Universidade de Aveiro
2021

JOSUÉ ANTÓNIO OLIVEIRA SILVÉRIO

Design de Engagement para Desenvolvimento de Jogos Indie

Engagement Design for Indie Game Development

Universidade de Aveiro
2021

JOSUÉ ANTÓNIO OLIVEIRA SILVÉRIO

Design de Engagement para Desenvolvimento de Jogos Indie

Engagement Design for Indie Game Development

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Comunicação Multimédia, realizada sob a orientação científica do Doutor Nelson Zagalo, Professor Associado do Departamento de Comunicação e Arte da Universidade de Aveiro

Dedico este trabalho à minha mãe que esteve sempre aqui para mim.

agradecimentos

Agradeço a todas as pessoas que fizeram com que este projeto fosse possível.

Agradeço ao professor Nelson Zagalo pela orientação científica e por todas as oportunidades que me deu, dentro e fora deste projeto.

Quero agradecer à minha mãe, que sempre esteve presente e que sempre fez de tudo para que eu pudesse estudar. Se estou aqui hoje é pelo trabalho dela.

Um agradecimento ao resto da minha família, que sempre me apoiou ao longo da vida.

Um agradecimento a todos os meus amigos, velhos ou novos, longe ou perto, físicos ou digitais, atuais ou passados, por me terem moldado e por terem feito com que os meus interesses tenham sido estes. Sem todas estas pessoas e todas as experiências que me proporcionaram não seria certamente a pessoa que sou hoje.

Um agradecimento especial aos meus amigos de NTC que, ao longo de uma licenciatura e de um mestrado repletos de trabalhos, noites mal dormidas, saídas até horas responsáveis, saídas até horas menos responsáveis, cafés que se estendem e muitas horas de risos, felicidade, e muitas mais emoções me acompanharam, acompanham e certamente vão continuar a acompanhar.

palavras-chave

Design de Jogos; Jogo Indie; Desenvolvimento; Engagement; Taxonomia de Jogadores

resumo

Os jogos indie surgiram como resposta ao espaço temático deixado livre pela indústria dos jogos AAA, preenchendo assim nichos. Isto significa que, normalmente, os jogos indies acabam por se perder num mar imenso de produtos listados nas lojas de jogos online. A agravar este problema temos o facto de que quando se desenha um jogo para um mercado de nicho, temos em vista também um público de nicho, reduzido. A forma de chegar a diferentes públicos tende a provocar receios nos criadores de jogos de poderem ver as suas obras adulteradas em função de objetivos comerciais.

O objetivo deste projeto passava por tentar perceber se pelo recurso ao Engagement Design Model, seria possível fazer várias as experiências de engagement, e consequentemente ampliar os públicos, utilizando o mesmo universo criativo. O modelo facilitaria o processo de definição das audiências, dando aos desenvolvedores mais tempo para outras tarefas.

Do trabalho realizado, concluímos que é possível ajudar os criadores indie, apenas pela via de ajustes no Engagement dos seus jogos sem ter de mudar a sua direção criativa, dando-lhes a possibilidade de fazer chegar a sua mensagem a mais jogadores, sem comprometer a sua visão.

keywords

Game Design; Indie Game; Development; Engagement; Player Taxonomy

abstract

Indie games have appeared as a response to the spaces left open by the AAA game industry, filling niches. This means that, more often than not, a lot of indie games get lost in the sea of products listed in today's online stores. Adding to this problem we have the fact that, when designing a game to fill a niche space, we will have a niche audience, which is, almost always, reduced in size and indie developers are afraid to tweak their games to reach more people because they do not want to give up on their creative vision.

This project aims to find what lessons can we learn by using the Engagement Design Model, which shifts the focus from experiences to engagement, when designing and developing an indie game. Thus, speeding the process of defining audiences, giving indie developers more time for other tasks that usually lack in quality, like marketing.

With these lessons, we also hope to help indie developers understand that it is possible to tweak their games' engagement without having to change their creative direction, giving them the possibility to deliver their messages to more people without compromising on their vision.

Table of contents

Table of figures

1. Introduction

The shrinking space of AAA games in the gaming scene opened up niches for indie developers to explore and increased even further the creative freedom of smaller studios.

Although indie games gain by filling niche spaces left by AAA games, because these games cater to a niche audience, they can have problems reaching a diverse set of players. This is not always a problem, as a game should have a defined audience and, most of the times, does not have the need to reach every player, but developers can, sometimes, be limited to a very small niche audience that might not be numerous enough to generate revenue at a level that will let developers live comfortably, or in some cases, might not even cover the cost of the development.

When we talk about game design we have to be aware that we are designing for people. Players are grouped by various traits like gender, age, and other demographic qualities. These are external factors but, when discussing player taxonomy, we usually refer to internal factors –these are known as psychographics (Schell, 2008) – where they find pleasure. We can find numerous models that group and describe players according to the experiences they seek while playing a videogame.

In Indie game development we usually see a lack in care for the user and little user-centric design (Schell, 2008) when compared to other disciplines of interaction design, although there are a number of different models for player taxonomy available. This is related to the fact that, more often than not, indie games are created to fill a niche space, as mentioned above, which leads to a very reduced audience that is also very hard to define.

This niche way of thinking can also lead to developers closing off their game to potential publics because they might have to change their creative visions to encompass a larger audience, since most player taxonomy models are created with experience in mind.

Interactive media pieces are not only products; they have messages and certain needs beyond design; but design impacts and transforms messages. To better the interaction design of these media there have been approaches to Experience as part of HCI, but this has not been able to deliver the needs of design. The Engagement Design Model (Zagalo, 2020) proposes, based on these limitations, a shift from experience to engagement design on the research of interactive media, namely games.

This shift is what motived this project – the design and development of a new videogame using the ED Model as a design tool. When in comparison to other player taxonomy models the Engagement Design Model presents itself as a more hand-on and practical model and gives us guidelines on how to implement each engagement stream (Zagalo, 2020), instead of simply describing a player type.

2. Research Question

Having the problem and goals in mind, the following question was outlined making sure it was feasible, pertinent, and clear (Quivy & Campenhoudt, 2008).

What lessons can we take from the Engagement Design Model when designing the same experiential universe for different types of players?

The main objective for this research is to find if we can use the Engagement Design Model for player taxonomy while designing a videogame to create the same experience for different types of players. The research aims to understand if the player profiles found in the model can be used to better design games.

3. Project Aims and Goals

The aim of this project is to design and develop a game aimed for all types of players, using the Engagement Design Model for player taxonomy, and learn from the process. Thus, the objectives for this project are:

a) Determine what lessons can be taken for indie developers so they can open their game up for more people, without having to compromise their creative vision, based on the documentation of the process.

The documentation of the process will help us find what lessons can be found, and applied to indie game development projects, when using the Engagement Design model. With these lessons, indie game creators can reach a broader market while minimizing the work load needed to cater to specific audiences.

b) Designing a videogame with three different engagement types, mirroring the three different streams present in the Engagement Design Model.

The Engagement Design model defines three streams of engagement – Progression, Expression and Relation. For this product we have to design and develop three distinct parts that mirror these three streams, in order to have a game that can appeal to all types of player using only one experiential universe.

c) Documenting the design and development process, connecting the decisions with the Engagement Design Model.

After the design of the three parts of the game starts the development of said game. This game will answer the need provided by each engagement stream and show the results of using the Engagement Design model during design and development.

Finally, although testing with users is important to help validate the lessons found through the analysis of the design and development process, due to the available time-frame and the already big complexity of the project, this will not happen and the conclusions derived from the project will be solely self-analytical.

4. Theoretical Framework

4.1. Indie

4.1.1. Indie Culture

The term Indie Game came from the mid-to-late 2000's game revolution, being borrowed from music and cinema. The definition of indie game is ambiguous. What once meant that a game was made by independent teams can now mean creative, emotional, or made by a small team.

The 2000's brought us the growth of the internet and, in time, online distribution. Online digital stores gave developers the tools to distribute their games directly to their audience, without the need for a middleman, thus eliminating the need for a publisher. It was also at this time that middleware and game engines started gaining popularity, making game development more accessible to hobbyists and newcomers to the industry.

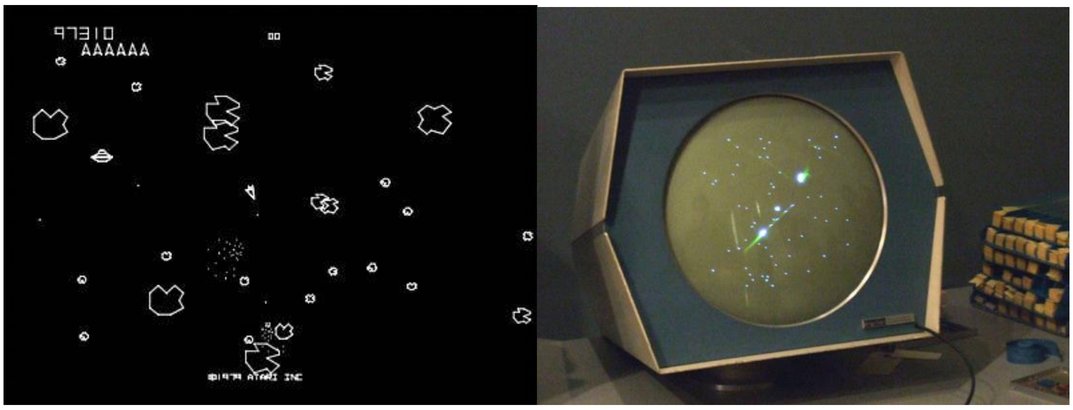


Figure 1 - Asteroids (Logg & Rains, 1979) & Spacewar! (Russell et al., 1962)

The growth of the internet also made it possible for consoles to have online functionality included and brought us platforms for crowdfunding, giving power to smaller teams and helping them get their projects funded. All of this factored in to help the great indie game renaissance of the 2000's. This does not mean that indie games appeared at this time. Indie games did not start with games like Super Meat Boy (McMillen & Refenes, 2010), Braid (Blow, 2008) or Cave Story (Amaya, 2004) but can be traced back to games like Asteroids (Logg & Rains, 1979) and Spacewar! (Russell, Graetz, Samson, & Witaenem, 1962), games developed by one person or small teams. One can say all games were indie games at that time, as there were no big publishers yet.

It is said that with the videogame crash of 1983 the indie game scene disappeared but, in the Indiecade keynote "State of the Union" (Foddy, 2014), Bennett Foddy states that this indie game spirit just migrated to computers. If we look at the current definitions of indie, we can find lineages that trace further back than the 2000's. We can find development tools as far back as 1983 with level editors in Lode Runner (Smith, 1983) and game development software like Garry Kitchen's Gamemaker (Kitchen, 1985) and Game Maker (Overmars, 1999) in 1999. Also in this year the game Counter-Strike (Le & Cliffe, 1999) came out, developed by two people as a mod for Half-Life (Laidlaw, 1998). There were also modding communities for several games and this year also marks the beginning of flash games.

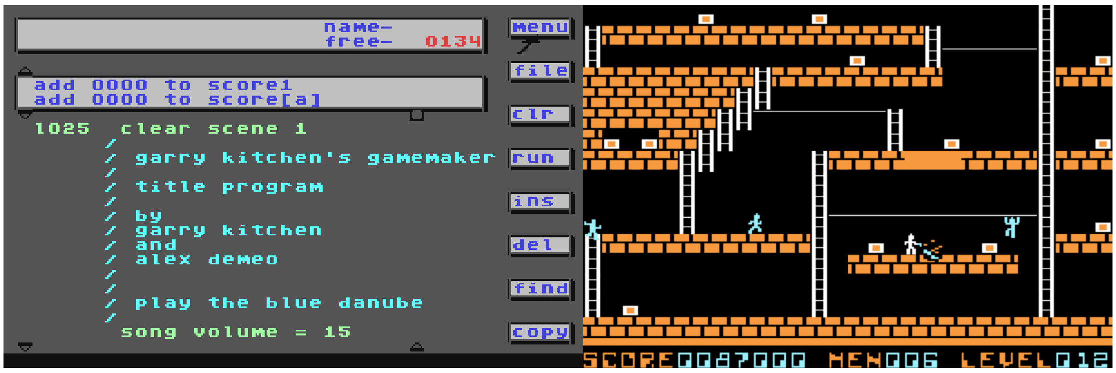

```
                        name-
                        free-      0134
add 0000 to score1
add 0000 to score[a]
1025  clear scene 1
      /
      / garry kitchen's gamemaker
      /
      / title program
      /
      / by
      / garry kitchen
      / and
      / alex demeo
      /
      /
      / play the blue danube
      /
      song volume = 15
```

Figure 2 - Lode Runner (Smith, 1983) *& Garry Kitchen's Gamemaker* (Kitchen, 1985)

Indie game distribution also predates the renaissance of the 2000's, not starting with platforms like Steam and Xbox Live Arcade. There were magazines that shared the code for games, public-domain games, and freeware, distributed through forums and online communities. Cases like Wizardry (Greenberg & Woodhead, 1981) also started appearing: games being developed by small teams that then sold the rights to the game to bigger companies. Current game publishers like Epic Games and Rockstar also started as indie studios, changing their ethos as they grew.

This spirit of creativity and independence was always present in the gaming scene – the 2000's only brought indie to the spotlight and to mainstream popularity. Indie Games are the foundation of the gaming world.

Jonathan Blow, the creator of Braid and The Witness, came to be what symbolises the epitome of indie game developer. With Braid, Blow went against what was conventional at the time and, because of that, Blow had to find alternative means to fund his project, including using his own money. The game was critically acclaimed, showing what was possible when someone was willing to take creative risks: with time manipulation puzzles the game was cryptic and hard, but also meaningful and beautiful. Blow took the messages he wanted to transmit to the players and tied them to the core gameplay. Here we can find another definition of indie games – "exploring more subtle and meaningful ideas".

Journey (Hunicke, Clark, Singh, & Bell, 2012) relit the "games as art" discussion, being an interactive hero's journey, having as an objective resounding in an emotional way with the players. Everything about this game screams indie: it was made by a small team of former students; it has a non-conventional approach to design, designing for emotions; and it looked like a non-viable product. This being said, we cannot truly say that Journey is an independent game in the traditional sense, since the team had an agreement with Sony to publish three of their games. Just like what happened with Portal (Swift, 2007), a small team of students received support from a bigger company, giving them time and money to bring their ideas to life.

Figure 3 – Braid (Blow, 2008) *& Journey (Hunicke et al., 2012)*

On the other hand, there were also a lot of teams that gained by not joining big publishers. The creators of Thumper (Flury & Gibson, 2016) left a company environment to make a game that explores the meaning of music, and just like indie games break the moulds in the game industry the music in this game distances itself from regular 4/4 songs, potentiating their creative abilities.

Another team that rejected corporate interference was Subset Games, a small team of two people and the creators of Faster Than Light (Ma & Davis, 2012), a spaceship management strategy game with rogue-like elements, and Into The Breach (Ma & Davis, 2018), a turn-based strategy game where the players control a small robot army, – procedural generated games that take inspiration from Rogue (Toy, Wichman, Arnold, & Lane, 1980) and Elite (Braben & Bell, 1984), having rogue-like elements.

Another game that has become and epitome of rogue-like games and of procedural generated games is Spelunky (Yu, 2008), made by three people. Spelunky is a platforming game that takes inspiration in spelunking ruins and, using a procedurally generated environment, lets the player explore ruins and caves in order to find jewels and other artefacts. Spelunky marked the resurgence of procedural generation in games and has since become one of the most important indie games.

These cases show us that the indie spirit can be present, even if a team receives support from a big company to develop the game.

In the following chapters we will discuss different topics related to indie games and their history in an attempt to give shed some light on this topic that, despite being very big in the gaming scene, is not always greatly understood.

4.1.2. Triple AAA Versus Indie

Recently, the game industry has seen a great increase in the number of independent titles. Once, the retail market was impenetrable to smaller studios, but digital distribution has made it easier for indie titles to reach a larger audience. This growth brought into question the size of a development team – why pay a large team of fifty people when you could have a smaller team of only five?

Minecraft (Persson & Bergensten, 2011) was initially developed by one person, and has since become the best-selling videogame, surpassing even Tetris (Pajitnov, 1984), and is present in all consoles and mobile. Other games like Braid (Blow, 2008) and Gone Home (Gaynor, 2013) have been highly praised for their production value, despite being made by small teams.

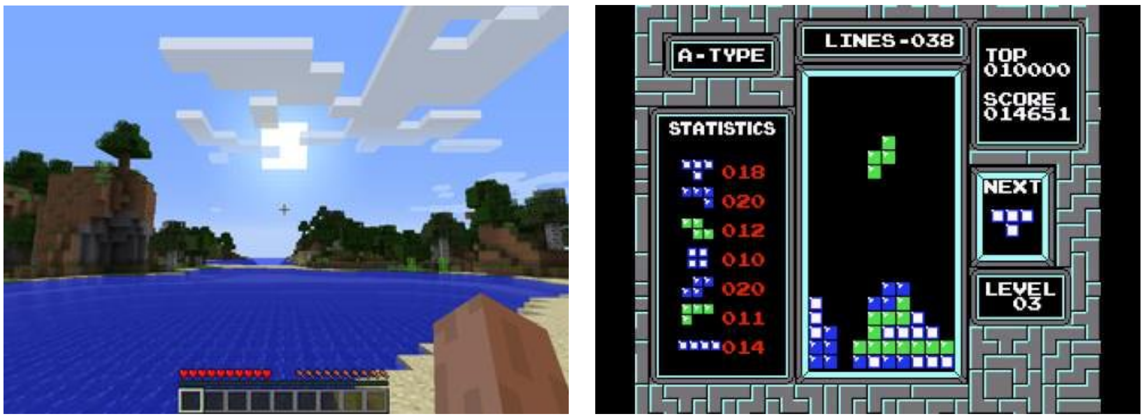


Figure 4 - Minecraft (Persson & Bergensten, 2011) & Tetris (Pajitnov, 1984)

You can say that the appeal of indie games stems from their ability to redefine genres, and it is this sense of ingenuity that makes it possible for these smaller teams to compete against the industry giants. The creativity that comes with indie games often comes from the reduced cost, and subjacent risk, when compared to AAA games. Thomas Was Alone (Bithell, 2012) cost less than 6000€ to develop, in comparison to games like Watch Dogs (Bélanger, 2014) and Titanfall (McCandlish & Alderman, 2014), that had a cost in the millions. It is this smaller potential lost that makes it possible for indie developers to take risks, create new experiences for players and to deviate from established genres.

It is becoming increasingly important to veer from the conventional genres. AAA games continue to dig deeper into the known due to their fear of alienating the AAA player. This fear of taking risks aided to catalyst the interest of indie games, that work outside of these exhausted genres. The shrinking space of AAA games in the gaming scene opened up niches for indie developers to explore and increased even further the creative freedom of smaller studios.

The decrease in creativity can also be attributed to other factors other than the cost. Games with larger teams tend to see little influence from the individual developers, where in smaller teams every member will have a stronger influence on the end product.

Due to these factors, companies are starting to produce titles with smaller teams and trying to take more creative risks. Recently we have seen this with Ken Levine and Patrick Plourde, director of some of the biggest AAA games in the last years; with Irrational Games, the developers of Bioshock (Levine & Finley, 2007), who are beginning to work with smaller teams and focusing on smaller projects; and Ubisoft, releasing the smaller project Child of Light (Cazzaro & Débant, 2014).

As more developers start to follow this trend, it is possible that the current split between indie and AAA will begin to blur. Mid-sized developers are already disappearing, with studios like Rare acknowledging that they need to change their business plan after a series of failed releases. The splintering of mid-sized companies might become the more appealing plan, since it is nigh impossible to create AAA games on a low budget and taking an approach like Ubisoft, creating smaller teams, may be more profitable.

It is likely than indie games with higher production values will become more prominent, with the increasing competition and expectations. Indie games will grow in numbers as developers will more likely want to work on a project where their creative input is valued.

Figure 5 - Child of Light (Cazzaro & Débant, 2014)

There is also a great risk, especially if the company is started from scratch, as opposed to being crated from a larger studio, as this can be quite expensive. Having the time to work on indie projects often means working two jobs at once or going long periods with getting any pay. Although some lucky few have found successes on their first attempt, like the developers of Bastion (Rao, 2011), indie development is rarely an easy route. Even studios considered to be immediate successes, like Rovio Studios, the creators of Angry Birds (Iisalo, 2009), created fifty-two games before they found their worldwide success.

This is a trend that does not look to be temporary. Niches left open by AAA games are numerous, and the indie scene will continue to flourish as creators situate themselves in this territory. Genres will refine themselves as what we have been playing for the last decades will be questioned and new and fresh experiences like Ether One (Bottomley, 2014), The Stanley Parable (Wreden & Pugh, 2013) and Gone Home (Gaynor, 2013) appear on the gaming scene.

The demand for AAA games is not likely to dwindle, as there will always be an audience for big game series with regular releases and what could be considered as marginal invention, like the FIFA Series, but the gaming scene is currently in uncharted territory and no one knows where it will go.

4.1.3. Indie Renaissance

In the beginning, physical retail stores were where most money was. Then, the steampowered.com domain was registered, and Steam was born. Introversion Software was one of the first studios to release a game on Steam. Co-founder Mark Morris, remembers the frustrations of physical retail, noting that the deals with stores were really bad for developers: there was no guarantee that, even if the store carried your game, the game would be on shelfs; the developers only got around three euros from the twenty euros retail price; the developers had to pay for the stock, and if the game did not sell well the store could just return the stock to the developers without giving them any money.

As prospects of developing full games were becoming smaller and smaller, indie developers started creating mods for existing games, to keep their skills sharp. When Valve released Half-Life (Laidlaw, 1998), they had no idea what they had just started in the gaming scene. Their free development kit created a lively modding community that needed a place to call home. This is where Steam enters.

A year after the announcement of the store, Steam opened. Overloaded, full of bugs and painfully designed, the store served the simple needs for distribution and update for Valve games, and for the mods of said games. While far from perfect, Steam was the first attempt at a digital distribution platform that homed in on the users' needs. Players could download the games onto their computers and keep them forever.

2005 was a mark for indie developers pushed underground by industry giants. Valve made a series of agreements that gave it the freedom to sell third-party games. Ragdoll Kung Fu (Healey, 2005) and Darwinia (Delay, 2005) would become the first titles on the service that were not made in-house.

Steam was growing ever larger and on its way to take on the mainstream gaming industry. On the previous year, Bill Gates announced the Xbox Live Arcade at the Microsoft conference. When the

Xbox 360 launched in 2005, the service took off. XBLA[1] proved to be important in getting big budget names behind the indie scene, providing the financial security and status of the Microsoft brand.

Together, Steam and XBLA gave indie developers a path out of the underground into the mainstream. The opportunity to make money from a game made in a garage appeared again, and there were a lot of new developers appearing from very different backgrounds. From big companies' employees looking for creative freedom to hobbyists, everyone started creating again. To aid the industries' growth, software was created – Game Maker and Unity became staples in the game makers' toolkit. A year after opening up to third-party games, the platform had now almost one-hundred games, and the problems of visibility started to be slightly noticed.

The industry flourished, public consciousness was at an all-time high and development ecosystem started appearing. Indies were here to stay. Some genres began to gain popularity and became the epitome of indie culture to many players. Old-school pixel art aesthetics took arcade players back to the 8-bit days, and narrative exploration games like Gone Home (Gaynor, 2013) and Dear Esther (Pinchbeck, 2012) introduced games as vehicles for thoughtful messages and stories.

