. The execution phase elaborates and builds on top of these decisions but doesn't change them.

There are two overarching goals during pre-production: To build awareness, and to develop the foundations for a great product. Building awareness is vital because even a game of high quality will fail to turn a profit if nobody knows about it or nobody wants it. Since independent developers rarely have the budget necessary for traditional marketing methods, they often rely on organic growth via community building and word of mouth. This is where the lean production philosophy of the start-up business scene provides an excellent framework. The most valuable aspect of this method is that it incorporates customers into the development process, where the developer can confirm as early as possible that he's working on a game that people will want to pay for. Lean methods are also used to design a game that is as captivating as it is cheap to manufacture, and plays to the strengths of its creator. This results in plans for a simple-but-deep game that is designed around the developer's current skillset and circumstances.

In this thesis, I followed these steps with examples of my personal game project – all the way to prototyping. However, this illustrates only one way of going through the process. It is stressed on multiple occasions that each choice – whether of genre, platform, or source of funding – must make sense in relation to every past and future decision. Funding is the final part of pre-production, and I have yet to take that step myself. Funding a game project takes time and patience. Each aspect in the execution phase must be iterated till it reaches acceptable levels of quality, and the highly competitive market has set this bar high. However, if these goals are met, the project has a chance of moving to the production stage of game development.

Production is all about hard work: Different values, such as motivation and discipline, become much more important than before. But if the pre-production stage is a success, the developer can always rest assured that there are fans eager to play his game. This kind of certainty is the single biggest asset for an independent game developer – to know that his work will be successful.

SOURCES

Adams, E. 2009. Fundamentals of Game Design, Second Edition. United States: New Riders

Adams, E. 2012. Game Mechanics: Advanced Game Design. United States: New Riders

Ries, E. 2011. The Lean Startup. United States: Crown Business

Rogers, S. 2010. Level Up!: The Guide to Great Video Game Design. United Kingdom: John Wiley & Sons, Ltd.

Schell, J. 2008. The Art of Game Design: A Book of Lenses. United States: Elsevier Inc.

Online Sources

Barnson, J. 2013. Why are most indie games 2D instead of 3D? Available at: www.rampantgames.com/blog/?p=5934 (Accessed 17.11.2014)

D'Angelo, D. 2014. Digging down to business: Shovel Knight planning and sales. Available at: www.gamasutra.com/blogs/DavidDAngelo/20140805/222585/Digging_down_to_business_Shovel_Knight_Planning_and_Sales (Accessed 5.8.2014)

DeLeon, C. 2009. Resourcefulness vs resources. Available at: www.hobbygamedev.com/articles/vol2/resourcefulness-vs-resources (Accessed 24.6.2014)

DeLeon, C. 2010. General concepts for beginning game developers. Available at: www.hobbygamedev.com/beg/general-concepts-for-beginning-game-developers (Accessed 7.6.2014)

DeLeon, C. 2011. Colorful oceans and chunky sauces. Available at: www.hobbygamedev.com/int/oceans-and-sauces (Accessed 12.11.2014)

Della Rocca, J. 2013. A brief introduction to project and studio financing options for indies. Available at: www.gamasutra.com/blogs/JasonDellaRocca/20130730/197257/A_Brief_Introduction_to_Project_and_Studio_Financing_Options_for_Indies (Accessed 5.9.2014)

Deszczulka, L. 2014. 5 things we learnt from Kickstarter after removing our game. Available at: www.gamasutra.com/blogs/LukaszDeszczulka/20141028/228751/5_things_we_learnt_from_Kickstarter_after_removing_our_game (Accessed 28.10.2014)

Dube, W. 2014. How an indie nobody raised $40,000 on Kickstarter. Available at: www.gamasutra.com/blogs/WilliamDube/20140814/223358/How_An_Indie_Nobody_Raised_40000_On_Kickstarter (Accessed 7.11.2014)

Fassihi, A. 2014. Postmortem: Shadow Blade. Available at: www.gamasutra.com/blogs/AmirHFassihi/20140410/215190/Postmortem_Shadow_Blade (Accessed 6.9.2014)

Fischer, J. 2014. Competitive advantage and the productivity frontier, or why Dark Souls is the Ikea of game development. Available at: www.gamasutra.com/blogs/JustinFischer/20140430/216624/Competitive_Advantage_and_the_Productivity_Frontier_Or_Why_Dark_Souls_is_the_Ikea_of_Game_Development (Accessed 21.7.2014)