The indie game industry was at an all-time creative high, pushing conventions with each release. The Chinese Room, with Dear Esther (Pinchbeck, 2012) – often considered the first "walking simulator" –was one of the first players in this innovation rush. The game is a great example of the history of indie games, being created firstly as a Half-life 2 (Laidlaw, 2004) mod, and started the discussion of "Is story enough on its' own to create an engaging experience?"

The number of developers leaving big companies to create indie studios was increasing, and some of the greatest games of the industry were created from these fertile grounds. Three of the most successful titles were introduced to the mainstream public with Indie Game: The Movie (Pajot & Swirsky, 2012). The documentary was pivotal to the growing acknowledgment of the indie scene. These games were created by people who were passionate about their ideas. Phil Fish's story behind Fez (Fish, 2012) ultimately enlightened the public to the artistically nuanced, determined and grafting culture of indie development.

Indie Game: The Movie's three titles – Super Meat Boy (McMillen & Refenes, 2010), Braid (Blow, 2008) and Fez (Fish, 2012) – along with Minecraft (Persson & Bergensten, 2011), would become the Indie Golden Era, and cement many of the aspects that define the industry in the publics' eyes.

In 2008, Blow released Braid. The XBLA title introduced players to personal stories with unique aesthetics. Playing someone's idea instead of a corporation's budget became central to indie.

After Blow's release, a behemoth entered the arena. Minecraft was released in 2009, celebrating the potential of the pixel and demonstrating that you did not need realism to climb to great heights. Minecraft was a success story that inspired a lot of other developers and brough the press' attention to the scene. Markus "Notch" Persson was acclaimed as a figure of creative freedom, evidenced by his life story and Minecraft Gameplay.

In 2010, Super Meat Boy followed suit, bringing a unique brand of comedy and mechanics mixed to create what would be acknowledged by Gamespot as "incredible" and "tough as nails" by the developers. The game introduced a key idea to the scene – creating insanely difficult games followed by a smile of relief.

[1] Xbox Live Arcade

In 2012, Fez was released, becoming an important title due to the creator's struggle to release the game to the public, due to perfectionism. Fish cemented the developer as a martyr and promoted the concept of a personal experience that players were drawn to, being tired of the AAA beige.

At the same time, more distribution platforms emerged. Humble Bundle launched in 2010, bringing the "take a penny leave a penny" idea to the gaming industry, and Itch.io was created in 2013, still offering an underground indie experience, like in the old days, today.

4.1.4. Common Problems

Indie game development can have a lot of limitations and cons when compared to AAA development. From mental to physical health, passing through monetary and social problems, there are a lot of factors that can hinder an aspiring game makers journey on indie development.

One of the problems that can arise is the fact that indie developers might be alone for long periods of time. This doesn't affect small teams, as you will socialize with your teammates, but for cases where a game is made by a single person, like Undertale (Fox, 2015), one can spend long periods of time working alone, which can and will have detriment to one's health.

More often than not, indie game developers will have to wear a lot of different hats, due to the small team sizes. This means a single person might be in charge of modelling, rigging, animating, and coding, all at the same time. There is also a need for the team members, that are usually programmers, artists, or musicians, to take other roles like Marketing, QA, Playtesting, Copy Writing, and Business Management, to name a few. Frequently, team members do not have knowledge in these fields which means that these tasks will have a subpar quality. We can see this in a lot of games, where marketing, or the lack there of, is the downfall of an otherwise good product.

This also means that an animator might have to divide his efforts to do marketing tasks, instead of pouring his full effort into animation. This splitting of attention can impact the final outcome of the project since one is not fully focusing on his specialty.

Due to this need to do tasks that are not one's specialty there is sometimes a lack of motivation, with indie developers sometimes having to sleep few hours so they can have time to work on game development. This is also tied to the lack of funding, as indie developers have to find alternatives ways to fund their projects, a lot of the times having to work two jobs to be able to pay expenses.

There are also subjacent health problems tied to this living style. A lot of developers spend a lot of time at home, without going outside, working and sometimes, mostly in solo developers, not socializing for long periods of time, leading to physical and, most importantly, mental illness.

As the industry grew, and as the market became more competitive, mental illness in the indie scene became normalized. Some people treat depression and anxiety as a normal part of indie development – some thinking it is abnormal to not have these problems – passing them as part of the process.

We can look at Hotline Miami (Söderström & Wedin, 2012) as an example of the modern indie mental toll. The game hit the market in 2012 and became an immediate success. Wedin was hospitalized for two weeks because his work affected his relationship resulting in a breakup. After this Wedin became extremely depressed, being admitted into the mental ward of the hospital. During this Wedin continued to work on the game, creating levels while hospitalized and injecting elements of his depression in the game.

Finally, there is the problem that, although indie games gain by filling niche spaces left by AAA games, because these games cater to a niche audience, they can have problems reaching a diverse set of players. This is not always a problem, as a game should have a defined audience and, most of the times, does not have the need to reach every player, but developers can, sometimes, be limited to a very small niche audience that might not be numerous enough to generate revenue at a level that will let developers live comfortably, or in some cases, might not even cover the cost of the development.

4.1.5. The Indie Apocalypse

Indie developer Jordi de Paco, from the studio Deconstructeam, sees the current years as the most challenging for small teams. De Paco says that anyone can make games, but it is really hard to do something that no one has ever seen, when in comparison to the beginning of the indie renaissance. Today, for something to stand out, there has to be innovation and quality.

De Paco states that his studio managed to make a profit just from pre-orders, back in 2014, but in their latest game sales only slipped into profit months after the release. And this is the exception – the game that was able to make it work.

Currently we see a daily avalanche of game releases into online marketplaces like Steam. This is seen by a lot of developers as a threat to their livelihood. Where we could see 20 releases a week on Steam, we now see 20 per day. This means that a game's release is very likely to be eclipsed by the heavy stream of releases. According to de Paco, hundreds of indie studios have sprouted in his home country, but only 10 of them are able to actively turn a profit.

Over the past decade, the number of games on the market has increased exponentially. Steam has transformed into a behemoth in the PC gaming world. Console storefronts, like Nintendo's eShop, have also gained a lot of traction. Indie developers competing for attention against blockbusters like Assassin's Creed (Désilets, Raymond, & May, 2007) and Call of Duty (Rieke & Fukuda, 2003) might think that the odds were against them from the start, but the growth of indie games have tightened the vice against them even more, making it hard to be notices in an over-crowded market.

The weekly trickle of games flowing into Steam has transformed into a busy river, with the number of games on the platform nearly doubling every year from 2014 — 1,772 that year, 2,964 in 2015, 4,207 in 2016, and 7,672 in 2017, as stated by Sergey Galyonkin of SteamSpy, a platform estimates the sales of games on the platform. In 2018, up until September, there were around 6,000 games released on Steam; the majority of them have sold fewer than 500 copies, at the time.

The Nintendo Switch's eShop also saw a surge of new titles, seeing around five or six games a week initially; this number as multiplied by a factor of four or even five, according to Eurogamer, during the year of 2017.

In the mobile market, the situation is even worse. After years of extraordinary growth, Apple's App Store market became so overcrowded that analysts reported almost a 30 percent decrease in the volume of new submissions in 2017. The Google Play Store experienced no such drop, growing at a rate of 17 percent year-on-year — a possible by-product of Apple's more aggressive policies toward app makers. Even with these reduced numbers, however, both stores receive hundreds of new arrivals per day, if not thousands.

As the daily torrent of games continues to pour into these storefronts — and many others — developers of all sizes are forced to consider the incapacitating effect that overcrowding is having on every level of development – from solo developers building pixel art dreams in their homes to AAA giants cranking away at massive worlds. But while passionate hobbyists might simply return to their day jobs if their indie ambitions stay unfulfilled, the scale of big-budget games spirals ever upward, meaning that a lot of employees may lose their jobs if the studio's big project does not work out.

Due to these changes, some studios have created new strategies to stand out from the rest, such as having their own unique and weird marketing campaigns. Some still hold on to the traditional mantra: Stay within your comfort zone, keep up a steady stream of sequels, and stick as close as you can to an established publisher. But others, stung by the fear of financial failure, have taken more drastic measures.

As competitors start to appear in every genre, some studios that usually explore more niche genres have found themselves moving toward more popular types of game, eager be a part of the next big trend. But, while it is certainly easier to follow trends, these changes come with problems for every type of developer.

When Finnish studio Housemarque released a twin-stick shooter into Steam in 2017, they did not expect the game to be a huge success. Even with these lowered expectations, however, the team found itself shocked at the lack of impact that the Housemarque name seemed to have on the consumers scrolling through Steam every day. When the sales numbers finally trickled in, the mood was sombre, with the studio's head of publishing, Mikael Haveri, describing it as "devastating." Mikael Haveri, the studio's Marketing Director, says that there are simply too many small developers making quality games in that style and genre for Housemarque's neoclassical approach to support a studio of its scale.

Months after the news of the game's subpar commercial performance, CEO Ilari Kuittinen announced a surprise declaration on the studio's website with a headline that shouted the now-infamous expression "ARCADE IS DEAD," stating that the team's arcade niche had failed to earn the sales that the studio required to survive. Haveri says that he views the company's change of focus as the first step toward a new image, reinforced by tactics like securing the attention of streamers and other influencers before launch, as well as communicating a single unique mechanic that defines a game instead of a list of unclear ideas.

Since Housemarque's departure from the old-school genres that constitute the "arcade" label, the team has shifted its focus to a new vein of action game: Stormdivers (Tikkanen, 2019), a fast-paced multiplayer third-person shooter that uses environmental effects to revamp the ever-growing battle royale genre. While Haveri is excited by this market-driven concept, he acknowledges that trying to break into the saturated competitive shooter field, where behemoths like Fortnite (Sugg, 2017) and Overwatch (Craig, Mercer, & Elliott, 2016) overshadow the spotlight, continues a dangerous scheme.

This plan of scouting a potential market first and letting the ideas follow is echoed by Dodge Roll, the developers behind Enter the Gungeon (Crooks, 2016), a light-hearted but difficult roguelite that takes cues from Edmund McMillen's The Binding of Isaac. While game director Dave Crooks is grateful for the millions of sales, he says that Enter the Gungeon is not necessarily in the genre that the team initially planned to make. When they got laid off from EA Mythic because of the studio closing, they discussed making a more traditional platformer like McMillen's other great hit, Super Meat Boy.

Man, how do you even make a platformer after Super Meat Boy? They have 350 levels. How did you even compete with that? – Dave Crooks

This is not to say Enter the Gungeon gathered the sort of pre-release excitement that the game's publisher, Devolver Digital, expected. For Crook, the only reason that the game managed to rocket its way to the top was the understanding that streamers were poised to embrace the game with unrivalled vigour, along with legions of roguelike fans salivating for a challenge of dexterity and wit.

Each of the aforementioned studios has a unique experience with the struggles that the pure inundation of games has brought onto the industry. But in the end, they all come to an agreement on one thing: Nobody knows what will happen.

4.2. Game & Player Taxonomy

4.2.1. Game

Games are a fundamental part of human existence (Crawford, Peabody, Art, Web, & Loper, 2003). According to Crawford, we have started to use game words and terms on aspects of our lives that are not games. In example we play along when we are in an uncomfortable situation, we play games when we are not sincere, and the use of many other expressions that are normal in our society show that gaming terms and expressions have permeated into our existence.

In order to define what is and what is not a game Crawford proposes a set of fundamental attributes that all games should have: **representation**, **interaction**, **conflict,** and **safety**.

Crawford states that a game is **a closed formal system that represents a part of reality, subjectively**. A game is a complete and self-sufficient structure, thus there is no need to refer to external agents. This means that a game should have explicit **rules** that cover all possibilities found in the game. Games are also subjective representations of reality. When we play a game, we know that our actions do not directly mirror reality, although there may be a metaphor that translates into the real world. Objective representation is only needed to a level that supports the player's fantasy.

Games are also **interactive** in their nature, unlike movies or paintings. Interaction is important in a representation of reality, because, in the real world, we are agents of change. When we act on the world in some way there is an expectation of a reaction. Puzzles are not games, in the sense that when we interact with some puzzle like a Rubik's Cube, we do not receive a response to our actions.

Interaction adds a social or interpersonal layer into the experience. It also transforms the nature of the challenge from a passive one to an active one - a game reacts to and acknowledges the player, being solved by the player's own solution and not by discovering the designer's solution.

Games also have a **conflict**. From the interaction between the player and the game a conflict naturally rises. The player has an objective and there are obstacles that prevent them from easily attaining this goal. If the obstacles are active or dynamic and react to the player in a purposeful way the challenge becomes a game. Conflict can be direct or indirect, violent, or nonviolent, but it is fundamental to all games.

Finally, games are a **safe way to experience reality**. When we think of conflict we think of danger. Games are a way to psychologically experience conflict and danger, without the risk of harm.

Other authors such as Caillois propose different key characteristics that define what should be called a game or not. A game should have the following characteristics (Berger, Caillois, & Barash, 1963) :

1. **Free**: It is not obligatory to play a game; once the activity becomes obligatory it loses all its attractiveness; there must be an intrinsic motivation to play the game.
2. **Separate**: A game has limits in time and space, that have been identified and set-in advance.
3. **Uncertain**: The course of the game cannot be determined, not its outcomes be achieved beforehand; there must be some leeway for innovations left for the player's initiative.

4. **Unproductive**: A game does not create any goods, wealth, nor new elements of any kind; the ending situation is identical to the beginning of the game.
5. **Governed** by rules: a game has its own rules, and is only governed by its legislation, suspending ordinary laws.
6. **Make-believe**: a game is a second reality, separate from real life.

These characteristics are formal in their nature, not prejudging the content of the games.

Games are “**a form of art in which participants, termed players, make decisions in order to manage resources through game tokens in the pursuit of a goal**” (Costikyan, 2006).

According to Costikyan games must have **decision making**, and not only interaction. A light switch is interactive: you flip up the switch and the lights turn on and vice-versa. It is interactive, but it is not a game. Interaction must have a purpose, so the player must make decisions.

A game must also have a **goal**. If there is no goal your choices become meaningless and there is nothing to strive toward; there is nothing that keeps you playing.

Games have an “**opposition**”, something that opposes the player, an obstacle, conflict. When a player is handed the solution and there is no struggle to get to the goal there is no interest to keep playing; the victory is bitter-sweet. More than having a goal one must struggle to get to that goal, giving you the thrill of victory.

Players must also have to **manage resources**, be it in-game currency, time or simply points. The player’s decisions must have a weight to them. If a game has more than one resource the decisions become more complex and the players must think about the decision that will get them the most desired outcome.

The resources have **tokens** that represent them. In board games they are the pieces, in card games they are the cards. Each game has its own unique tokens that give the players the means of managing resources. The less tokens there are, the more detailed they must be; when you have only one or two tokens there are very-well defined rules of what you can do with that token.

A game must also give **information** about itself. What good is a game that has a complex weather system when the player has no way to know when it is raining or what effects does the rain have on the game. Give the players the information needed to make decisions, directly or indirectly. Give ways to find the information without overfeeding the player. For a player to achieve a goal there are several things they must know and there must be a way to find that information.

There are also other characteristics that strengthen games such as **diplomacy**, **variety of encounters**, **roleplaying**, **socializing**, **narrative tension**, and others, although these are not fundamental for defining a game.

Based on other authors, such as the aforementioned ones, Juul finds 6 characteristics that define what should be considered a game (Juul, 2002):

1. **Rules**: Games have rules.
2. **Variable**, quantifiable outcome: Games have variable, quantifiable outcomes.
3. **Value assigned to possible outcomes**: The different potential outcomes of the game have different values, some positive, some negative.
4. **Player effort**: The player invests time and effort to get to a certain outcome.
5. **Player attached to outcome**: Players are attached to the outcomes of the game.
6. **Negotiable consequences**: The same set of rules can be played with or without real-life consequences.

Katie Salen and Eric Zimmerman, based on other authors and trimming some of the characteristics that they deemed unnecessary, defined a game as "**A game is a system in which players engage in an artificial conflict, defined by rules, that results in a quantifiable outcome**" (Salen & Zimmerman, 2003). We can deconstruct this statement into the following parts, further discussing this definition:

1. **System**: A game is a system.
2. **Players**: A game must have participants that actively play.
3. **Artificial**: Games have a boundary that separate them from real life in both time and space.
4. **Conflict**: There must be a contest of powers. From cooperation to competition, from solo to multiplayer conflict, this contest can take many forms. Conflict is central to games.
5. **Rules**: Rules make the structure of the game, saying what a player can and cannot do.
6. **Quantifiable outcome**: Games have a goal or an outcome. At the end of the game the player must have won or lost some king of score. This separates games from other play activities.

With this definition we can create direct parallelisms with the Engagement Design Model, described in a later chapter, since just like with the ED Model we define Players, Rules and Quantifiable Outcomes.

For the rest of this document the definition of "game" that will be used is the one by **Salen and Zimmerman**, as it is concise and is the most useful definition for this research's goals.

4.2.2. Game Genres

When talking about games one thing that is very important to define is game genres, especially when talking about designing for different player types, as they can help us find the defining features certain games should have and help designers to understand what players are looking for in a particular genre. The following table tries to list and define the principal game genres, when defining genres by gameplay:

Table 1 –Game genres by gameplay, retrieved from (Lee, Karlova, Clarke, Thornton, & Perti, 2014)

Genre	**Description**
Action	Games with a heavy emphasis on a series of actions performed by the player in order to meet a certain set of objectives (e.g., Super Mario Bros. (Miyamoto & Tezuka, 1985))
Action/Adventure	Games which are set in a world for the player to explore and complete a certain set of objectives through a series of actions (e.g., The Legend of Zelda: Breath of the Wild (Hidemaro & Aonuma, 2017))
Driving/Racing	Games involving driving various types of vehicles as the main action, sometimes with an objective of winning a race against an opponent (e.g., Ridge Racer (Tanaka, 1994))
Fighting	Games involving the player to control a game character to engage in a combat against an opponent (e.g., Mortal Kombat (Boon & Tobias, 1992))
Puzzle	Games with an objective of figuring out the solution by solving enigmas, navigating, and manipulating and reconfiguring objects (modified from Wolf, 2002) (e.g., Tetris (Pajitnov, 1984))

Table 1 - Game genres by gameplay, continued

Genre	Description
RPG	Games with an emphasis on the player's character development and narrative components (e.g., Pokémon Crystal (Tajiri, Masuda, & Sugimori, 2000))
Shooter	Games involving shooting at, and often destroying, a series of opponents or objects (Wolf, 2002) (e.g., Borderlands (Armstrong, 2009))
Simulation	Games intending to recreate an experience of a real world activity in the game world (e.g., Trauma Center: Under the Knife (Kitano, 2005))
Sports	Games featuring a simulation of particular sports in the game world (e.g., Fifa 20 (EA Romania, 2019))
Strategy	Games characterized by players' strategic decisions and interventions to bring the desired outcome (modified from Apperley, 2006) (e.g., Shogun: Total War (Simpson, 2000))

Although the most common way to describe game genres is by **gameplay** there are other factors that can determine a game genre like **artistic style**, **presentation,** or **target audience,** among others, although these factors are not very important for defining game genres in this investigation. **Purpose**, on the other hand, will be important because, like gameplay, game genres defined by purpose can influence gameplay and rule choices that, in turn, affect the types of players that will be engaged in the game. Just like the table above, the following table seeks to define game genres by purpose and to give a small description of these genres:

Table 2 - Game genres by purpose, retrieved from (Lee et al., 2014)

Genre	Description
Education	Games in which the goal is to support learning. There are a broad range of educational games, from those teaching spelling to computer programming to animal facts, etc. (e.g., Brain Age: Concentration Training (Matsushita & Kanno, 2012))
Entertainment	Games in which the goal is to allow the player to have fun. A large majority of games have entertainment as their purpose. (e.g., Super Mario Bros. (Miyamoto & Tezuka, 1985))
Exercise	Games in which the goal is to get players to move their physical bodies and burn calories or participate in some type of athletic pursuit. (e.g., Dance Dance Revolution (Katsunori, 1998))
Meditation	Games which help support players' engagement in meditation and mindfulness activities. (e.g., Wii Fit (Matsunaga, 2007))

Table 2 - Game genres by purpose, continued

Genre	Description
Party	Games designed to be played in the setting of a social gathering. These games are designed for relatively short-duration play, allow for multiple players and quick turn-taking, and may also be designed to be spectator-friendly for the enjoyment of those who are not currently playing. (e.g., Move or Die (Berbece, 2016))
Social	Games designed to involve around heavy social interaction rather than playing in solitude. The players engage in group activities such as making friends, chatting, sending daily gifts, teaming up for tasks, etc. (e.g., World of Warcraft (Pardo, Kaplan, & Chilton, 2004))
Serious Games	Games that have an explicit and carefully thought-out educational purpose and are not intended to be played primarily for amusement. (Abt, 1970)
Games with a purpose	Games that use humans as a way to solve simple problems that computers have a hard time to solve. (Von Ahn & Dabbish, 2008)

Having defined game genres by gameplay and purpose we can now move to defining player taxonomy and discussing different player taxonomy models. With this information we can now understand the different player types preferences in game genres and talk about the preferred games by each player profile.

4.2.3. Player Taxonomy

When talking about game design we must understand that we are designing for people. Players can be grouped by different characteristics like gender, age, and other demographic traits, but these are all external factors and, when we talk about player taxonomy, we usually refer to internal factors – this is called psychographics – where they find pleasure. We can find numerous models that group and describe players according to the experiences they seek while playing a videogame. In the following sub-chapters, we will discuss player taxonomy from a demographic viewpoint and then we will talk about some of the more well-known psychographic player taxonomy models.

"A good game designer should always be thinking about the player, and should be an advocate for the player" (Schell, 2008).

Demographics

In game design the most important demographic variables are age and gender (Schell, 2008). As we grow older our preferences change and, independently of age, males and females tend to have different tastes. Below we will have a brief analysis of the age groups game designers have to consider.

a) **Infant/Toddler (0-3)** – At this age, their interest is mostly centred around toys. Games tend to be too complex for this age group.

b) **Pre-schooler (4-6)** – Usually when the first interest in games appears. The games are very simple and played with parents, that can bend the rules to keep the games fun.
c) **Kids (7-9)** – The "age of reason." At this age children have entered school and are able to solve problems. It is at this age that children start making decisions about what they like, no longer following their parent's choices.
d) **Preteen (10-13)** – At this age children have a tremendous growth and start to think more deeply. Children this age get passionate about their interests. In boys, these interests tend to be games.
e) **Teen (13-18)** – Teens are interested in trying new experiences. We often see a stringer divide between boy's and girl's interests. Males normally continue to be interested, and usually get more interested, in competition and mastery, while females are more focused on real-world problems and communication.
f) **Young Adult (18-24)** – Adults tend to play less than children. They do not stop playing, but at this point in life they have more defined interests and other responsibilities. Young adults, normally, have both time and money, which makes them big consumers of games.
g) **Twenties & Thirties (25-35)** – At this point in life it is expected for adults to form families. These responsibilities add up and, because of this, time becomes scarce. Most adults this age tend to be casual gamers, playing only occasionally or with children. There are also some hardcore gamers that are a very important market because they normally buy a lot of games and are vocal about likes and dislikes, influencing the buying choices of their social network.
h) **Thirties & Forties (35-50)** – Usually, players this age tend to look for games that the whole family can enjoy. As the children in the family get older this age bracket must make decisions about game purchases. They tend to be casual gamers, because they are very caught up in their family and career.
i) **Fifties & up (50+)** – At this point in life these players children have moved out and they tend to have a lot of time in their hands. Some return to old games that they used to play; others look for new experiences. Adults this age are very interested in social games. Complex fine motor control can be frustrating to these players because their hands and eyes are not the same as they were young.

More important than understanding every age group is knowing that all play activities are connected to the childhood. When designing a game for an age group, designers must keep in mind what was popular when this age group was young.

> *"To truly communicate with someone you must speak the language of their childhood"* (Schell, 2008).

Gender is also important when designing games. Males and females are different. They have different tastes, preferences, experiences, and skills. Where these differences stem is not relevant to game designers, acknowledging the differences is.

According to (Koster, 2004), the core of playing and winning games is mastering abstract formal systems, which is usually more enjoyed by males than females. Yet, we still see a lot of interest in games in the female audience. This means that the game's core can have a variety of experiences that appeal to both genders, such as socialization, story, creativity, and learning. There is no definitive list of differences between genders and their interests, but there are some generalizations that we can make. For males we usually see more simple and primal goals such as:

1. **Mastery** – Males like mastering things. The importance or usefulness of that thing is secondary to how hard it is. The more challenging the better.
2. **Competition** – Males enjoy competing against others, to prove their superiority.

3. **Destruction** – Males like destruction. Just like a child destroys their block construction after making it, this feeling of chaos and excitement moves males. Videogames can be a great outlet for this destruction, as there are no real-world repercussions.
4. **Spatial puzzles** – It has been shown by studies that males tend to have better spatial reasoning skills than females. Consequently, puzzles in a 3D space are usually more interesting to males.
5. **Trial and error** – Males often like to learn things through trial and error, tying to their liking of mastering things.

In females we tend to see a preference for games with social behaviour and emotional discoveries. There is a liking for experiences that can be applied to their own lives:

a) **Emotion** – Females like to explore the human emotion, and more than a component of the experience, the emotion is the experience itself.
b) **Real world** – Females usually like entertainment that has a relation to the real world. While growing up we will see girls playing house, pretending to be doctors and other activities from the real world, while boys tend to like more fantastic settings. This is seen in adulthood as well as females usually prefer social games.
c) **Nurturing** – Female players tend to like to nurture. We often see females giving up a victory to help other, partly because the relationship and the feelings of others are more important than winning the game.
d) **Dialog and verbal puzzles** – What males tend to have more in spatial skills females more than make up to in increased verbal skills.
e) **Learning by example** – Females usually prefer to learn by example. They like clear tutorials that guide you step-by-step, so that the player knows what to do when faced with a task.

There are a lot more differences, such as the female capacity for multi-tasking while males tend to focus on one task at a time. Whether you consider these variables or not, the important thing is to make the game the most fun for the player.

Besides demographics we have psychographics, the study of what players find pleasurable. These are internal factors, while demographics are external factors, for preferences in experiences. Psychographics can be about life choices, such as "cat person", or "football fan", but most psychographic kinds, when talking about players, are more tied what a person enjoys. Below we will discuss some of the most important models for player taxonomy and then we will discuss a new model that will serve as the core for this investigation.

Bartle's Model

In his article "Hearts, clubs, diamonds, spades: Players who suit MUDs" Richard Bartle identifies four different approaches to playing MUDs [2] based on the inter-relationship between two dimensions of playing style: action versus interaction, and world-oriented versus player-oriented (Bartle, 1996). In his player taxonomy model Bartle describes the following player types, and pairs them with a suit from a deck of cards, in order to summarize them:

[2] MUDs stand for Multi-User Dungeons and are a predecessor to the modern Massively multiplayer online role-playing game (MMORPGs) (Bartle, 2004), being online multiplayer worlds, usually in text.

Table 3 - Bartle player types

Player type	**Description**
♦ Achievers	Achievers view point-gathering and rising in levels as their main goal, and all is ultimately subservient to this. They only explore to find new sources of treasure, or improved ways of getting points from it. Socialising is a relaxing way of discovering what other players know about accumulating points. They only kill to eliminate rivals, or to gain a large number of points (if points are awarded for killing other players). Achievers are Diamonds because they seek treasure.
♠ Explorers	Explorers love having the game show its internal machinations to them. They try increasingly different actions in wild, out-of-the-way places, looking for interesting features (i.e., bugs) and figuring out how things work. Scoring points may be necessary to get to a new phase of exploration, but it is tedious. Killing is quick and might be a constructive exercise, but it causes too much trouble in the long run if the deceased return to seek revenge. Socialising can be a source of new ideas to try out, but most of what people say is trivial or old news. Fun comes from discovery. Explorers are Spades because they dig around for information.
♥ Socialisers	Socialisers are interested in people, and what they have to say. The game is merely a stage where things happen to people. Inter-player relationships are important: empathising with people, sympathising, joking, entertaining, listening. Some exploration may be useful to understand what everyone else is talking about, and points-scoring could be required to gain access to communicative spells s (as well as to obtain a certain status in the community). Killing is something only ever to be forgiven if it is a fruitless, impulsive act of vengeance, perpetrated upon someone who has caused pain to a friend. The ultimate objective is not to gain levels or to kill; it is getting to know people, understanding them, and forming beautiful, long-term relationships. Socialisers are Hearts because they empathise with other players.
♣ Killers	Killers have fun imposing themselves on other players. Commonly, people attack other players with the intent of killing off their characters. The greater the pain caused, the greater the killer's joy for causing it. Point-scoring is usually necessary to become powerful to begin causing havoc, and exploration is needed to find new and effective ways to kill people. Even socialising is occasionally valuable beyond taunting a recent victim, for instance finding out someone’s playing habits, or debating tactics with fellow killers. There are all just ways to get to a goal, though; only the knowledge that a person, somewhere, is distraught by what you have done, yet can do nothing about it, is the true adrenalin-shooting, juicy fun. Killers are Clubs because they hit people with them.

As seen on the table above these player types, naturally, cross over, and players will change their player type depending on their current mood. However, most players have a primary player type that they usually stick to, only changing their play style to advance in their main goal (Bartle, 1996).

These player types can be displayed in an interest graph like the one below, that represents the players interests on two axes: the x-axis represents player versus world-oriented action and the y-axis represents action versus interaction.

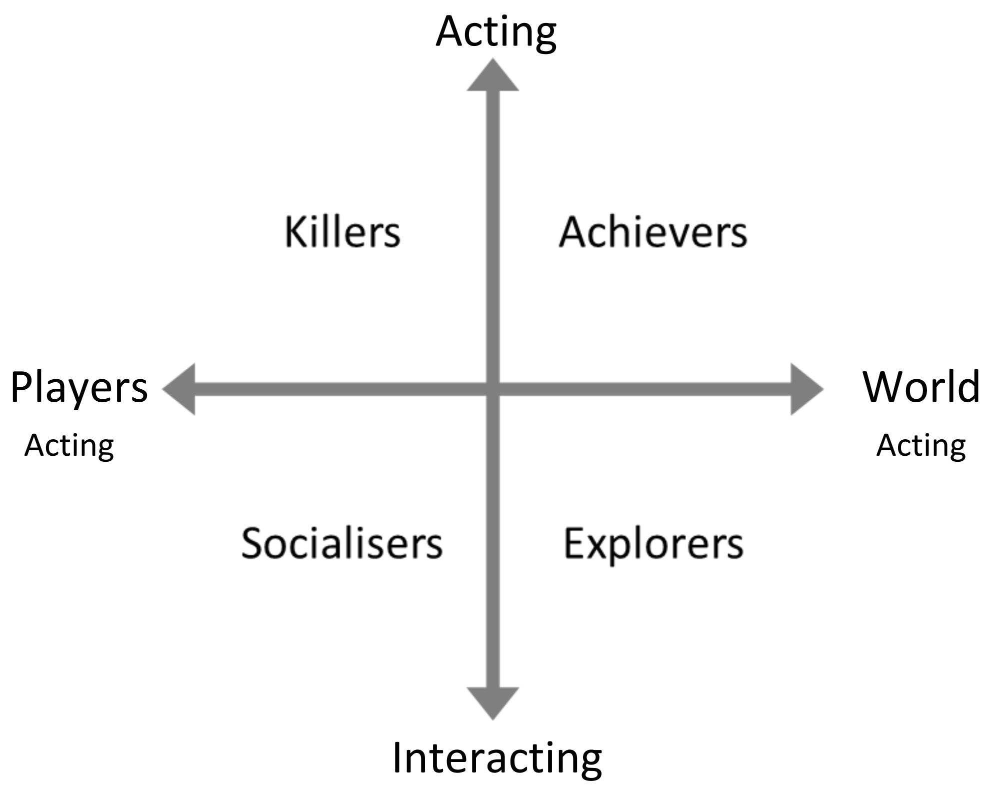


Figure 6 - Bartle player Types

Seeing this graph, we can consider the following details for each of the play styles:

a) Achievers are interested in **ACTING** on the **WORLD**– doing things to the game. Their main motivation is to master the game and to attain an in-game status. Achievers like their status and how quickly they achieved it.
b) Explorers love when the game surprises them – they **INTERACT** with the **WORLD**. These players are filled with a sense of wonder and are always out to find new information. They are proud of their knowledge of the game, especially if new players search them for that knowledge.
c) Socialisers **INTERACT** with other **PLAYERS** – they have interest on doing things with people. They are interested on the characters, as the game world is only a stage for the relationships. These players are proud of their relationships, their influence, and their links.
d) Killers are interested in doing things to people – they like **ACTING** on **PLAYERS**. These players have fun demonstrating superiority over other players and are only interested in affecting real people, not computers. They are proud of their reputation and of their refined fighting skills.

Bateman's Models

A few years after Bartle describes his player taxonomy model, Christopher Bateman presents the Demographic Game Design (**DGD1**) (Bateman & Boon, 2005), based on gameplay preferences and the Myers-Briggs Type Indicator personality test (Myers & Myers, 1980). In this model Bateman describes four different profiles:

Table 4 - DGD1 Player Types, retrieved from (Bateman & Boon, 2006)

Player Type	**Description**
Conqueror	This player type is motivated by personal triumph over adversity. Conqueror players know that patience will pay off and tend to finish the games they start.
Manager	This player type is associated with mastery and systems. For managers, the victory is in mastering the game, and not reaching a specific goal in and of itself. Manager players may not finish games they start.
Wanderer	This player type is interested in experiencing new things. They get their emotion from the feeling that something new might be coming, from a good story or from the beautiful world they see themselves in.
Participant	Players of this type are moved by emotions and involvement. They get their fun from interacting with people, but also from enjoy involving characters.

Later Bateman created a survey for a second Demographic Game Design model, known as **DGD2**, that has a greater focus on different types of fun (Bateman, n.d.-c). This new model was then used to create the **BrainHex** model (Bateman, n.d.-b), based on neurobiological research papers. In this new model there are seven player types, distributed around a hexagon with one in the centre, each paired with a different part of the nervous system:

Table 5 - BrainHex Player Types, retrieved from (Bateman, n.d.-a)

Player Type (Play Style)	**Description**
Seeker (Seek)	This player type is moved by finding new and curious things. This behaviour stimulates the part of the brain that processes sensory information as well as the memory association centres.
Survivor (Escape)	This player type likes the sensation of fear and the excitement they get from escaping a scary threat. This sensation is produced by the amygdala.
Daredevil (Rush)	This player type likes living on the edge, risk taking and the thrill of the chase. They are moved by adrenaline.
Mastermind (Solve)	Mastermind players find their fun in solving a difficult puzzle and in devising strategies to overcome problems. These actions are tied to the decision centre of the brain.
Conqueror (Defeat)	These players like winning against adversity. They like winning against difficult odds and fighting tooth and nail for their victory. They are moved by adrenaline and norepinephrine, the latter one gets us angry and motivated.

Table 5 - BrainHex Player Types, continued

Player Type (Play Style)	Description
Socialiser (Relate)	Socialisers like relating to people. They like talking, helping and being around other players. They are moved by oxytocin, a chemical produced by the social centre of the brain, that gives us the sense of unity.
Achiever (Collect)	Achievers are the most goal-oriented class, and these players enjoy collecting and completing all things in a game. These players tend to like grinding and are moved by dopamine, a chemical released from the pleasure centre of the brain.

All these classes can, naturally, cross to form subclasses, giving us a total of 42 subclass profiles. For example, a player can be an Achiever-Seeker, which signifies that their primary motivation is completing the game, but they also get much pleasure just from the discovery of new things. Also, as we can see on the table above, some of the classes cross when talking about the chemicals and parts of the brain that they are related to, such as the Conqueror and the Daredevil sharing the release of adrenaline. This model serves as a successor to the DGD1 and is the most recent model from Christopher Bateman.

Gamer Motivation Model

More recently we have seen a large empirical study, that has given us the Gamer Motivation Model (Yee, n.d.-a). This study started with 3000 samples in 2007, then the model was reiterated using 140 000 samples (2015) collected from an online tool and is now still being adjusted and counts with more than 400 000 players (2019). These samples have given way to six different player motivation clusters, each divided into two motivations (Yee, n.d.-b):

Table 6 - Gamer Motivation Model, retrieved from (Yee, 2015)

Motivation Cluster	Motivation	Description
Action	Destruction	Gamers who score high on this component are **agents of chaos and destruction**. They love having many tools at their disposal to blow things up and cause relentless mayhem. They enjoy games with lots of guns and explosives.
	Excitement	Gamers who score high on this component enjoy games that are **fast-paced, intense, and provide a constant adrenaline rush**. They want to be surprised. They want gameplay that is full of action and thrills and rewards them for rapid reaction times.

Table 6 - Gamer Motivation Model, continued

Motivation Cluster	**Motivation**	**Description**
Social	Competition	Gamers who score high on this component enjoy competing with other players, often in **duels, matches, or team-vs-team scenarios**.
	Community	Gamers who score high on Community enjoy socializing and collaborating with other people while gaming. They like **chatting and grouping up with other players**.
Mastery	Challenge	Gamers who score high on Challenge enjoy playing **games that rely heavily on skill and ability**. They are persistent and take the time to practice and hone their gameplay so they can take on the most difficult missions and bosses that the game can offer.
	Strategy	Gamers who score high on this component enjoy games that require **careful decision-making and planning**. They like to think through their options and likely outcomes. These may be decisions related to balancing resources and competing goals, managing foreign diplomacy, or finding optimal long-term strategies.
Achievement	Completion	Gamers with high Completion scores want to **finish everything the game has to offer**. They try to complete every mission, find every collectible, and discover every hidden location.
	Power	Gamers who score high on this component strive for **power in the context of the game world**. They want to become as powerful as possible, seeking out the tools and equipment needed to make this happen.

Table 6 - Gamer Motivation Model, continued

Motivation Cluster	Motivation	Description
Immersion	Fantasy	Gamers who score high on Fantasy want their gaming experiences to allow them to **become someone else, somewhere else**. They enjoy the sense of being immersed in an alter ego in a believable alternate world and enjoy exploring a game world just for the sake of exploring it.
	Story	Gamers who score high on Story want games with **elaborate storylines and a cast of multidimensional characters** with interesting back-stories and personalities.
Creativity	Discovery	Gamers who score high on Discovery are **constantly asking "What if?"** For them, game worlds are fascinating contraptions to open and tinker with.
	Design	Gamers who score high on this component want to **actively express their individuality** in the game worlds they find themselves in.

When comparing this model to the others presented, we start to see a shift in the paradigm from models based on the players experience to the players motivations and engagement. It's based on this shift that we see the Engagement Design Model (ED Model) (Zagalo, 2020), that we will discuss on a following chapter, appear.

4.3. User Experience Design & Engagement Design

User Experience (UX) design is the process used by design teams while creating products that deliver meaningful and applicable experiences to users (Norman, 2016). This encompasses the design of the whole process of getting and integrating the product, including aspects of branding, design, usability, and function.

Because UX design covers the entire user journey, it is a multidisciplinary field. UX designers come from different backgrounds such as visual design, programming, psychology, and interaction design. To design for human users also means that it is necessary to work with a heightened scope concerning accessibility and adapting to many users' physical limitations, such as poor vision. A UX designer's tasks vary, but usually consist of user research, making personas, designing wireframes and interactive prototypes, and testing designs. These tasks may vary, but they always require the designers to be the users' advocate and to keep the users' needs at the centre. This is also why a lot of UX designers work in a user-centred work process and keep channelling their efforts until they address all of the relevant issues and user necessities optimally.

4.3.1. Interaction Design

With the growth of the internet, the normalization of home and leisure computing, and the appearance of the digital interactive consumer, the cultures of design and engineering started to gravitate towards a common interest in unrestricted use and user experience. Close to the turn of the century, the notion of Interaction Design started to gain popularity as a way to hail a more designer way to approach the topic, going beyond utility and efficiency and considering also aesthetic qualities of use.

In The Encyclopaedia of Human-Computer Interaction (Christensen, 2012), Jonas Lowgren offers a simple formulation of Interaction design, capturing the heritage of the term and drawing some lines to demark the field:

"Interaction design is about shaping digital things for people's use"

The notion of shaping is used to suggest a designerly activity and implies five major characteristics.

a) Design changes situations by shaping and implementing artefacts
b) Design is about transformation and the means to start change in a particular situation is the designed artefact.
c) Design is about exploring possible futures
d) Design, as opposed to analytical and critical studies, is focused on what could be. Outlining design as exploration means that it makes sense to spend time in early phases of work, looking for possibilities before choosing a particular direction. This also means that interaction design often involves the participation of future users.
e) Design frames the "problem" in parallel with creating possible "solutions"

From the previous notions we can conclude that when something is designed the situation where it is used is no longer the same. This means that analysing something just to define a "problem" is limited. Exploring possible futures implies not only different "solutions" but also different "problems." As a consequence of this traditional system development and engineering processes are not considered design processes.

Design involves thinking through sketching and other representations

When sketching, the designer is not only copying images from his inner eye. Drawings are experiments that respond with insights into weaknesses, strengths, and changes in a loop of thinking. This also applies to other sketching media used; the notion of sketching is more connected to the mindset than to the medium used. If an external representation engages the designer in a conversation about details and implications of an idea, and if it is disposable, quick and tentable, it is a sketch. It could be a napkin or a computer drawing – what matters is the purpose and intention.

Design addresses instrumental, technical, aesthetical, and ethical aspects throughout

Each possible future introduces considerations and trade-offs in all these dimensions, and there is no obvious way to sequence them. This is also true for interaction design. Technical choices influence the aesthetics of the resulting interaction, instrumental decisions on features have ethical consequences, and so forth. For an interaction designer, users are whole people with complex responsiveness and design processes need to be handled appropriately.

Interaction design shapes digital things. This means Interaction designers work in digital materials, such as software and communication networks. These digital materials pose specific requirements on sketching practices, for example. When designing a new interaction technique, where previous experience is not abundant, it might be necessary to experiment with constructions in software. These constructions are to be made with a sketching mindset, which means they are to be quickly made, disposable, have many variations on the same theme and focused on behaviours and effects.

This digital material shaping is done for the people’s use. The historical notion of people’s use is tied to workplace settings and instrumental reasons – using a software to get the job done, as quickly and efficiently as possible. The growth of technology outside the workplace meant that there were, now, different uses for tech, such as pleasure and entertainment. This broadened the understanding of use, and had a major impact on interaction design, with the rise of the notion of user experience to encompass all types of non-instrumental, aesthetical, and emotional qualities in the human use of a digital thing.

In conclusion, Interaction Design can be defined as shaping digital things for people’s use. Interaction designing is knowledge-intensive and multidisciplinary in nature.

4.3.2. User-centric design

User-centred design, or UCD, is defined as an iterative design process in which designers focus on users and their needs in each of the phases of the design process. In UCD, designers involve the users throughout the process, using a variety of research and design techniques, to create usable and accessible products for them. Designers mix investigative methods and tools, and generative ones to develop an understanding of the users’ necessities.

Usually, each iteration of the UCD approach involves four different phases. Firstly, designers try to understand the context in which users may use a system. Then, they identify and specify the users’ needs. After this, a design phase follows, in which the designer team develops solutions. Finally, there is an evaluation phase where the designers assess the outcomes of the tests against the users’ context and requirements. This is done to see if the solution matches the context and satisfies all the users’ needs. Following, the team will further iterate these four phases, until the results are the ones expected.

In User-centred design, projects are based around an understanding of the users, environments, and tasks. The goal of the process is to find and address the whole user experience. Thus, design teams should be multidisciplinary – including ethnographers, psychologists, and engineers, for example, the teams should have domain experts, stakeholders, and the users themselves. Experts can conduce evaluations of the designs, using guidelines and criteria, however, there must be kept in mind two crucial points. Firstly, to cover the entire user experience, users must be involved in the evaluation; and secondly, there must be long-term monitoring of use.

When users are brought into every phase of the design process, effort and other resources are invested into a way to find what works well, what does not and the reasonings behind this. Users are a warning system that can be used to course-correct and fine-tune designs. They can uncover a lot of aspects, be they positive or negative, that the design team may overlook in areas such as usability and accessibility. This is the reason behind the importance of understanding the benefits a user-centred design can bring to a project.

Although being user-centred is more costly to a project there are some ways in which UCD pays off. David Benyon, Professor with over twenty-five years of experience in the field of HCI, distinguishes four ways in which UCD is worthwhile:

1. With close user involvement, it is more likely for products to meet the users' expectations and needs. This leads to an increase in sales and lowers the costs brought by customer services.
2. UCD leads to safer products, by tailoring them for people in specific contexts and with specific tasks, reducing the chances of human error arising.
3. Putting designers in contact with the users deepens a sense of empathy. This is essential in creating ethical designs that respect the users' privacy and quality of life.
4. By aiming at all users of a product, designers can recognize the diversity of cultures and human values through UCD.

4.3.3. Engagement Design Model

Interactive media artefacts are not only products; they work with messages and are more than just their design; but the design has an impact and transforms the messages. To ameliorate the interaction design of these media there has been an approach of the Experience as part of the HCI, but this was not able to deliver the needs of design. This model proposes, based on these limitations, a move from experience to engagement design on the research of interactive media.

Engagement is defined as "a quality of user experiences with technology that is characterized by challenge, aesthetic and sensory appeal, feedback, novelty, interactivity, perceived control and time, awareness, motivation, interest, and affect" by (O'Brien & Toms, 2008). The ED Model is composed by a triad of Engagement Streams, made up of universal notions created by cognitive contextual needs, subject profiles, and artefact representations that serve the design of flow paths through the expected user's experience (Zagalo, 2020).

Through player motivation tests it was found that there were very marked profiles, that tend to prefer one of the following functions – **know**, **do**, **feel** – over the others. Thus, the Engagement Streams are based on these three functions. The names of the streams were based on the correlations of empirical data, that resulted in personality profiles – **Abstracters**, **Tinkerers**, **Dramatists**. The connection between curiosity and motivation provided contextual patterns – **Problem Solving**, **Unfamiliar**, **Empathising**. And the correlation between representation in design gave the expected outcome for each stream – **Goals**, **Craftworks**, **Changes**. The relation of these three concepts give us the following three streams:

- **Progression** – a stream that makes the user go through a series of resolutions, making progress towards the desired goals.
- **Expression** – opens spaces for the creation of different world views, that result in expressive original artefacts.
- **Relation** – pushed the user to interpret people, resulting in change.