Gile, C. 2014. The benefits of less detail. Available at: www.gamasutra.com/blogs/ChristopherGile/20141112/229910/The_Benefits_of_Less_Detail (Accessed 17.11.2014)

Heikkinen, H. 2014. Marketing mobile apps. Available at: http://henrihei.github.io/MarketingMobileApps (Accessed 16.8.2014)

Heinäpurola, T. 2014. Prototyping and code quality. Available at: www.gamasutra.com/blogs/TimoHeinapurola/20140502/216870/Prototyping_and_Code_Quality (Accessed 5.9.2014)

Henshell, T. 2014. Why I've said goodbye to mobile in favor of PC. Available at: www.gamasutra.com/blogs/ThomasHenshell/20140807/222732/Why_Ive_Said_Goodbye_to_Mobile_in_Favor_of_PC (Accessed 7.8.2014)

Hindman, B. 2011. Free for all: another attempt at free-to-play vs. freemium. Available at: http://massively.joystiq.com/2011/09/07/free-for-all-another-attempt-at-free-to-play-vs-freemium (Accessed 4.7.2014)

Håkansson, O. 2014. Game design deep dive: The digging mechanic in SteamWorld Dig. Available at: www.gamasutra.com/view/news/225257/Game_Design_Deep_Dive_The_digging_mechanic_in_SteamWorld_Dig (Accessed 10.9.2014)

Ismail, R. 2014. Game a week: Getting experienced at failure. Available at: www.gamasutra.com/blogs/RamiIsmail/20140226/211807/Game_A_Week_Getting_Experienced_At_Failure (Accessed 13.5.2014)

Lahti, J. 2014. Myy ensin edes jotain. Available at: www.tuplaamo.fi/2014/03/myy-ensin-edes-jotain (Accessed 5.7.2014)

Larochelle, S. 2014. The 3 fans that make or break your indie game community. Available at: www.gamasutra.com/blogs/SalimLarochelle/20140613/219229/The_3_Fans_that_Make_or_Break_your_Indie_Game_Community (Accessed 16.8.2014)

Natanson, E. 2013. Mobile game development: When to think of monetization? Available at: www.gamasutra.com/blogs/EladNatanson/20131009/201918/Mobile_Game_Development_When_to_Think_of_Monetization (Accessed 4.7.2014)

Nutt, C. 2014. RPG balancing wisdom from an Obsidian veteran. Available at: www.gamasutra.com/view/news/224020/RPG_balancing_wisdom_from_an_Obsidian_veteran (Accessed 22.8.2014)

Obenchain, L. 2013. A process for mobile game development. Available at: www.pilotnorth.com/mobile/a-process-for-mobile-game-development (Accessed 6.11.2014)

Oedekoven, K. 2011. Getting lean in your design and development. Available at: www.korpg.com/blog/getting-lean-in-your-design-and-development (Accessed 27.5.2014)

Paffrath, R. 2013. Opinion: What NOT to do when starting as an indie game developer. Available at: www.indiegames.com/2013/11/opinion_what_not_to_do_when_st.html (Accessed 6.11.2014)

Patel, J. 2014. Why do you need a precise game design document? Available at: www.openxcell.com/need-precise-game-design-document (Accessed 15.8.2014)

Powers, M. 2014. One for the designers. Available at: www.gamasutra.com/blogs/MattPowers/20140517/217512/One_for_the_Designers (Accessed 25.6.2014)

Ramachandran, A. 2014. The most important game design lesson I ever learned. Available at: www.bigfatphoenix.co.in/2014/03/the-most-important-game-design-lesson-i.html (Accessed 6.11.2014)

Rose, M. 2013. How to talk to the video game press in 2013. Available at: www.gamasutra.com/view/news/198381/How_to_talk_to_the_video_game_press_in_2013 (Accessed 8.6.2014)

Rose, M. & Short, T. 2014. 5 tips for making a cross-platform game for mobile and PC. Available at: www.gamasutra.com/view/news/220349/5_tips_for_making_a_crossplatform_game_for_mobile_and_PC (Accessed 7.7.2014)

Rossi, M. 2013. So, you wanna be a game designer? Available at: www.wannabe.urustar.net (Accessed 20.6.2014)

Ryan, T. 1999. The anatomy of a design document. Available at: www.gamasutra.com/view/feature/3384/the_anatomy_of_a_design_document (Accessed 25.6.2014)