For each stream there is a 6-level categorization. There are three essential levels (marked by an asterisk (*)), and three complementary ones: **Motives**, **Users***, **Volition**, **Patterns***, **Design**, **Outcomes***.

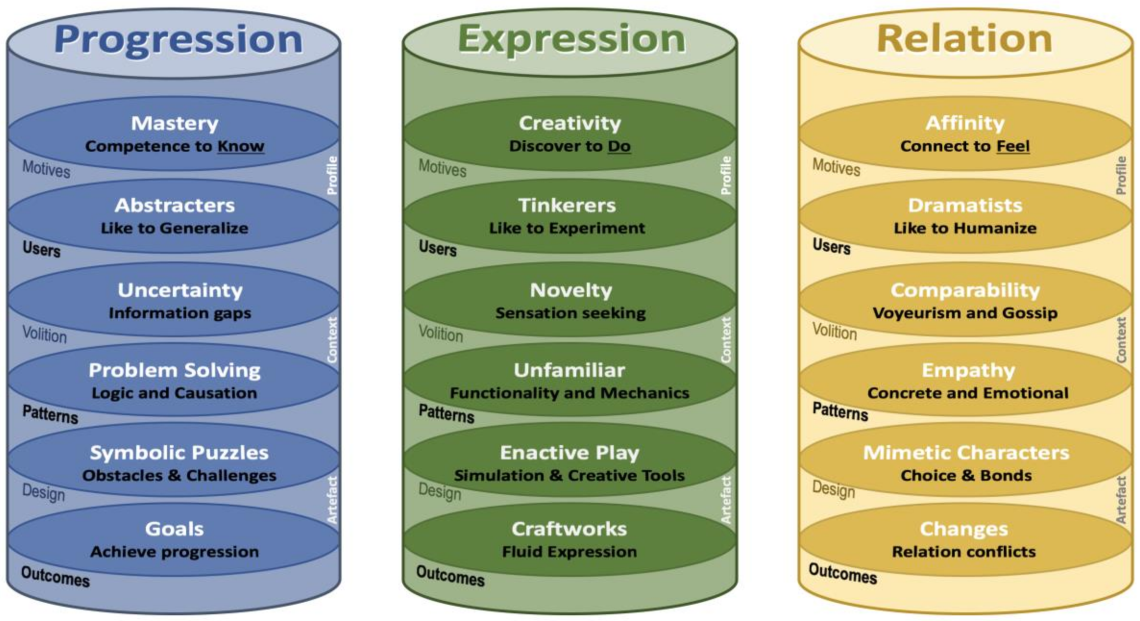


Figure 7 - The Engagement Design Model, made of three engagement streams: Progression, Expression and Relation (Zagalo, 2020)

Having the model structure described we will now discuss each different stream, giving examples of games that fit into each one.

Progression

According to Zagalo, the progression stream is based on the abstracters' profile, motivated by mastery and knowledge improvement. Abstracters are focused on formal knowledge, its usefulness, understanding the organization of information, its structure and how it can be used outside of the strict structure presented. Thus, the production of interactive media needs to be done around problem solving, allowing deduction processes, based on the response to information or solution uncertainties. This means designing mechanisms based on puzzles, that is, the construction of hurdles to challenge the player on their quest to a goal. Completing these challenges is a step forward in expertise and mastery, giving a sense of progress, effectively rewarding the player.

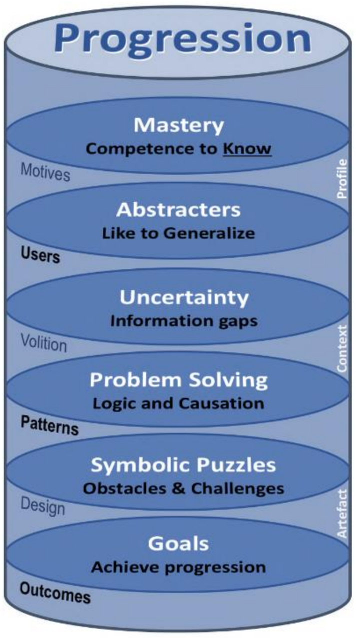


Figure 8 - Progression Stream

The most rudimentary, non-digital examples of progression engagement design are found in crossword puzzles and sliding puzzles, that help to categorize the two main puzzle modes in videogames: analytics and mastery. Pertaining to the analytic side, we can find games that involve deduction to get to solutions; on the other hand, on the mastery side, we have puzzles that involve dexterity, that need dedication and training to reach a resolution.

Figure 9 - Superliminal (Shin, 2019) & Spelunky (Yu, 2008)

For the analytic side we can find examples of puzzle games such as Portal (Swift, 2007), Myst (Miller & Miller, 1993) and Superliminal (Shin, 2019), or of strategy games such as Shogun: Total War (Simpson, 2000), Civilization (Meier & Shelley, 1991) and StarCraft (Metzen & Phinney, 1998) – these games require an analysis of in-game elements, their mental breakdown and resulting change to reach outcomes. For mastery games we have action sub-genres like platformers – Super Mario Bros. (Miyamoto & Tezuka, 1985) and Move or Die (Berbece, 2016) – fighting games – Mortal Kombat (Boon & Tobias, 1992) – or sport games – Fifa 20 (EA Romania, 2019). There are also games that are a combination of puzzle and mastery games, like shooters such as Doom (Romero, Petersen, McGee, Green, & Shanbell, 2016) or dungeon crawlers / roguelikes like Spelunky (Yu, 2008) and Diablo (Brevik, Schaefer, Schaefer, Sexton, & Williams, 1997).

Expression

The Expression stream is based on the Tinkerer, motivated by experimenting and doing things – they are driven by creativity. Content needs to provide several possibilities for creating new elements, letting players experiment with mechanics. For this stream, the design is based on the creation of simulated environments where users are free to play with variables and to create new experiences. This leads to self-made products, with the user's expression.

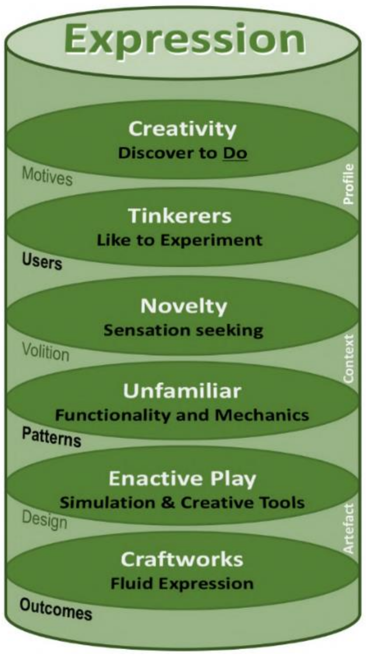


Figure 10 - Expression Stream

This stream is the least identifiable of the three, only giving its due value in the 21st century. In "Makers. The Industrial Revolution", Chris Anderson talks about tinkerers and the professions of makers, writing an entire book around this ecosystem. From crowdsourcing to the creation of new interactive tools for creation, this is a medium that as evolved, and is still evolving, leaps and bounds in the past years, accounting for a fundamental movement that involves tinkerers and their transformative capacity. This stream puts into evidence the preference for autonomous behaviours, that have been enhanced by recent technologies.

The basics of this stream can be found in maker kits, like Arduino kits that are more open in their nature, or in more recent kits for diverse topics, like homebrewing and cooking, that are usually packed with precise instructions. Thus, we can define two different profiles for creation kits: guided kits, with distinct instructions and guides to follow; and open kits, that let the user be creative and expressive when assembling products. On the more guided side we can find creation-based games like Super Mario Maker (Hino & Hosaka, 2015) and simulators like Universe Sandbox (Dixon, 2012). In the more open artefacts, we can find games that are sandboxes like Minecraft (Persson & Bergensten, 2011) or Spore (Wright et al., 2008), and virtual worlds like Second Life (Rosedale, 2003).

Figure 11 - Minecraft (Persson & Bergensten, 2011) & Spore (Wright et al., 2008)

There are also other artefacts that may require the user to learn programming languages, like Roblox (Baszucki & Cassel, 2006), dipping the users toes in the progression stream. These, however, are more focused on the creative side, thus, not belonging to the progression stream.

Relation

This stream's users are dramatists, motivated by the connections with other humans, and have the desire to humanize everything that they are related to. Patterns stem from the interrelation of people, permitting the particularization of feelings, boosting empathy. Situations allow dramatists to compare themselves and one's mental models to other peoples, indirectly through gossip and voyeurism. The design for this stream is based on the performance of human conflicts that can give an analysis and experimentation of changes, giving choices and decisions about relations, following the practices of behavioural mimicry.

Like Progression, the Relation stream feels like a classic approach to the world. Everyone likes a good movie, and a good story, with people like us and like the one we like. We like seeing into the lives of others to get the feeling of sameness or superiority. The focus of the stream is the relationships, their connections, dependencies, and dominions, and seeing the world made up by social relations. Putting humans above information, knowing that masses are made of individual subjects, all different. Each person is a world that deserves time and effort, that each part is more that the whole.

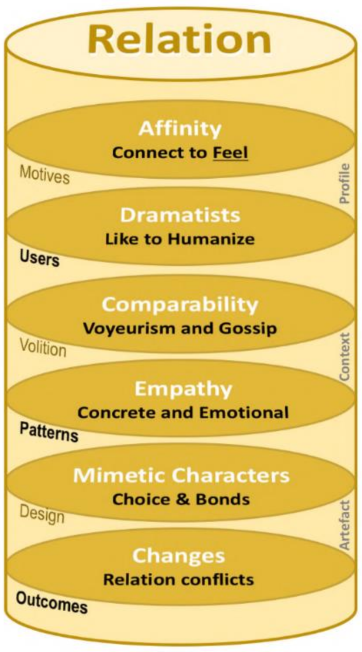


Figure 12 - Relation Stream

At the basis of this stream, we can find the Doll's House, that give us a window into the home and the characters that populate these spaces. We can see two main modes: the linear, that is marked by the spaces and hierarchy of the dollhouse; and the choices mode, based on the freedom of the characters that occupy these spaces, like dolls. In the linear field we can find artefacts like Gone home (Gaynor, 2013) or Florence (Wong, 2018); for the choice we have episodic games like The Wolf Among Us (Kaufman, Pinney, & Casillas, 2013) or The Walking Dead (Vanaman, Rodkin, & Darin, 2012). In linear games we are following the characters story, not being able to change them. On the latter the player must resolve the conflict, changing the story. These narrative structures can have various genres, just like movies, being very diverse on their form.

There is the need to explain why we do things. Social comparison and human improvement through social interaction might explain the need for socialisation and the pleasure we derive from observing others. The foundation of the Relation stream is the motivation to feel. We search for knowledge but, unlike in Progression, we do not attain goals, nor do we progress. Feelings make dramatists interact, privileging the human and the feelings, instead of abstract reasoning.

Figure 13 – Florence (Wong, 2018) & The Walking Dead (Vanaman et al., 2012)

5. Methodology

Due to the documental nature of this project, it was decided that it would be best to use a Research and Development methodology, based on the Kanban, a lean design and development method. Below we will discuss the Kanban and how it was used in this project.

Lean software development is an adaptation of lean manufacturing procedures and values and is described as "...a way to do more and more with less and less - less human effort, less equipment, less time, and less space - while coming closer and closer to providing customers exactly what they want" by Womack and Jones (1996).

The Kanban, which can be translated to signboard in Japanese, is a scheduling system created by Taiichi Ohno (1988) for use in the Toyota Company that was later adapted from the automobile industry and changed to create a software development, as described in David Anderson's book Kanban (2010).

In its most simple form, a Kanban board is comprised of three columns populated with cards. Each card represents a task, and each column serves as a bin to group the task based on if they were not started yet, if they are being worked on, or if they are done.

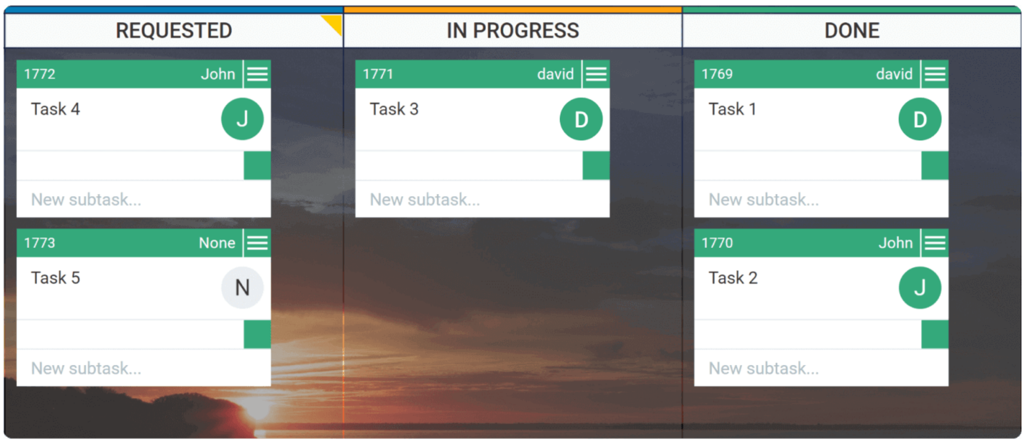


Figure 14 - Kanban Example

As tasks are started and completed the cards are moved to the respective column and this gives us a quick and visual representation of the amount of work in each state and, if well organized, can give us foresight of the near future of the design and development process.

The simple three-column layout can be expanded and adapted to fit each projects needs to have more columns with more process steps or using different colours to signal importance or types of tasks.

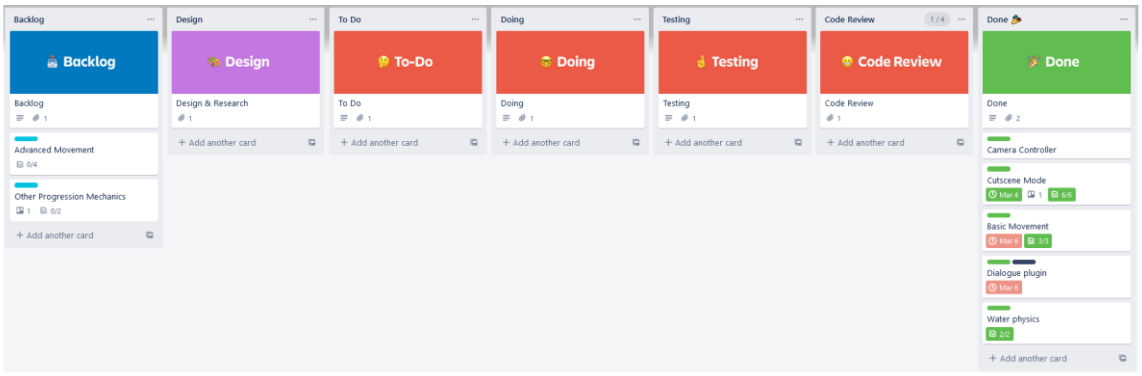


Figure 15 - Progression stream board

As with most popular work tools it was adapted from a whiteboard with sticky papers with tasks to a digital system, using Trello – a website that lets us create completely customizable kanban board, with as many columns as one needs and with cards that can have different types of media like images or links to websites, also letting us create a deadline for each of the tasks.

This methodology gives us a better understanding of whole process of design and development (Gulliksen Stray, Moe, & Dingsøyr, 2011; Ikonen, Pirinen, Fagerholm, Kettunen, & Abrahamsson, 2011; Polk, 2011; Senapathi, Middleton, & Evans, 2011), which is very important since our goal is to analyse the process of the design and development of a new indie game while using the Engagement Design Model as a guideline for the design choices. This better understanding of the process leads to a more detailed and realistic documentation of the work process and, ultimately, helps greatly in our research by reducing the clutter and noise of said documentation. This was the main reason for the use of the Kanban, due to the importance of the work documentation used in the end to create design lessons.

The Kanban also helps us keep up with deadlines, since the time allocated to the design and development of this project is much shorter than what is expected for the process of development of an indie game, by decreasing the time to delivery (Middleton & Joyce, 2012; Taipale, 2010) and by helping problem solving (Ikonen et al., 2011), helping us reduce the amount of bugs in the project and by aiding us in the detection of bugs in an easier and faster way.

Since the design and development process is heavily influenced by the Engagement Design Model it was decided that each of the three design streams – Progression, Expression and Relation – would have its own board, so that the design and development for each of the streams was segmented and did not bleed much into the other streams.

Figure 16 - Kanban boards for the three streams

Each of these three boards was then divided in to a seven-column process, which was deemed the best for this type of development:

1. Backlog

When a task is idealized it is added to the backlog column, which olds all of the tasks, regardless of their feasibility and difficulty. When a task is added to this column it then receives a tag of "Important" or "Quality of Life" which then helps us organize the tasks in importance order.

2. Design & Research

Before starting the development process of a task, it goes into a design and research process, where it is studied and designed to a point where we can decide if the task is valuable enough to invest the time and resources to actually implement. If deemed important enough to develop the design of the task is then completed using the Engagement Design Model as a guideline.

3. To-Do

After this research and design process the tasks that are moved to this column and become the task next in line for development as soon as there is time.

4. Doing

This column holds all of the tasks that are in progress, and each of the cards in this column will receive updates and comments to document the process and to mark the "doneness" of a task.

5. Testing

After a task is implemented it is then tested, to check if all of the developed features are working properly and to see how it affects already present features.

6. Code Review

If the features are working properly all of the technical work is then reviewed and cleaned up, giving us a better and more organized work.

7. Done

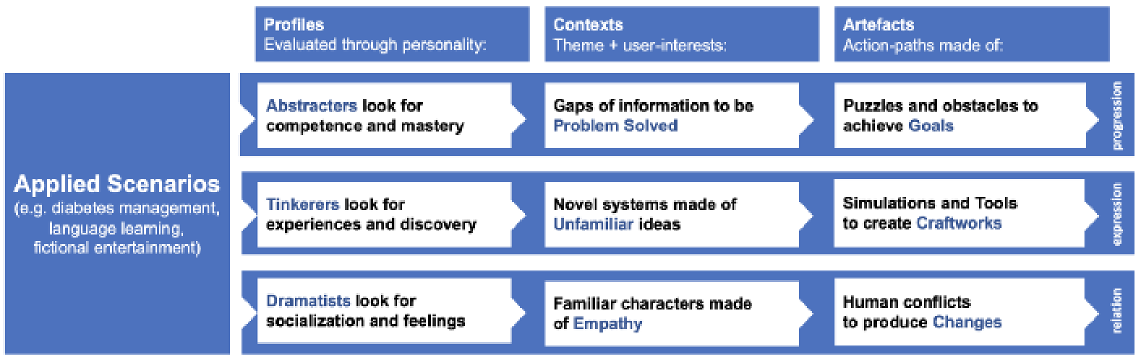
Finally, the tasks are moved to this column to signal that they are done, giving us a visual aid to see what percentage of work is already completed.

6. Design & Development

The aim of the project, as stated in previous chapters, is to develop an indie game, aimed at all types of players, by mirroring the three engagement streams from the Engagement Design Model (Zagalo, 2020) in three different parts of the game, and to document the process of said development to understand what lessons can be learned for a rapid design and development process, while preserving authorship. The choice of the usage of the ED Model came from the way it presented itself as a more hand-on and practical model which gave us strategies on how to employ each engagement stream, instead of simply describing a player type, like other player taxonomy models. With this in mind, there was a need to develop a functional demo of a game containing different types of engagement, alluding to the three engagement streams described in the Engagement Design Model and in previous chapters.

Each stream is comprised of six levels, that can be combined to create three major levels: **Profile**, comprised of the **Users** and the **Motives**, defining the profiles of the players with their denomination and their motive; **Context**, comprised of **Volition** and **Patterns**, giving us what makes each streams user tick; and **Artefact**, comprised of **Design** and **Outcomes**, what materializes the context and what comes from this materialization.

This chapter will serve as a technical and game design documentation, referencing the Engagement Design Model's structure of Profile, Context and Artefact, while discussing core features and design choices made during the game design and development process.



Profiles
Evaluated through personality:
Contexts
Theme + user-interests:
Artefacts
Action-paths made of:
Applied Scenarios
(e.g. diabetes management, language learning, fictional entertainment)
Abstracters look for competence and mastery
Gaps of information to be Problem Solved
Puzzles and obstacles to achieve Goals
progression
Tinkerers look for experiences and discovery
Novel systems made of Unfamiliar ideas
Simulations and Tools to create Craftworks
expression
Dramatists look for socialization and feelings
Familiar characters made of Empathy
Human conflicts to produce Changes
relation

Figure 17 - Demonstration of the model applied to design scenarios (Zagalo, 2020)

In the figure above, Zagalo summarizes each major level for each of the engagement streams, creating a new representation of the model in the applied logic of the goal-directed process, making it easier to understand what each level means and how to better realize the knowledge present in model.

6.1. Progression Stream

In accordance with Zagalo and based on the findings of Yee (2015) that identified Mastery and Achievement as two of the six main motivations in players, the progression stream is based on the abstracters' profile. Abstracters are driven by mastery and knowledge expansion. They are concentrated on formal knowledge, its usefulness, understanding the structure of information, its structure and how it can be used outside of the strict structure presented. Therefore, the production of interactive media needs to centre around problem solving, allowing deduction processes, based on the response to information or solution uncertainties. This means designing mechanisms based on puzzles, that is, the construction of hurdles to challenge the player on their quest to a goal. Completing these challenges is a step forward in expertise and mastery, giving a sense of progress, effectively rewarding the player.

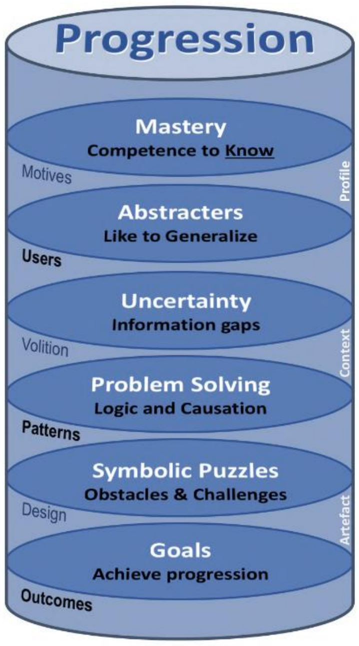


Figure 18 - Progression Stream

6.1.1. Profile

From the beginning it was set that the game to develop would be a platformer. This game genre was selected due to two main reasons. Firstly, when creating a project with this scope it is important to reduce the complexity of certain design aspects as much as possible, and platformers are one of the simpler, yet expansive, game types to develop as seen in the sheer number of platformers created through the ages. This simplicity makes it the perfect vessel for a game with multiple facets, designed to cater to all types of players. Secondly, and more importantly it is the game genre best suited for Abstracters, as seen on the speedrunning community. Speedrunners fit, more often than not, in the Abstracters profile, as players motivated by the Mastery of a game with a very defined goal in their mind, finishing the game in the least amount of time possible. In speedrunning we see a myriad of different games, with different engagement types and appeals, but one of the game genres more prevalent in the scene is the 2D platformer, more specifically Super Mario Games like Super Mario World (Miyamoto et al., 1990) and modified versions of this game to include more challenging levels and puzzles, like Super Dram World 2, created by PangeaPanga, a famous level designer in the scene. It was also decided that the game would fit in the *metroidvania* genre. In this genre, inspired by and named after Metroid (Yokoi & Okada, 1986) and Castlevania (Nagata & Akamatsu, 1986), games have an expansive world that has certain areas gated, by the lack of player ability and yet to be revealed mechanics, that are unlock throughout the game by the player completing tasks and unlocking new power-ups and abilities.

Figure 19 – Metroid (Yokoi & Okada, 1986) & Castlevania (Nagata & Akamatsu, 1986)

For the Progression Stream the design focus was the gameplay feel and making the game speedrun, the act of completing the game as fast as possible, friendly. For this to happen, the first and most important thing to do was the player movement and all that entails.

Firstly, there had to be done a research to find games from the same genre in order to determine what types of movement and what mechanics where the best fit for Cave Game, the game to be developed.

Games like Super Mario Bros. (Miyamoto & Tezuka, 1985), Spelunky (Yu, 2008) and Super Meat Boy (McMillen & Refenes, 2010) come to mind when thinking about satisfactory movement in platformers. All of these games gave important insights about the design of the movement, but one game stood above them all as a main inspiration for the game feel. Celeste (Thorson & Berry, 2018) is an award-winning platformer, known for its engaging story, retro looks and beautiful game design and feel.

Figure 20 – Spelunky (Yu, 2008)

In Celeste (Thorson & Berry, 2018), we find a simple, yet refined, set of moves that can be chained and used in specific in-game situations resulting in different moves. The player can move horizontally, jump, grab walls, climb said walls and, finally, dash once, with the dash being refreshed when the player touched the ground. These moves can be used in a myriad of ways, such as using a dash after the player is launched by some in-game object in order to reach greater velocities and leap greater distances.

Figure 21 – Celeste (Thorson & Berry, 2018)

It was this "simple yet refined" approach to movement that served as a main inspiration for Cave Game's player movement mechanics. To give the movement a greater depth, physics were used to prompt actions, such as applying a sideways force to the character for the horizontal movement or applying an upwards force on the character to simulate a jump. With this movement model it was possible for the player explore these mechanics in order to find new ways to use them to traverse the level.

Also inspired by Celeste (Thorson & Berry, 2018), there were added a number of quality of life features to make the movement feel even better. Features like **jump corner correction**, that makes the player move sideways if the character hits a corner when jumping, **halved gravity jump peaks**, that make the jump floatier while the jump button is held in order to give the player greater in-air control, and **landing correction**, that gives the player a little help when landing after a jump in order to get them up a platform if they miss the top by a few pixels, where added to make the movement a lot more enjoyable and to keep the feeling of flow when playing the game.

Players that fall into the Progression Stream also tend to like to solve puzzles and engaging in different activities that get them to advance in the game and that give them a sense of accomplishment. This served as a catalyst to the idea of implementing puzzles in the levels, such as logic and physics puzzles.

For Cave Game these puzzles would be closely connected to the *metroidvania* roots of the game, functioning differently at different times, based on the power-ups the player has already obtained.

For this playable demo, the power-up chosen to be implemented would be the ability to walk underwater, and the puzzles created for the demo would be centred around the use of water, whose implementation will be detailed in a future paragraph, and the interaction between the player and the water, that will now be explained.

Before obtaining the power-up at the end of the demo, that gives the player character the ability to sink and walk underwater, bodies of water make the player float and impede the player's movement by reducing the jump height and the speed of the character.

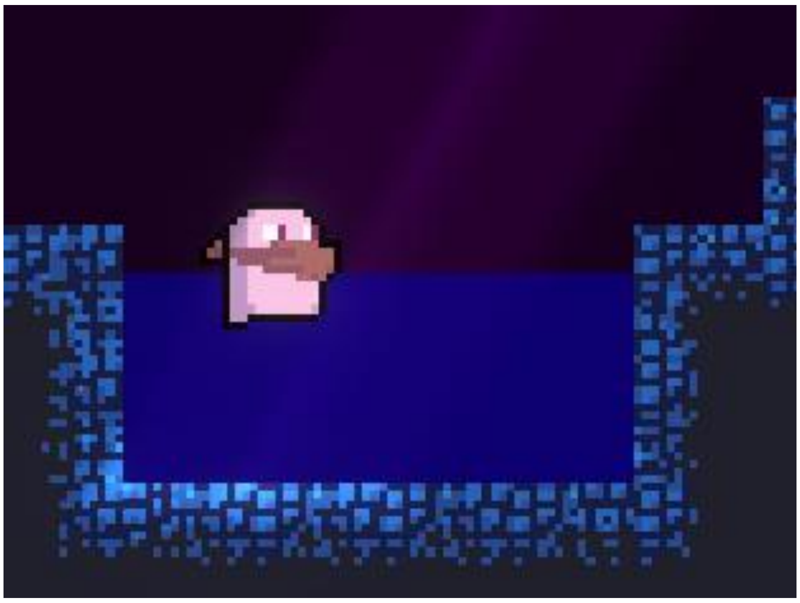


Figure 22 - Player floating example

These mechanics are used in physics-based puzzles that make use of the player's buoyancy in water to create inverted platforming challenges, where the player needs to use the water to go under obstacles as if the gravity is inverted, or where the water is used as an elevator for the player to reach higher parts of the map.

This "character buoyancy" is also used to gate sections of the map that are flooded, meaning the player can only get to these rooms after finding the power-up that, once obtained, makes the player sink to the bottom of bodies of water. While at the bottom of bodies of water the player has a floatier and greater jump, and still suffers from a decreased movement speed.

Having most of the movement mechanics in place it was time to define the map layout and the type of camera movement that made most sense for the demo. Still taking inspiration from Celeste (Thorson & Berry, 2018), it was decided that the map layout would be divided in singular rooms with individual challenges.

In platforming games, we usually see a side-scrolling camera that follows the player along the level, moving horizontally with the character staying roughly in the centre of the screen. We see this type of camera movement in games like Super Meat Boy (McMillen & Refenes, 2010), Super Mario Bros. (Miyamoto & Tezuka, 1985) and Castlevania (Nagata & Akamatsu, 1986).

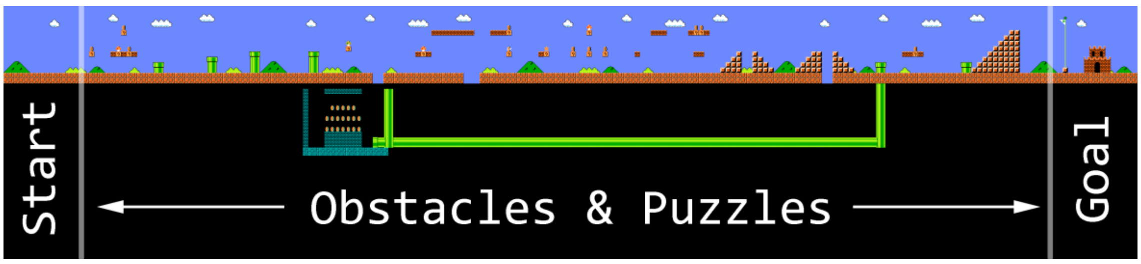


Figure 23 – Super Mario Bros. (Miyamoto & Tezuka, 1985) World 1-1 (division according to the ED Model's Artefact)

For Cave Game this type of camera movement made little sense when taking into consideration the room layouts. To accommodate the layout, the camera movement moved away from the more conventional camera in platformers and went with a less dynamic approach, having a fixed camera showing a full room and the player moving within said room. When The player leaves the room to enter a new one the camera follows the player to show the new space, animating this transition between rooms.

We often see this type of camera movement, not in platformers, but in action and adventure games, like The Legend of Zelda (Tezuka, Miyamoto, & Terui, 1986) where the player moves in around the world in segmented zones, transitioning between zones with this type of camera movement. This also happens inside dungeons, where there is a division in singular rooms and the player moves freely within the room, while the camera only moves to follow the player into a new room.

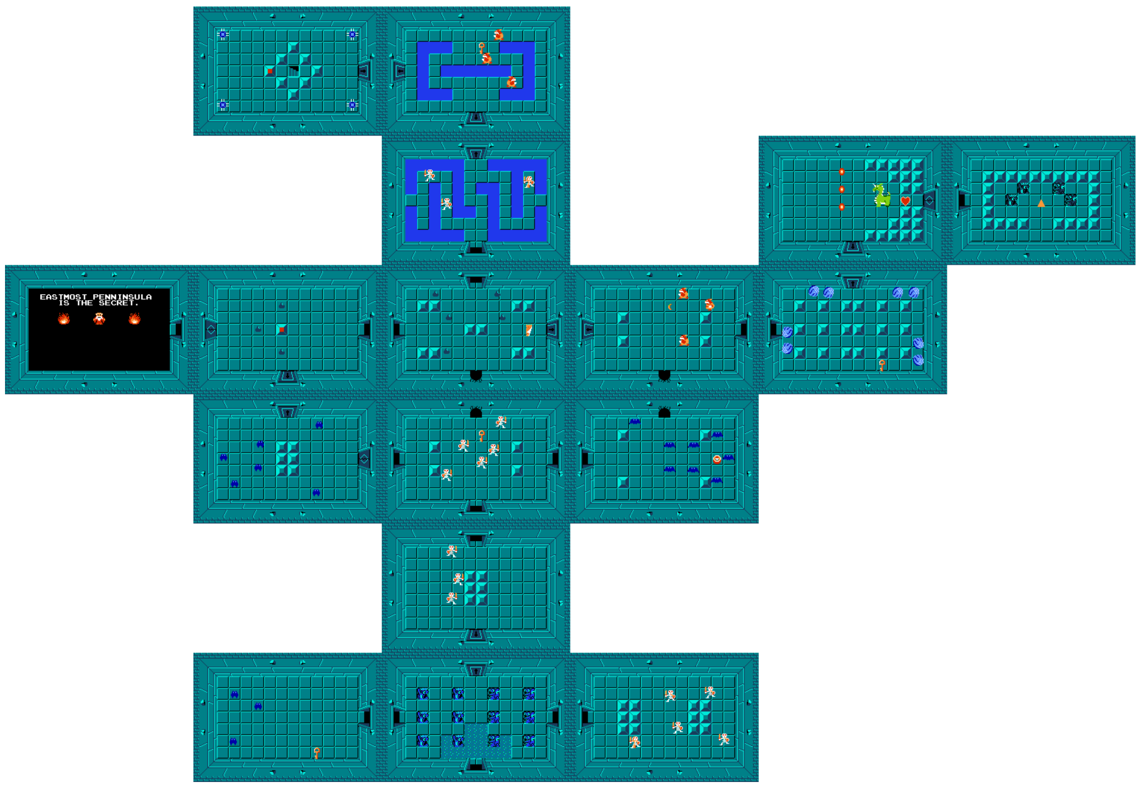


Figure 24 - The Legend of Zelda (Tezuka et al., 1986) - First Dungeon

These choices made it is possible for the player to have a greater control on the character, not having to worry about unseen obstacles and not having to adapt to the environment moving around the character.

6.1.2. Context & Artefact

The Progression Streams context tells us that Users are moved by gaps in information that can be problem solved through the artefacts, puzzles, and obstacles for the player to achieve a goal.

After deciding and developing the player and camera movement there was a need to create an area where the player could move freely, to test all the work done, and that served as a framework for the final map. It was decided from early on that the map design and structure would be tile-based. This means that the map is comprised by different tiles, placed in a grid, with different behaviours. These tiles will be referred to as “blocks” from here on out, as they serve as building blocks for the map and, during the development, they were classified as blocks.

The simplest and most common block, the “Ground”, was also the one that had the most work put into it. It was decided that, from a graphical standpoint, the ground would work as a single entity that would have a connected texture and a border. The implementation of this feature was harder than expected. Firstly, there was a need to have different tiles drawn, corresponding to the different states the Ground block could have, totalling sixteen states.

The next step was to give the block the ability to determine in which state it should set itself. For this to happen the block need to be able to read the neighbouring blocks. Having this information the block is able to determine which state it should be in using the following rules: firstly the block checks the blocks directly up, down, left and right, and removes the border on every side that touches another Ground block; secondly, the block checks the corners and removes the corner border on corners that have another Ground block; finally, the block sends a signal to neighbouring Ground blocks to update, if it’s state changed since the last update, otherwise the block doesn’t send out the order to update.

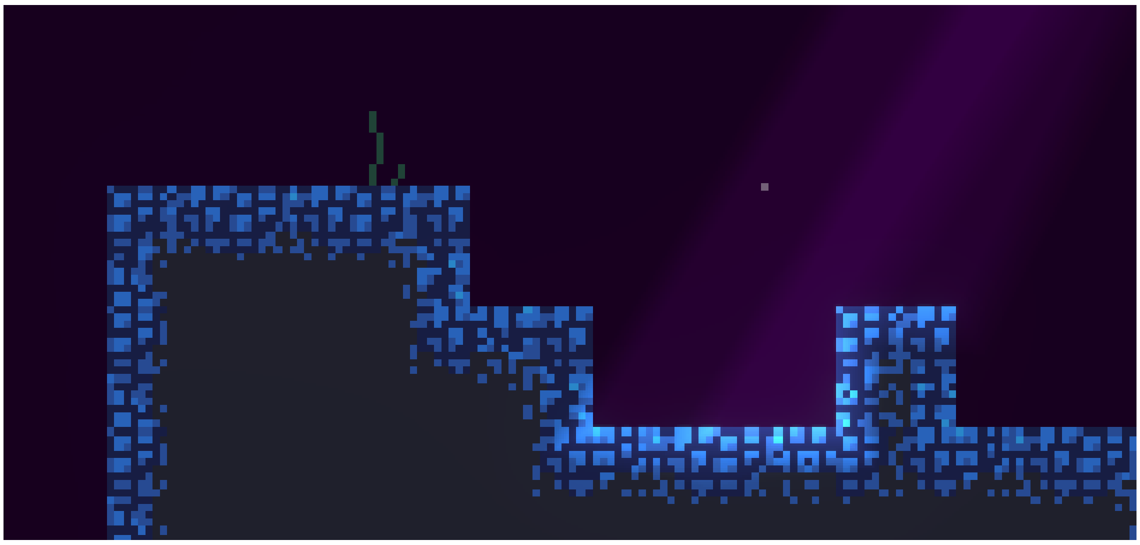


Figure 25 - Ground block states example

Also important for the level structure are the “Spawn” and “Checkpoint” blocks, used to control the flow of the game. The Spawn block has the function to spawn the player on the map. After the player is spawned the game remembers the position of the Spawn block, to respawn the player, in case of death.

Like the Spawn block, the Checkpoint functions as a location for the player to spawn. The difference between these two blocks lies in the fact that the player needs to activate a Checkpoint, by passing through it in order to respawn at a Checkpoint. When the player touches a Checkpoint the sprite of the block changes to signal that the Checkpoint is activated, and the game swaps the last spawn position to the current checkpoint.

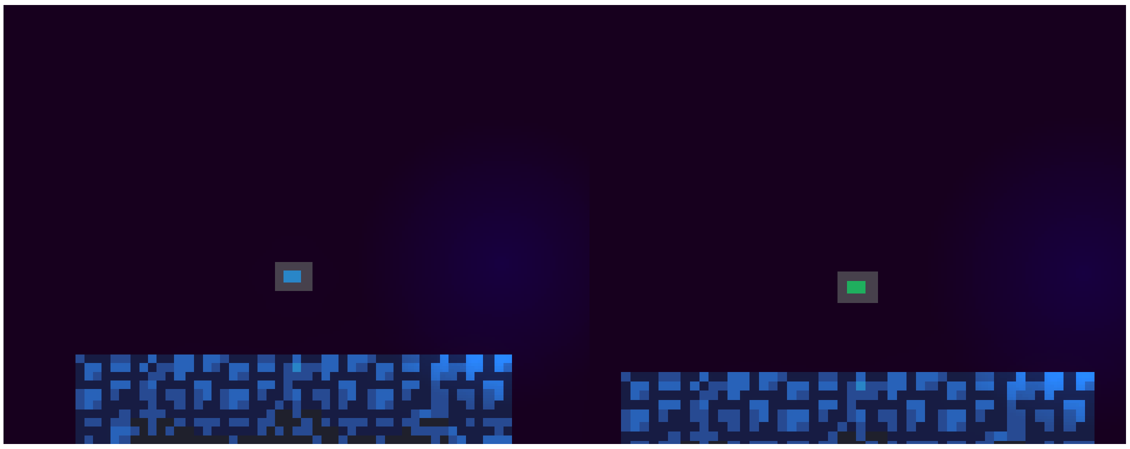


Figure 26 - Checkpoint inactive (left) and active (right) states

In order to test these blocks, there was a need to create a hazard for the player. The “Spikes” block was created with this intent. When the player touches this block a character animation is played, to signal that the player was hurt, and then the player respawns in the last Checkpoint, or on Spawn if no Checkpoint was activated. This process takes very little time to happen to make sure that the flow of the game is not broken.

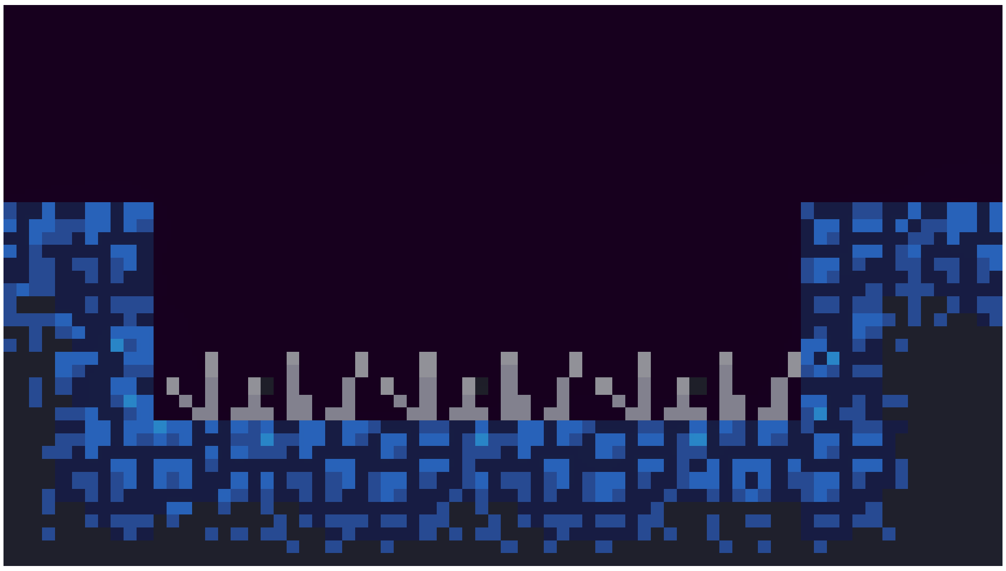


Figure 27 - Spikes example

The final structure block is the “Finish” block, created to signal the end of a custom level, created on the Level Creator, which will be explained in depth in the following chapter. This block is activated when the player crosses it and ends the custom level, sending the player to the Custom Level Selection screen, also discussed in the following chapter.

Other blocks were designed with the purpose to diversify the level design, the “Platform” and the “Spring” blocks. The Platform block functions like a semi-transparent block, in the sense that its collision box only works from certain angles. This means that the player can pass through a Platform when coming from the bottom, but the Platform works like solid ground when stepped on.

The final block to be created was the Spring block. The Spring, like the name implies, applies an upwards force on the player, springing them up, when stepped on. To signal the activation of the block the sprite changes to a sprung-up state, changing back shortly after.

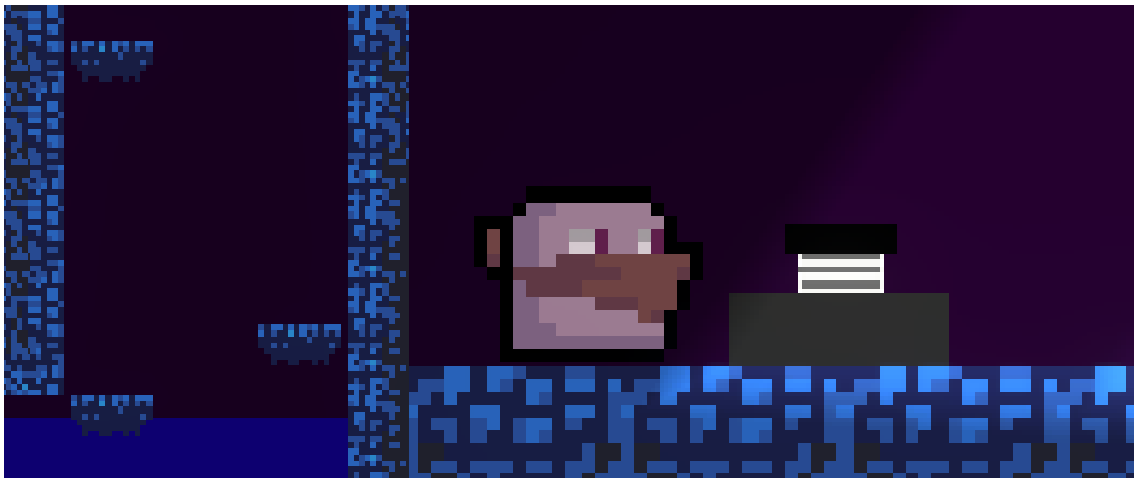


Figure 28 - Plarform & Spring blocks

Some other blocks like a "Conveyer" block, that moves the player along a conveyer belt, and the "Moving Platform", functioning much like the "Platform" but moving along a predetermined path, were designed and were in development but ended up, ultimately, scrapped due to incompatibility with the games themes, sticking out as blocks that did not make sense in the scenery.

Knowing that Abstracters are moved by the need to discover new information through the world shifted the focus of the level design focus from a "Making a platforming level" to "How do I incorporate puzzles in the level and how do 'hinder' the players' progress without the experience being stressful?" mindset. The idea to incorporate water bodies in the level that served as a "inverse platforming" using the floating mechanic described in the previous chapter surged from the need to withheld information from the player in a platforming scenario, seeing that most platformers tend to have platforming with regular gravitational forces where the player always falls in the same direction, with the exception of some games like VVVVVV (Cavanagh & Foddy, 2010) where the player can invert the gravity, making the character fall upwards.

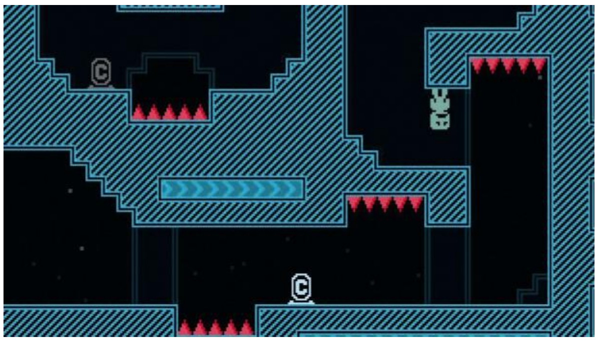


Figure 29 – VVVVVV (Cavanagh & Foddy, 2010)

Using the water bodies made it possible to create puzzles within the levels themselves, and made the player stop and think on how to continue through the level until the they discover that with enough speed it is possible to use the water bodies to traverse the level. This layout also gave some fluidity to the levels for players replaying the game in a speedrun type of playthrough, as seen on the figure below. All of this feeds into the Abstracter's volition: Uncertainty – not knowing how something will work until tried.