Sarkar, S. 2014. How Superhot's playable prototype led to Kickstarter success in one day. Available at: www.polygon.com/2014/5/22/5738846/superhot-kickstarter-interview-success-playable-prototype (Accessed 10.6.2014)

Schweer, E. 2011. Bootstrapper's guide to game development. Available at: www.indiebits.com/bootstrappers-guide-to-game-development (Accessed 4.5.2014)

Schweer, E. 2012. 9 steps to self publishing your first indie game. Available at: www.indiebits.com/self-publishing-steps (Accessed 10.7.2014)

Schweer, E. 2012. You don't need qualifications to be a game developer. Available at: www.indiebits.com/you-dont-need-qualifications-to-be-a-game-developer (Accessed 1.5.2014)

Taylor, P. 2009. Building buzz for indie games. Available at: www.gamasutra.com/view/feature/4117/building_buzz_for_indie_games (Accessed 16.8.2014)

Taylor, P. 2012. How to be an indie game developer. Available at: www.mode7games.com/blog/2012/06/12/how-to-be-an-indie-game-developer (Accessed 3.5.2014)

Toresson, J. 2013. Indie game marketing: A love story – part 1 (getting a solid base). Available at: www.gamasutra.com/blogs/JohanToresson/20131021/202762/Indie_Game_Marketing_A_love_story__Part_1_Getting_a_solid_base (Accessed 16.8.2014)

Tristem, B. 2014. Why early feedback is so valuable, and not just the money. Available at: www.gamasutra.com/blogs/BenTristem/20140806/222757/Why_early_feedback_is_so_valuable_and_not_just_the_money (Accessed 6.8.2014)

Turner, A. 2014. The history of flat design: How efficiency and minimalism turned the digital world flat. Available at: www.thenextweb.com/dd/2014/03/19/history-flat-design-efficiency-minimalism-made-digital-world-flat (Accessed 18.10.2014)

Uribe, S. 2014. Postmortem of Flip, by Perro Electrico, or how you can make sure you will not make money on your game. Available at: www.gamasutra.com/blogs/SebastianUribe/20140520/218157/Postmortem_of_Flip_by_Perro_Electrico_or_how_can_you_make_sure_you_will_not_make_money_on_your_game (Accessed 10.6.2014)

Venturelli, M. 2014. Constraining the space of possibility. Available at: www.gamasutra.com/blogs/MarkVenturelli/20140828/224398/Constraining_The_Space_of_Possibility (Accessed 1.9.2014)

Wester, F. 2013. Is Kickstarter even worth it for indie game developers? Available at: www.gamasutra.com/blogs/FredrikWester/20130904/199571/Is_Kickstarter_Even_Worth_It_for_Indie_Game_Developers (Accessed 6.9.2014)

York, T. 2012. Making lean startup tactics work for games. Available at: www.gamasutra.com/view/feature/168647/making_lean_startup_tactics_work (Accessed 13.5.2014)

Yotes, T. 2013. A college student's game development process. Available at: www.gamasutra.com/blogs/TonyYotes/20131105/204055/A_College_Students_Game_Development_Process (Accessed 20.6.2014)

Yu, D. 2010. Finishing a game. Available at: www.makegames.tumblr.com/post/1136623767/finishing-a-game (Accessed 7.6.2014)

Figure Sources

Brouwer, E. 2013. LEGO Sydney Opera House 10234 unveiled! Available at: www.alegoaday.com/lego-sydney-opera-house-10234-unveiled (Accessed 12.11.2014)

Grapefrukt games. 2013. Rymdkapsel. Available at: https://play.google.com/store/apps/details?id=com.grapefrukt.games.rymdkapsel1 (Accessed 12.11.2014)

Langford, J. 2012. Review: XCOM: Enemy Unknown (PS3). Available at: www.psnation.com/2012/10/08/review-xcom-enemy-unknown-ps3 (Accessed 12.11.2014)

Stone, T. 2012. FTL: Faster Than Light. Available at: www.pcgamer.com/ftl-faster-than-light-review (Accessed 12.11.2014)

Superhot Team. 2014. SUPERHOT – an FPS where time moves only when you do. Available at: www.superhotgame.com (Accessed 12.11.2014)

Tan, M. 2011. Review: Frozen Synapse. Available at: www.destructoid.com/review-frozen-synapse-202827.phtml (Accessed 12.11.2014)

Vetica. 2014. Sapido. Available at: www.vetica-group.com/projects/detail/project/c/show/sapido (Accessed 12.11.2014)