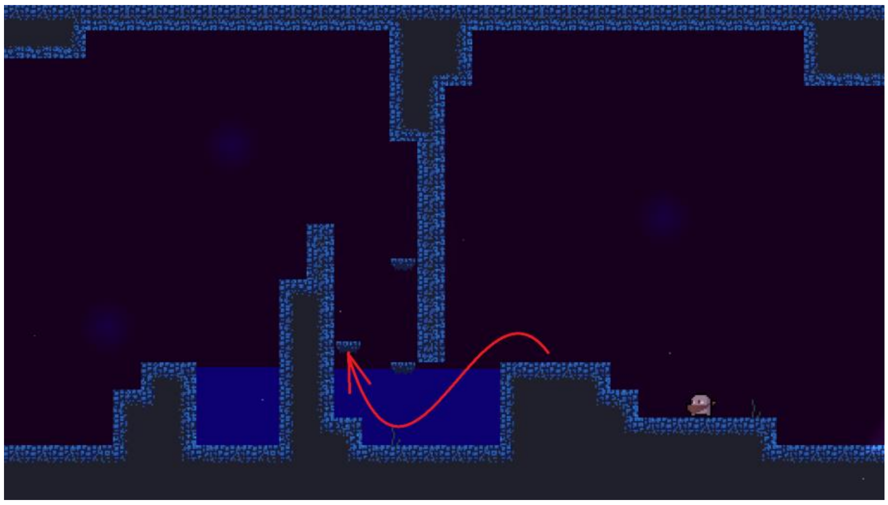


Figure 30 - Level flow example

Implementing water bodies and all the physics attached was a difficult task, but with the help of some articles describing the implementation of spring and particle physics in games (Hoffman, 2012) it became a much more possible task.

In a very simple way, there are two main components to the water bodies: the surface and the body itself. Firstly, and most importantly, we will explain the surface, as the body of the water depends greatly on the surface.

We can think of the water surface as a series of points on the end of vertical springs. Each spring has a default size and can be stretched or compressed and will try to return to its original size when in these states.

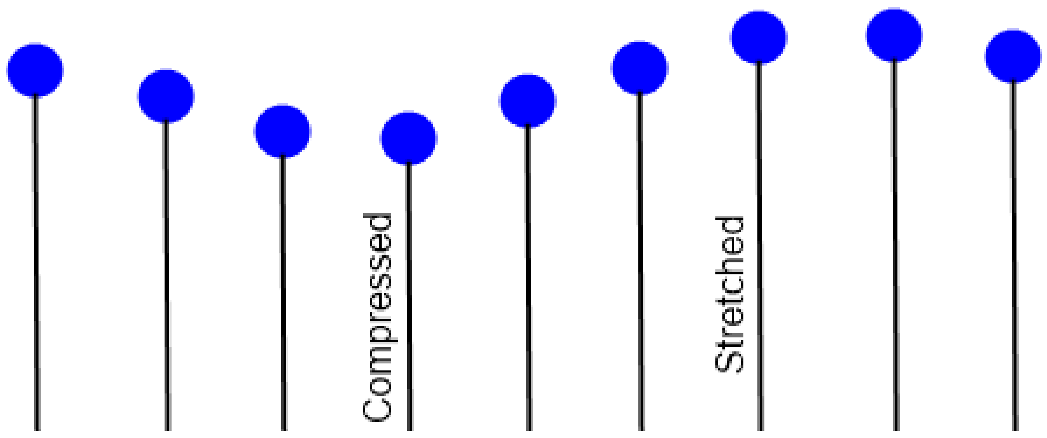


Figure 31 - Spring representation (Hoffman, 2012)

In order to simulate waves, each spring will have an impact on their neighbour, pulling them as they stretch and compress, spreading waves through the surface. These calculations are done using Hooke's Law (Hooke, 1660), Newton's Second Law of Motion (Newton, 1687) and adding dampening to the tension of the springs, resulting in the following formula to calculate each particle's acceleration.

$$a = -\frac{k}{m}x - dv$$

This formula gives each particle its acceleration and lets us control how much "springiness" our surface will have. For example, by altering d, the dampening factor, we can change the way our waves will propagate. If the dampening factor is high the water will behave like a thicker liquid, such as molasses, and if the dampening factor is lower waves will propagate for a longer time.

The second part of the water is the body itself. This part is much simpler, as the code knows the water body bounds, using a rectangular collider, and draws a trapezoid using the surface as the top side of the shape. Since graphics cards cannot draw trapezoids, the shape is divided into two triangles, being then rendered on the screen.

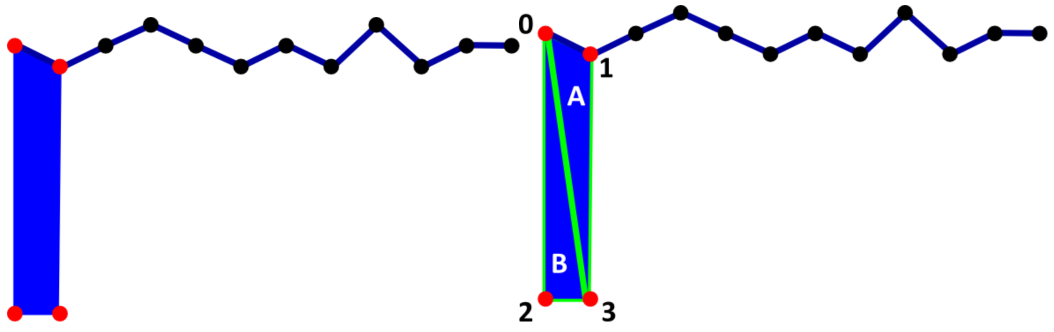


Figure 32 - Trapezoid drawing representation (Rose, 2014)

All of these things combined result in a simple yet satisfying body of water that, as a puzzle mechanic or as a gatekeeping mechanism to keep the player away from desired locations until they can traverse underwater, serves to create uncertainty in the player's experience and makes them want to master this new type of movement.

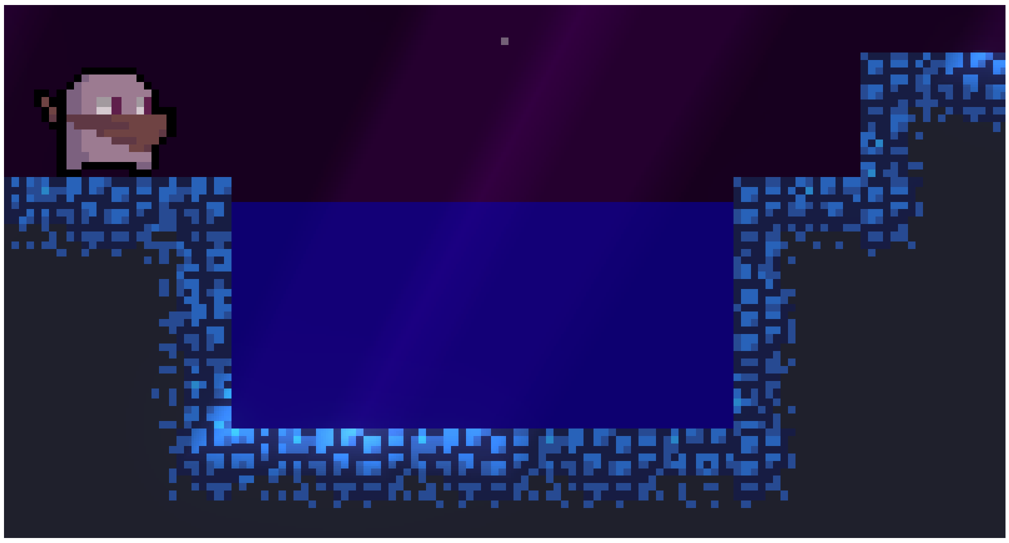


Figure 33 - Water body example

The last great task for this engagement stream is the design and implementation of a world map and the rooms that are the building blocks for this world.

The overall layout of the map went through several changes but was always a simple looping path with some detours due to the nature of the playable demo.

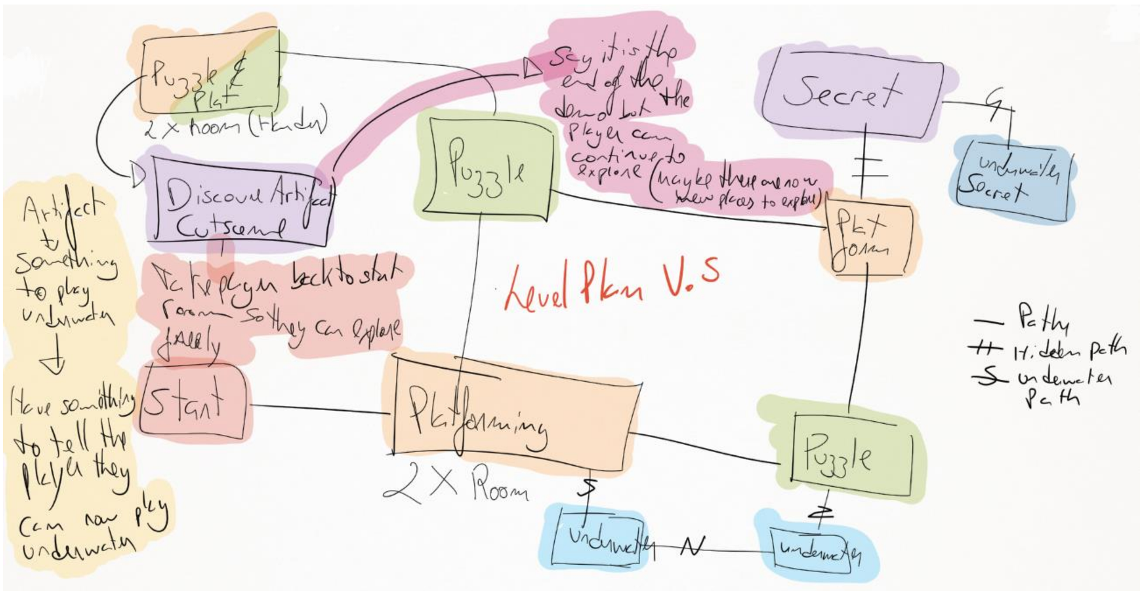

Puzzle & Plat
2 x Room (Hidden)
Discover Artifact Cutscene
Say it is the end of the demo but the player can continue to explore (maybe there are now new places to explore)
Secret
Underwater Secret
Puzzle
Plat form
Level Plan V.5
Artifact to something to play underwater
Have something to tell the player they can now play underwater
Take player back to start room so they can explore freely
Start
Platforming
2 X Room
Puzzle
Underwater
N
Underwater
— Path
# Hidden path
S underwater Path

Figure 34 - Initial map layout

The rooms were divided into five categories: **Platforming**, **Puzzle**, **Underwater**, **Secret** and **Structural**. **Platforming** rooms are rooms based around platforming challenges, such as jumping over gaps. **Puzzle** rooms are rooms that function as a pace regulator and that give the player different challenges in the form of puzzles. **Underwater** rooms are self-explanatory, rooms that are underwater. **Secret** rooms are rooms that are not part of the main story and are only there for players that tend to explore more of the map. Finally, the **Structural** rooms are rooms like the initial and the final room that serve as structure of the map, starting and ending the map.

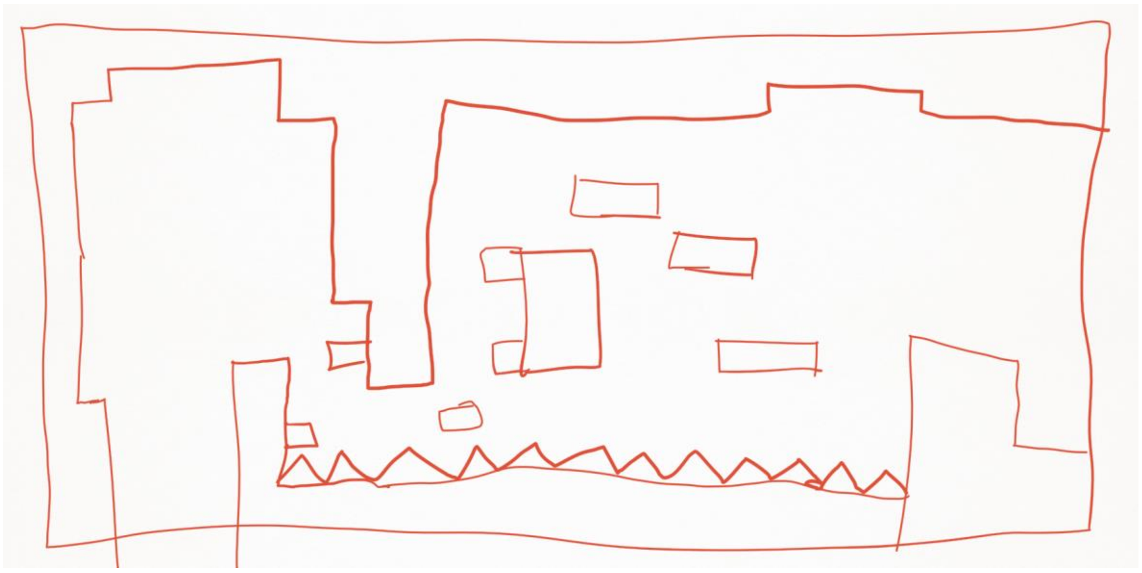


Figure 35 - Platforming room example

Every rooms starts as a simple drawing, to roughly define setting and layout. After that, the level is constructed in the game engine, using grid, with the blocks explained in previous paragraphs.

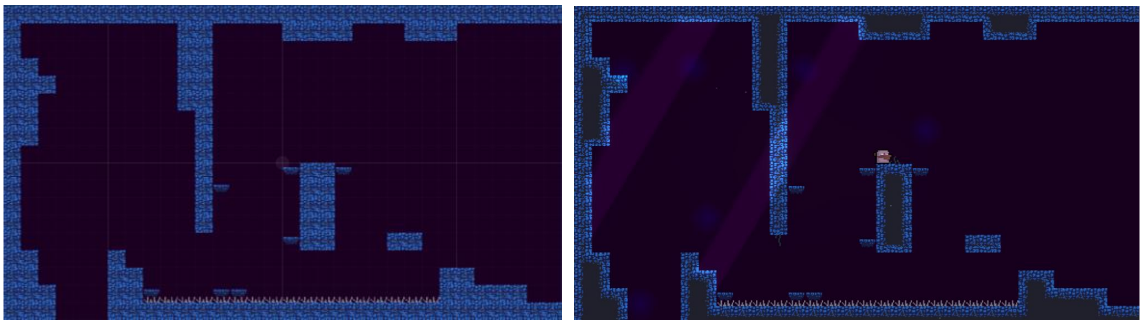


Figure 36 - Room constructed (left) & finalized (right

After creating the designed room in the game, it enters a state of testing and tunning. Playing through the room gives us the possibility to see, for example, if the distances between platforms or the height of walls is correct for the player to be able to do them, and to see if they pose a greater or smaller challenge than expected. Abstracters strive to accomplish a goal, but for them to feel this accomplishment they must go through the process of solving puzzles and surpassing obstacles to reach said goal. This fine tuning of each room helps thread the fine line between creating a boring room with no challenge and creating a very difficult and frustrating one, leading us to a room with just the right level of "frustration" and accessibility. After this technical process it is time to finalize the room. To do this lighting is added and background details, like small plants and dust particles are inserted into the room, to give the feeling that that space belongs to a world.

After all the rooms were created it was time to lay them out in the world to create a level. Every room was created in order to connect seamlessly with the following and preceding rooms, to make the level construction go as smoothly as possible.

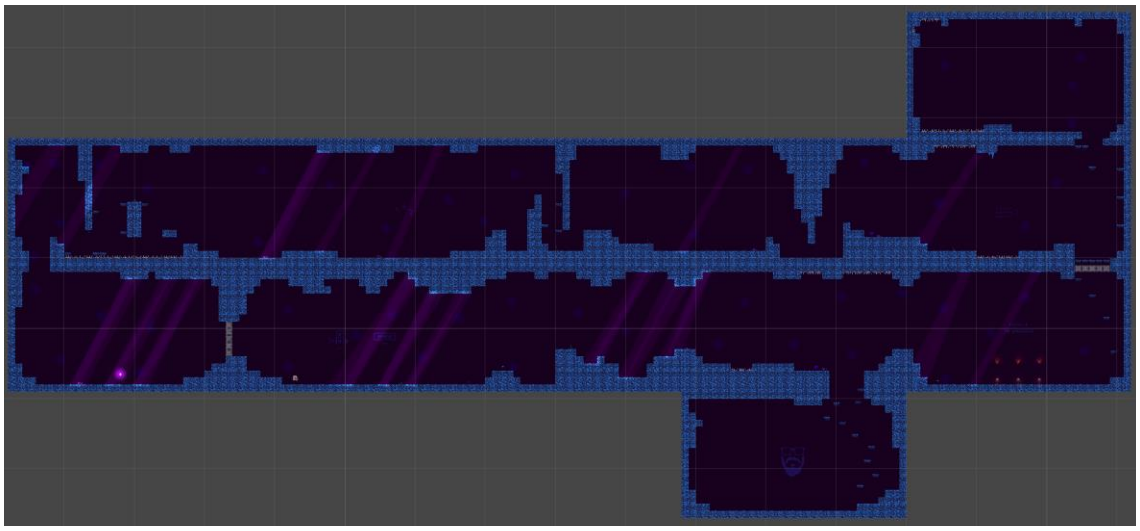


Figure 37 - Level layout

After building the level it was time to create a simpler puzzle, to control the players pacing and to give them a varied experience throughout the world. This simple puzzle also sets precedent for the possibility of more puzzles down the line.

The puzzle is a simple colour matching one, asking the player to match the colours of pedestals with the colours of the light directly above said pedestal. Until this task is completed the game gates the players progression by way of a wall on the top of the room.

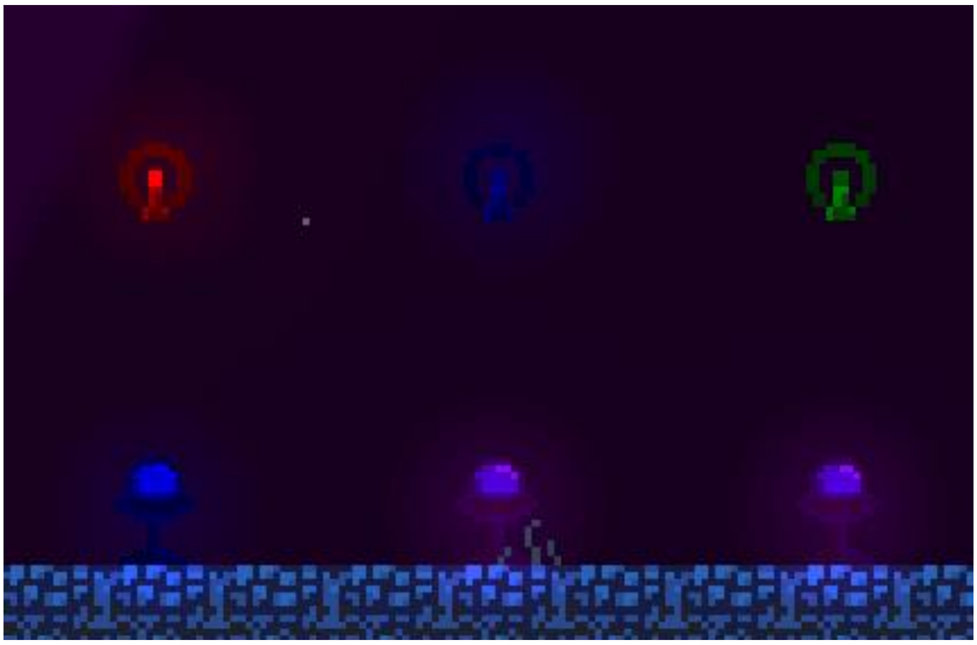
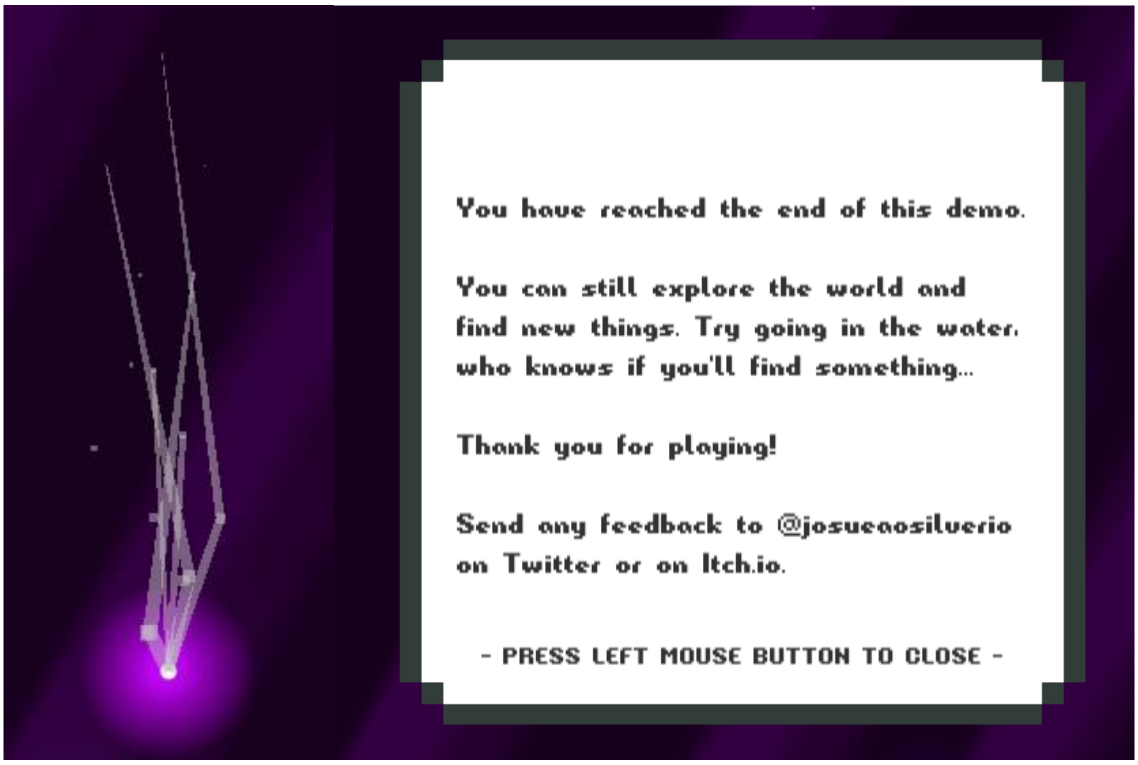


Figure 38 - Light puzzle

When the game starts, the lights colours are randomized from a set of six different colours. After that, the process is repeated for each of the pedestals. Finally, the game checks for the possibility that the puzzle has solved itself when setting up. On the off chance that this is the initial setting the game resets the puzzle and starts the randomizer again, setting new colours, until there is a usable setup for the game.

To solve this puzzle, the player must interact with each pedestal, cycling the colours, until the colour matches the one on the light above. When all the pedestals match their respective light, the game opens the gate on the top of the room, letting the player continue.

All of the aforementioned features create a **problem-solving** experience, that takes the players **uncertainty** and uses it as a motivation, making the player want to continue to advance in the game, **completing puzzles and overcoming obstacles** in platforming rooms, in order to reach the final **goal** – the power-up placed in the final room.


You have reached the end of this demo.

You can still explore the world and find new things. Try going in the water, who knows if you'll find something...

Thank you for playing!

Send any feedback to @josueaosilverio on Twitter or on Itch.io.

- PRESS LEFT MOUSE BUTTON TO CLOSE -

Figure 39 - Power-up (left) & End game information (right)

When the player reaches this final room, they are gifted with a power-up, giving them the ability to tread underwater and giving them access to formerly inaccessible rooms due to the water. For the Progression stream a goal, the outcome of the stream, serves as a way to advance progress. To tell the players that they have progressed and that all their hard work was not for naught. Much like a marathon runner receiving a medal at the end of a race, for Abstracters what matters the most is overcoming obstacles and solving puzzles, but the goal validates this behaviour and rewards the player for doing so, creating a loop where players traverse through the level available to them **solving puzzles** and **overcoming obstacles with the goal** to **find new powerups** letting them **access new information** and new sections of the map where players will then go to the beginning of the loop and repeat this process.

In the following chapter we will explore the design and development for the Expression stream, more focused on creative expression, and the creation of a Level Maker.

6.2. Expression Stream

The Expression stream is centred on the Tinkerer profile. Tinkerers are driven by the need to experiment and to create – propelled by creativity. Interactive products need to provide multiple opportunities for creating new elements, giving the players space to experiment with mechanics. In this stream, design is centred on the creation of simulated settings where players are free to tinker with variables and are able produce new experiences. This leads to self-made products, with the user's expression.

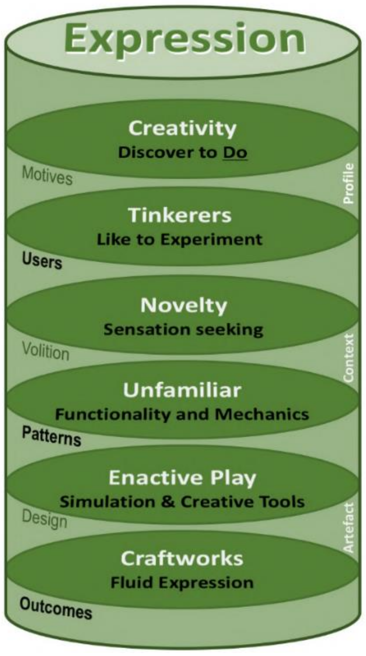


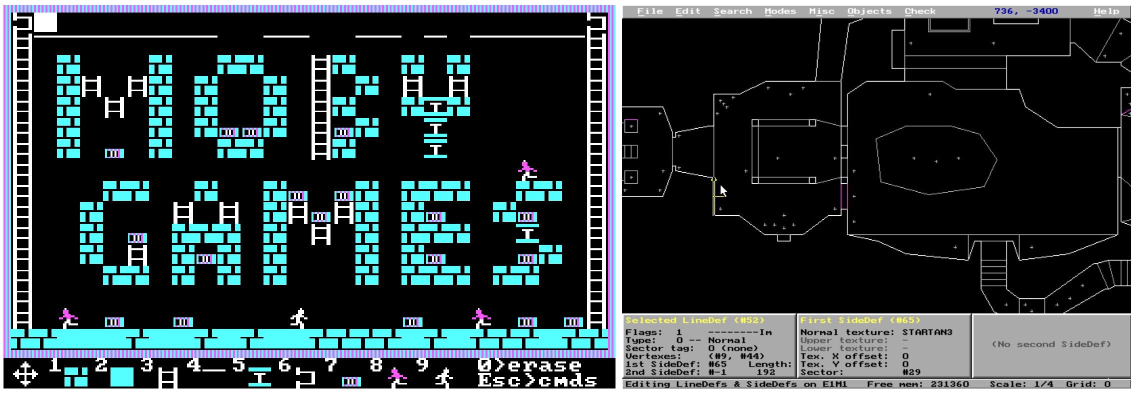
Figure 40 - Expression stream

6.2.1. Profile

As stated in a previous chapter, the Expression stream is based on the Tinkerer, motivated by experimenting and creating things – driven by creativity. Content made for this stream needs to provide several possibilities for the creation of new elements, letting players experiment with mechanics. For this stream, the design is based on the creation of simulated spaces where users can play with variables and create new experiences. This leads to self-made creations, with the user's expression.

For Cave Game, the expression of this stream was decided as soon as the game genres were defined. Cave Game can be categorized primarily as a Platformer and secondarily as a Metroidvania; both of these genres are great conduits for level creators and editors.

We can date level creators as far back as 1983, with Lode Runner (Smith, 1983). This platformer was one of the first games to feature a level creator where players could make their own levels and tailor the game experience as they saw fit.


MOBY GAMES
1 2 3 4 5 6 7 8 9 0>erase Esc>cmds
File Edit Search Modes Misc Objects Check 736, -3400 Help

```
Selected LineDef (#52)          First SideDef (#65)
Flags:  1    --------Im         Normal texture:  STARTAN3        (No second SideDef)
Type:   0 -- Normal             Upper texture:   -
Sector tag:  0 (none)           Lower texture:   -
Vertexes:    (#9, #44)          Tex. X offset:   0
1st SideDef: #65   Length:      Tex. Y offset:   0
2nd SideDef: #-1    192         Sector:          #29
Editing LineDefs & SideDefs on E1M1   Free mem: 231360   Scale: 1/4  Grid: 0
```

Figure 41 - Lode Runner (Smith, 1983) & Doom (Romero et al., 2016) level editor, respectively

These level creators would grow side-by-side with games and appearing in different game genres, such as Shooters like Doom (Romero et al., 2016) featuring the first level editor in a 3D game. Some games even rose to fame because of their level editors like Trackmania (Castelnérac, 2003), a racing game that includes a racing track creator and online multiplayer where players can share their tracks and play ones created by other players.

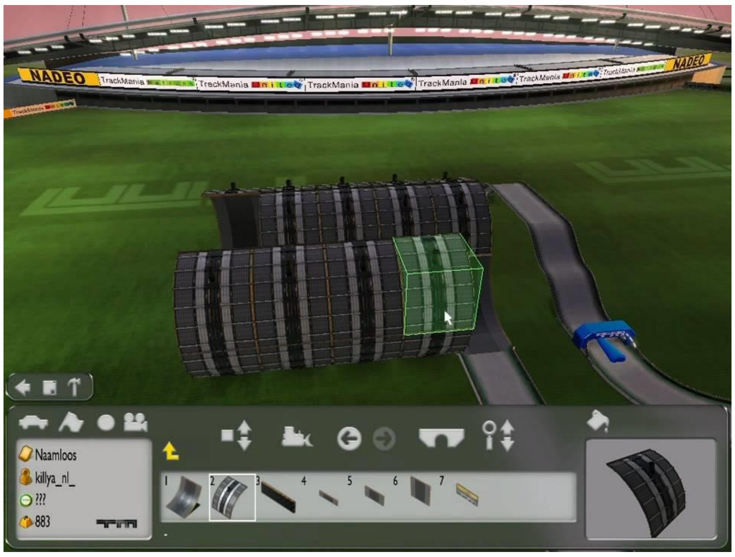


Figure 42 - Trackmania Nations Forever (Castelnérac, 2003) Track Editor

The growth in level editors popularity culminated in 2015, and later in 2019, with Super Mario Maker (Hino & Hosaka, 2015) and its sequel Super Mario Maker 2 (Kimura, Hino, Oshino, Tezuka, & Yamamura, 2019). These games main feature is the possibility to create Super Mario levels in the style of different Super Mario games, like Super Mario Bros. (Miyamoto & Tezuka, 1985) and Super Mario World (Miyamoto et al., 1990) and to share them online, letting other players play them. Both these games were commercial and critical hits, setting the bar for what players expected when using a level creator.

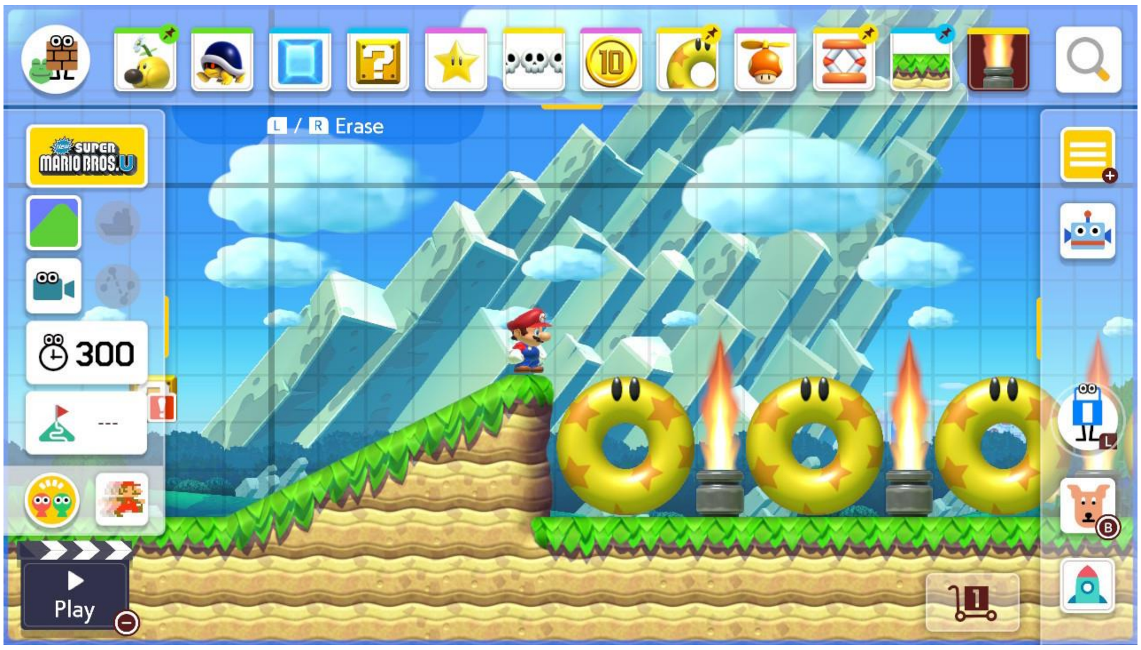


Figure 43 - Super Mario Maker 2 (Kimura et al., 2019)

6.2.2. Context & Artefact

In the Expression Stream Tinkerers, based on the "Creativity" motivation that is comprised of the Discovery and Design secondary motivations as described by Yee (2015), are moved by new systems comprised of unfamiliar ideas, realised in simulations and Tools to create craftworks – new levels and experiences.

In a sense, the Expression and Progression streams' users have **very similar motivations**, to **discover what is unfamiliar**, but as where Abstracters want to discover information to progress towards a goal, Tinkerers are fascinated by **novel** information and its **creative potential**. When an Abstracter finds a new mechanic, like a spring, their initial though might be "In what way can I use this to progress through the game" and look for places that can be access using the new jump height provided by the mechanic, but for the Tinkerer the objective is to use this new mechanic in creative ways to express themselves, like creating a level where players are moved by springs or creating a puzzle with the springs in mind.

Taking all the aforementioned games and tools into consideration the work on Cave Game's level editor was started, with the first task being the structure for the Block placement.

Firstly, the level was divided into a 32 by 18 grid. After this there was a need to create a visual aid for the player so they could visualize the grid in order to make the level creating experience a better one. A renderer was created that draws lines on the borders of each column and line, drawing a grid on the screen that outlines the block placement spaces.

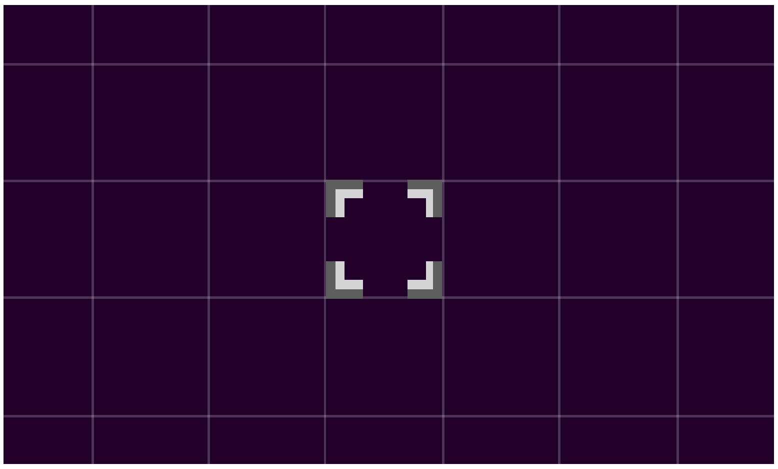


Figure 44 - Grid & cursor

After the grid was in place it was time to make the player cursor snap to each block space in order to give the player a greater control when making the levels. When in the level editor the game hides the players cursor and replaces it with a custom one with the size of a block (seen in the figure below) to signal which space the player is interacting with.

Having this structure in place it was time to create a simple UI to let the player use the different features of the level creator. Since most platforming levels tend to feature the gameplay near the bottom of the screen it made sense that the UI would be on the top portion of the screen.

This UI was, initially, comprised by only four buttons, corresponding to the basic four functions: **Drawing**, **Erasing**, **Clearing** and **Opening the Blocklist**; explained in the following paragraphs.

The **Drawing** function lets the player place blocks on the grid, in order to draw the level. When drawing over an already placed block the newer one remains, deleting the previous one. When in the Drawing function the cursor turns green to signal that the player is in Drawing mode.

The **Erasing** functions works much like the Drawing function, only deleting the blocks instead of placing them in the level. When in this mode the cursor turns red.

The third function, **Clearing**, opens a dialog box to confirm this action. If cancelled, the dialog closes and the editor returns to the last function, but if accepted, the level is cleared, deleting all the blocks on the screen.

The final function, the **Blocklist**, opens a window with all the available blocks, so the player can change the block used in the Drawing function. This selected block has a green outline as feedback to the player.

When using both these last two functions the cursor will enter a standby state, so the player does not place or delete blocks unintentionally. During this standby mode, the cursor will turn grey.

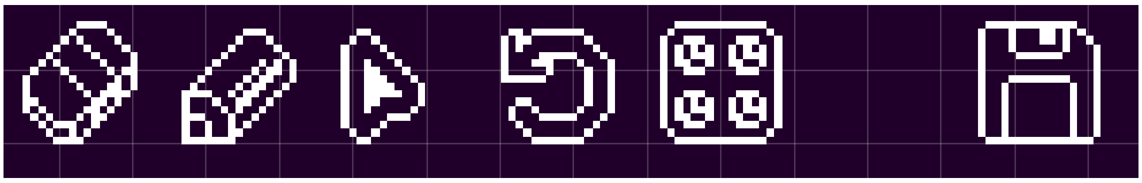
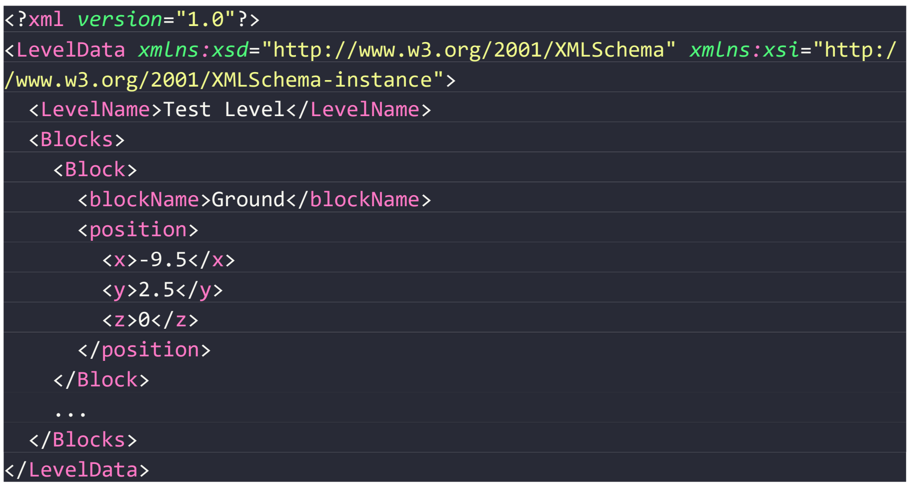


Figure 45 - UI Buttons

Being able to design a level is one of the most important features of a level creator, but it is for naught if one is not able to save said levels. This was one of the greatest technical challenges during the development.

After some research it was decided that the best format to save levels would be Extensible Markup Language, or XML for short, since it supported a simple yet logical structure to save the levels.

Each level is saved with the following structure:



```
<?xml version="1.0"?>
<LevelData xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http:/
/www.w3.org/2001/XMLSchema-instance">
  <LevelName>Test Level</LevelName>
  <Blocks>
    <Block>
      <blockName>Ground</blockName>
      <position>
        <x>-9.5</x>
        <y>2.5</y>
        <z>0</z>
      </position>
    </Block>
    ...
  </Blocks>
</LevelData>
```

Figure 46 - XML formatting

The "LevelName" tag defines the name of the level; the "Blocks" tag holds all of the blocks; and each "Block" tag defines what blocks appear in the level and in what positions. Each block has a name and XYZ coordinates to be placed, when loaded from the XML file.

When loading a level, the code goes through each block and places a block of the same name in the saved coordinates, recreating the level made by the player.

These levels can be shared in a simple way, foregoing the need to send the actual level file to other players. This will be better explained in following paragraphs but described in a simple way the player is able to generate a code that can be sent to other players.

With the ability to save levels, the need to create a place to view and interact with these levels appeared. A "Custom Levels" page was added in the main menu, where players can view and interact with the custom levels they created or that other players sent them.

One of the first features to be added was the ability to copy the code level to the clipboard, to be able to send the level to other players. This function converts the XML file to a Base64 code that can be sent to other players through text messaging.

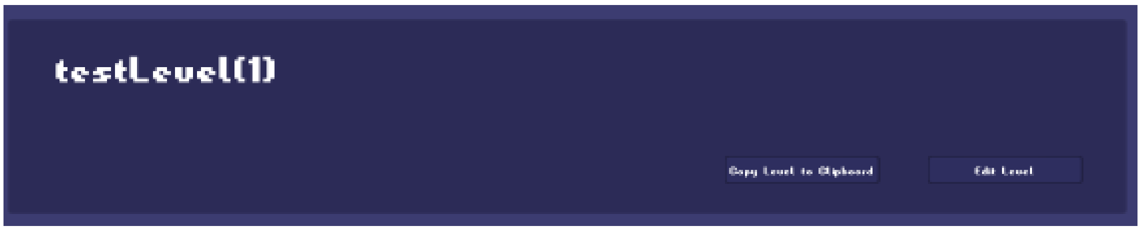


Figure 47 - Custom level list item

The player on the receiving end can then use the “Paste Level from Clipboard” to decode the Base64 string back to an XML file that is then added to the custom level folder and list. When pasting a level code, the game will verify if the level is valid and will also compare the level name to all other levels and will append a “(#)” suffix to the level name, where the “#” is the number of how many levels with the same name exist, if the level already exists.

Play and edit levels were also added to each level on the list, so the player is able to play and edit said levels. Every level is editable, including levels created by other players, to give players the ability to create levels cooperatively.

If the player has not yet created levels, or pasted a level from another player, a message will appear instead of the level list, encouraging the player to create a level.

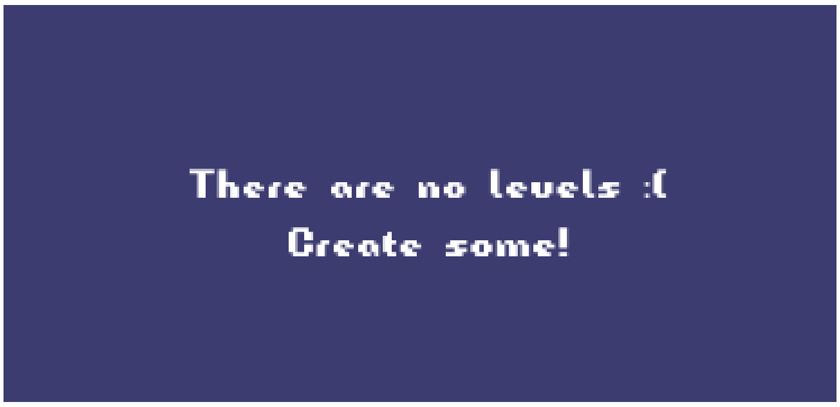

There are no levels :(
Create some!

Figure 48 - "No levels" message

Finally, it was time to create the remaining features for the level editor. The first of these features is the Playtesting mode for the editor. In this mode the player is able to play the level while editing and can change and tweak the level in a faster way. This way the player can press a single button, play the level, and then exit the play mode to make fast changes, instead of needing to save the level and then navigate to the level in the custom level in order to player, to then have to go back to the editor if changes are needed.

The next feature is paramount to the level creator, the warning system. This system is in place to make sure that the level structure is valid and that every important block exists. The system checks all the blocks placed and if it does not find a Spawn and/or Finish block the system issues a warning to the player.

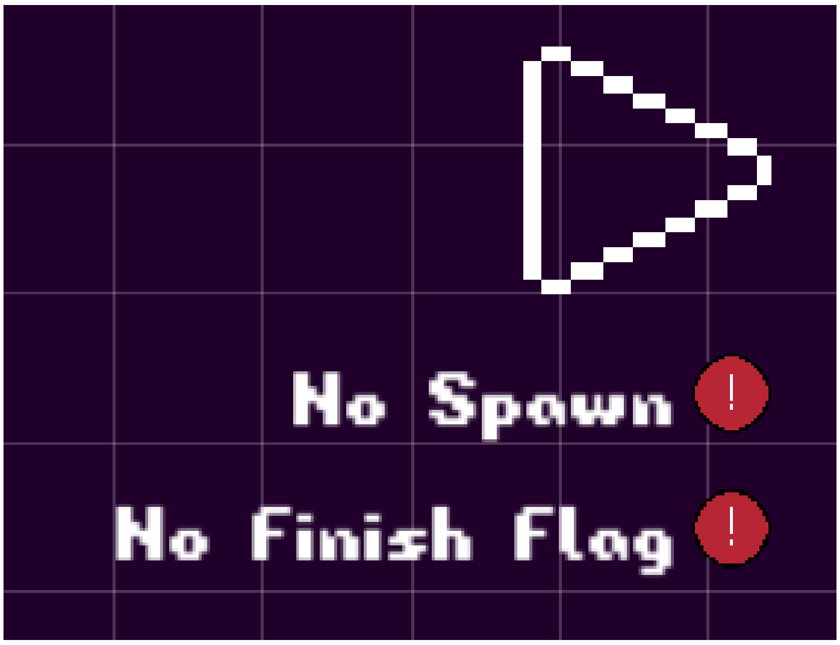


Figure 49 - Playtest button and Warning system

Lastly, and also related to the warning system, is the restriction of the structural blocks usage. When the warning system senses a Spawn and/or a Finish block it will send a signal to the Block Drawing module to restrict the usage of the detected block. This way each level will only have a single Spawn and Finish blocks, making it impossible for the level to have two or more of each of these blocks.

With these tools **Tinkerers** are given a **creative** outlet where they can explore **new** and **unfamiliar** tools and mechanics in a safe environment and **learn how these mechanics work through enactive play**, while using them to **create a new work**, a new level for others to play or build upon.

Having finished the work for the Expression Stream it was time to set eyes on the Relation stream of the Engagement Design Model. This stream leans heavily in human emotion and empathy and, thus, in a game's narrative and characters. The next chapter will discuss the Relation Stream and how the design and development for this stream went, although this would prove to be a challenge, due to the small scope of the playable demo, since there was little space and time to construct a solid narrative and to get the player to empathize with characters on such little time.

6.3. Relation Stream

This stream's users are Dramatists. Motivated by the relationships with other humans they have the desire to humanize everything that they are connected with, described in the Gamer Motivational Model (Yee, 2015) in the Immersion and Social motivations. Patterns rise from the interrelationship of individuals, allowing the particularisation of feelings, heightening empathy. Settings allow dramatists to compare themselves and one's mental models to others', indirectly through gossip and voyeurism. This streams design is based on the execution of conflicts that can give an analysis and experimentation of changes, giving the possibility of choice and decisions concerning relationships, following the practices of behavioural mimicry.

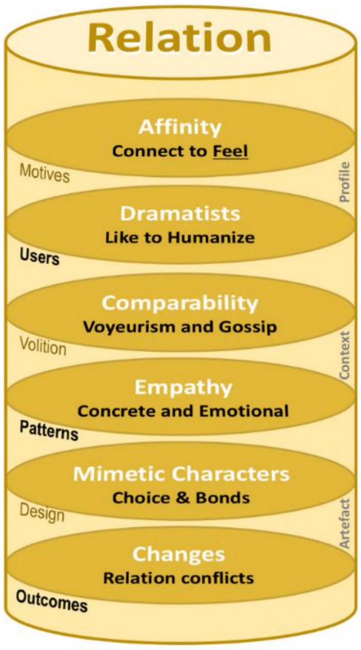


Figure 50 - Relation Stream

6.3.1. Profile

When asked about what makes a game good a lot of players will immediately point at the story. From games like Phoenix Wright: Ace Attorney (Takumi, Inaba, & Matsukawa, 2001) and Danganronpa: Trigger Happy Havoc (Marutani, Saito, & Terasawa, 2010), that can be considered visual novels, Firewatch (Rodkin, Vanaman, Anderson, Moss, & Remo, 2016) and Journey (Hunicke et al., 2012), telling their stories through the world and its characters and events, or The Stanley Parable (Wreden & Pugh, 2013), giving meta-commentary about the game itself, games are a great vehicle to deliver a story to someone.

Figure 51 - Danganronpa: Trigger Happy Havoc (Marutani et al., 2010) and Journey (Hunicke et al., 2012), respectively

More that movies and in books, where the player sees the story unfold as someone looking through a window, in games the player can participate in the stories, creating a greater bond with the characters and the world. The Relation Stream stands upon these relations and the sense of empathy with the world and the people that inhabit the world.

6.3.2. Context & Artifact

Dramatists look for familiar characters made of empathy and are moved by human conflict that results in changes. This made the decision to populate the world with characters with which the player can talk with, discovering more about the characters and the world itself.

As seen in screenshots in different chapters and in the game's name, Cave Game had its theming decided early on. The game's events would take place in caverns and would be heavily inspired by the locations. Everything from the game's graphics to the character design and their dialogues are centred around the world's theming.

Although the game's theme was decided early on the story was left on the backburner while the more structural parts of the game were created. Around the same time that the level design process started the story also started being developed, both these processes heavily influencing each other.

Cave Game's story can be explained in the following summary:

"You wake up in a cave system without any memories. As you continue to explore the caves you learn more about them and the creatures that inhabit this place. Through helping these cave's people, you get closer to knowing the truth about this space and about yourself".

In a first instance the intent was to have the story be told by the environment having little to no story being told to the player through cutscenes and having NPC's telling the story to the player, preferring a "Show, don't tell" approach to the storytelling, but after some discussion it became apparent that there would be a need for NPC's to fulfil the needs brought by the players belonging to the Relation Stream.

On a first stage the NPC's were supposed to only have a single dialogue, with no branching paths, but this was not ideal, as NPC's did not feel like real characters and were more akin to cardboard cut-outs, with simple speech lines. To fix this there was a need to create branching paths for the dialogues, to make the characters multi-dimensional and to give a sense of life to the world.

After some research, the story creation tool chosen was Yarn Spinner, the same tool used to create the dialogues in Night in the Woods (Holowka, Benson, & Hockenberry, 2017), a game renowned for its story telling and characters.

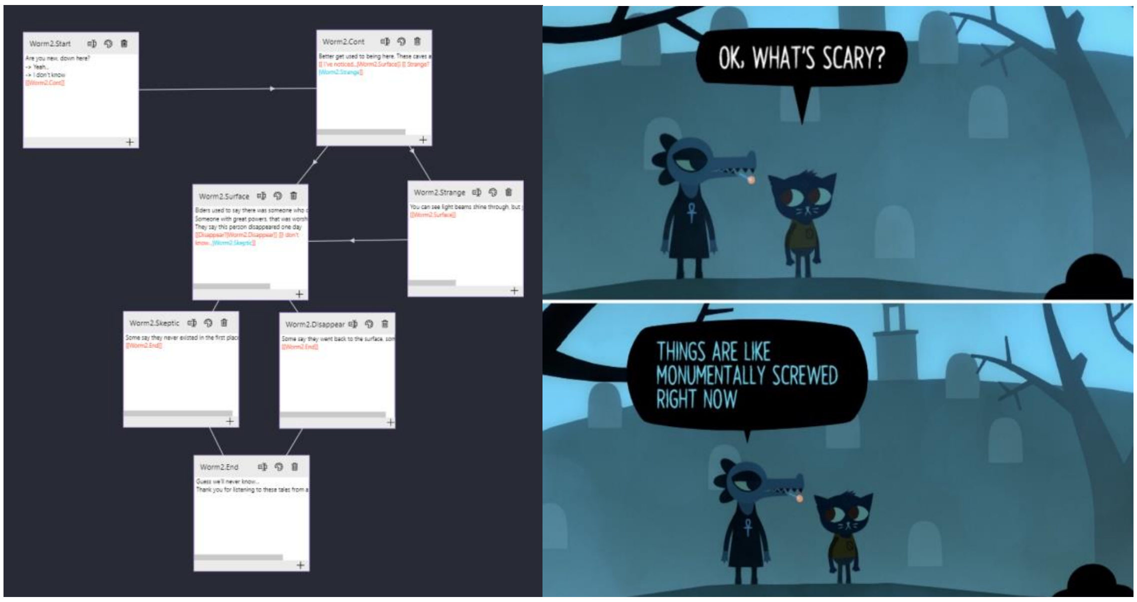

Worm2.Start
Are you new, down here?
-> Yeah..
-> I don't know
[[Worm2.Cont]]
Worm2.Cont
Better get used to being here. These caves a
[[I've noticed...|Worm2.Surface]] [[Strange?
|Worm2.Strange]]
Worm2.Surface
Elders used to say there was someone who c
Someone with great powers, that was worsh
They say this person disappeared one day
[[Disappear?|Worm2.Disappear]] [[I don't
know...|Worm2.Skeptic]]
Worm2.Strange
You can see light beams shine through, but
[[Worm2.Surface]]
Worm2.Skeptic
Some say they never existed in the first plac
[[Worm2.End]]
Worm2.Disappear
Some say they went back to the surface, som
[[Worm2.End]]
Worm2.End
Guess we'll never know...
Thank you for listening to these tales from a
OK, WHAT'S SCARY?
THINGS ARE LIKE MONUMENTALLY SCREWED RIGHT NOW

Figure 52 - Yarn Spinner dialogue and Night in the Woods (Holowka et al., 2017), respectively

Yarn Spinner was chosen, instead of other more-known tools like Twine, because of its simplicity of use and because of the streamlined process to implement the Yarn dialogues in the Unity Game Engine, as Yarn Spinner developers created a Unity package to make the process as quick and as painless as possible.

Each character started as a simple dialog flow, where there were no choices, to set the main information said by the character. After this, player character responses were added in order to help the player create a connection by talking with the NPC instead of being talked at – fulfilling the need for **socialization and comparability**. Finally, branching paths were added, to make the characters more real, adding depth and a reason to continue to talk to each character to discover new information – **mimicking a real person** and creating a more believable character. With the branching dialog paths it was possible to increase the reality of the characters, having them talk about themselves and still being able to talk about the world in which they exist, and even **gossip** about other characters, if inquired by the player. This way, it was possible to start creating connections with the non-playable characters while not having much time to deepen said relations.

These characters that populate the games world serve as a catalyst for change in the world and in the players themselves, as they learn more about the characters and the world that surround them and as they become more invested. Players' opinions on characters may change based on what other characters say about them or through the consequences of some actions taken by characters.

Due to the small scope of the game, it is very difficult to reach the outcome of the Relation Stream – Change – since this, more often than not, occurs after some time as passed since the creation of a relationship, when players have had time to create a better opinion on the story and characters and have become invested in them. Because this small demo is only the beginning of the story, players are still in an embryonic stage of their relations with the characters and there was not a full story arc to grant change. To counteract this, a "call-to-action" was added to the game in the form of a disembodied character that serves the purpose of "change-bringer" to the demo. With this character calling upon the players to help them and the world of the game, the change between the character being a simple bystander and having a more important role is put into motion and gives the player something to look forward to and to continue playing the game.

Having finished work for this stream, only smaller, although important, tasks remained for the completion of the game that will be discussed in the following chapter.

6.4. Other features

With all the development for the engagement streams done, other work was needed in order to make auxiliary systems to control the game logistics.

Cave Game needed a system to manage the level loading. This system saves the level between scenes and makes sure that the correct level loads when loading a custom level, since the game needs to send the level code between scenes to load the correct level in the editor.

Another very important system is the Scene Manager, giving the possibility to switch between scenes and lets the game load and unload scenes on top of other scenes, to make menus. This system serves as a manager to control the game flow and to change between menus and gameplay.

Up until now all the features discussed were part of the "feel" in the "look & feel" of the product. Having finished these features, it was time to make the game more aesthetically pleasing. When talking about videogames two major characteristics are always referred: the **gameplay** and the **graphics**.

Some people only praise a game's graphics, while other people say that the only thing that matters is the gameplay. Both these views are incomplete: a beautiful game that has no substantial gameplay will have no loyal players, since they will all move on after trying the bad gameplay; and a game that is not aesthetically pleasing will attract little to no players even if the gameplay is extremely refined. Aesthetically pleasing does not mean realistic 3D graphics with beautiful shadows or complex particle systems. Both 2D and 3D games can be aesthetical without having realistic graphics. 3D games like The Legend of Zelda: Wind Waker (Miyamoto, Aonuma, Koizumi, & Oyama, 2002), with its cell-shaded cartoon look, and Okami (Kamiya, 2006), bringing the Japanese Sumi-ê art style to life, prove that 3D games can be beautiful and acclaimed for their graphics without being photorealistic.

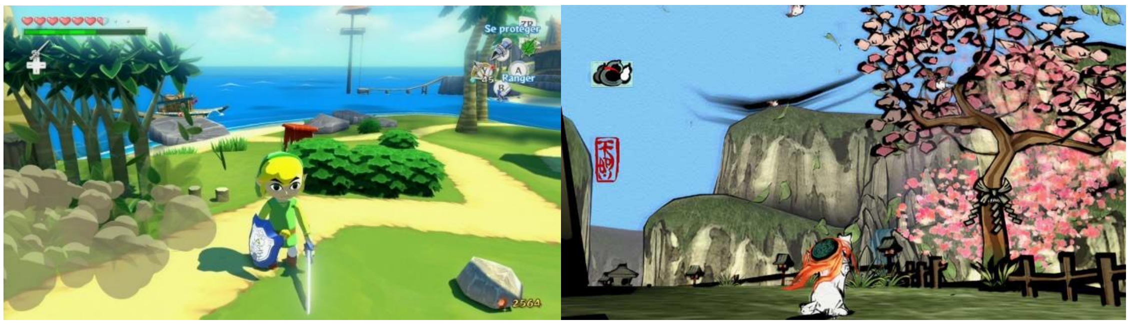


Figure 53 - The Legend of Zelda: Wind Waker(Miyamoto et al., 2002) and Okami(Kamiya, 2006) (left and right, respectively)

2D games can also be charming, like Guilty Gear Xrd REV 2 (Ishiwatari & Yamanaka, 2017), a fighting game that presents a cell-shaded look mixing the tradition 2D found in fighting games and 3D models to have more freedom with their camera movement, and the critically acclaimed Hollow Knight (Gibson, Pellen, & Kazi, 2017), a beautiful hand-drawn platforming game that uses its character and set design to create a cohesive and believable world.

Figure 54 - Guilty Gear Xrd REV 2(Ishiwatari & Yamanaka, 2017) and Hollow Knight (Gibson et al., 2017) (left and right, respectively)

With all this in mind the decision to ameliorate Cave Game's graphics was easy to make. Avery game needs to be pleasing to look at, even if it has great gameplay. A game is not a complete product without gameplay and graphics.

Due to the lack of graphical artists in the team, it was decided that Cave Game would mostly use free graphical assets found online. After scouring a lot of websites for assets that fit the overall theming and feel of the game two artist were found that offered free to use assets that matched the searched for feel. The first was Dmitry Mozgin, with the Space Cave Tileset (Mozgin, 2019); and the second one was LimeZu, with Mini Characters 3 (LimeZu, 2019). Space Cave Tileset was the Tileset used in the game since it was very close to the wanted look and ambience for the game world. LimeZu contributed with the character design. Both artists assets were slightly tweaked, with permission, in order to fit better in the game and with each other.

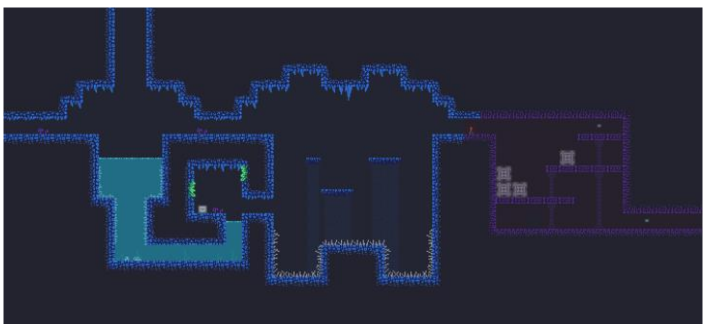
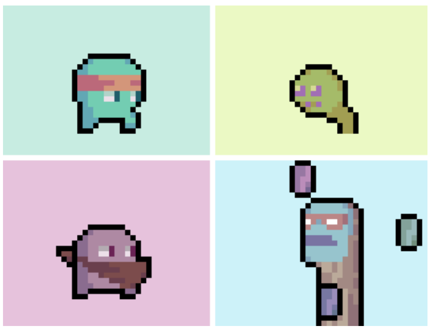



Figure 55 - Space Cave Tileset (Mozgin, 2019) and Mini Characters 3 (LimeZu, 2019) (left and right, respectively)

After applying these assets to the game its look became a lot more cohesive, but it still lacked a little something: ambience. Being a 2D game the use of lights was a bit limited, but Unity, the game engine used in the development, as of the 2020 version, now comes with 2D lighting capabilites built-in. These lights were paramount in setting the mood for the game, simulating cracks in the cave that let light rays through and giving the possiblity to use lights in the games puzzle.

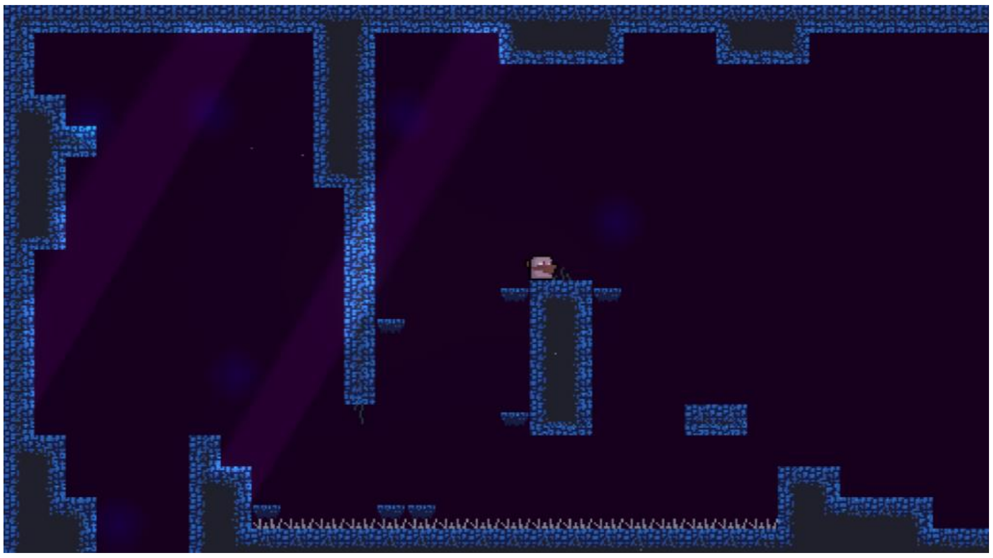


Figure 56 - Example of the lighting

Other details were added to the game to help set the mood, like plants growing on the floor and walls, and floating particles that appeared and disappeared on the screen, simulating dust in light. Post-processing was also added to the camera in order to make some colour corrections and to give Bloom and Motion Blur to the game's camera, to make the graphics more pleasing.

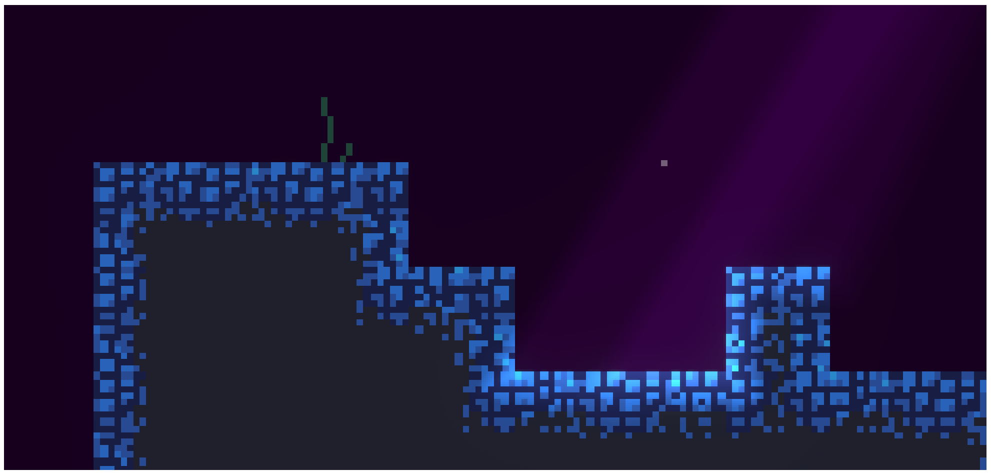


Figure 57 - Example of lighting, dust particles and plants

After all of this graphical work the game still lacked something to help tie up the experience into a single cohesive product. What was missing was sound: background music and sounds to help give weight to the actions. For the music, the attention was, again, directed to the internet, due to the lack of musical skills of the team. The BGM was provided by Ozzed, a musician that creates free 8-bit music for games. The track "Dunes at Night" (Ozzed, n.d.), from the homonymous album, was used as the background music for the game, creating an ominous and calm setting for the game. As for the rest of the sounds, like the jumping and the getting hurt sounds, were created using the Bfxr (Neuhaus, n.d.) free tool, that creates randomly generated sounds of the desired type.

The final task in the development was the creation of the menus and other pages besides the game itself, like the Credits, and connecting said pages to each other. This task, although small, is paramount to the completion of the product, since Cave Game has several different parts, due to its nature, it was vital that there was a simple way to navigate to each part of the game.

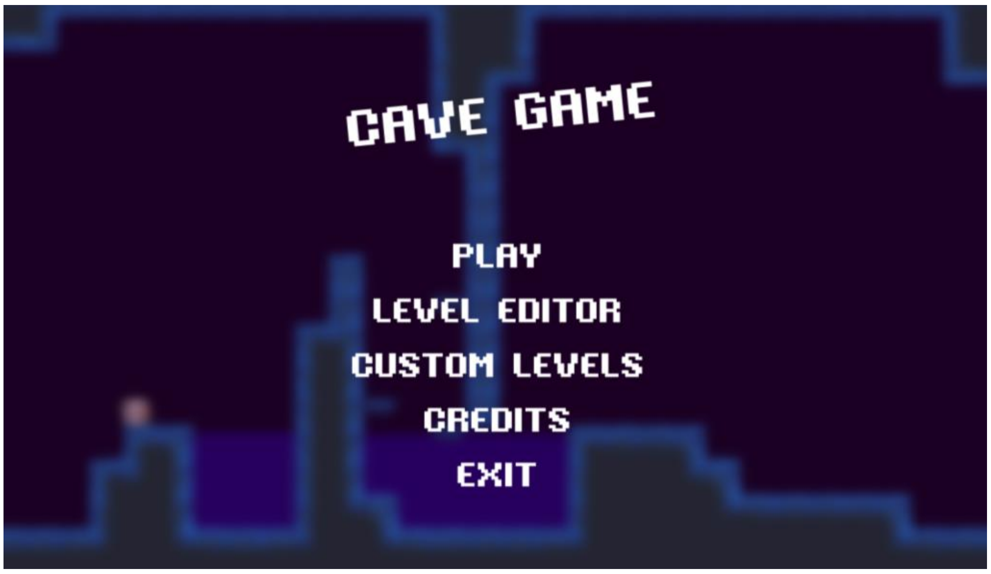


Figure 58 - Main Menu

Two menus were created: a main menu, in the beginning of the game that lets the player enter the game or go to custom levels or the level editor; and a floating menu that appears whenever the player presses the Escape key in the keyboard, that lets the player return to the Main Menu or exit the game.

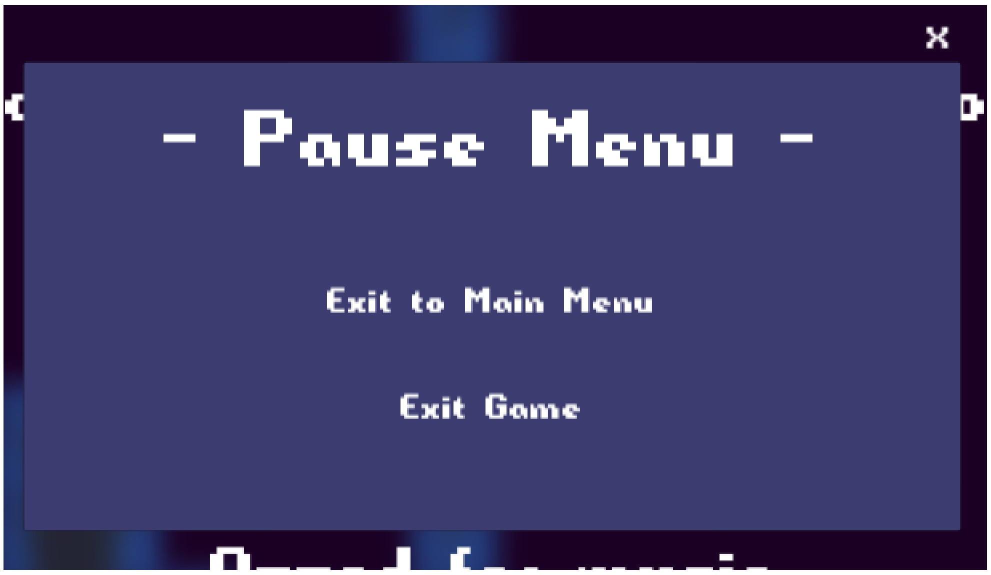


Figure 59 - Pause Menu

A simple credits page was also created to give credit to the artists and musician that provided art assets and the music for the game.

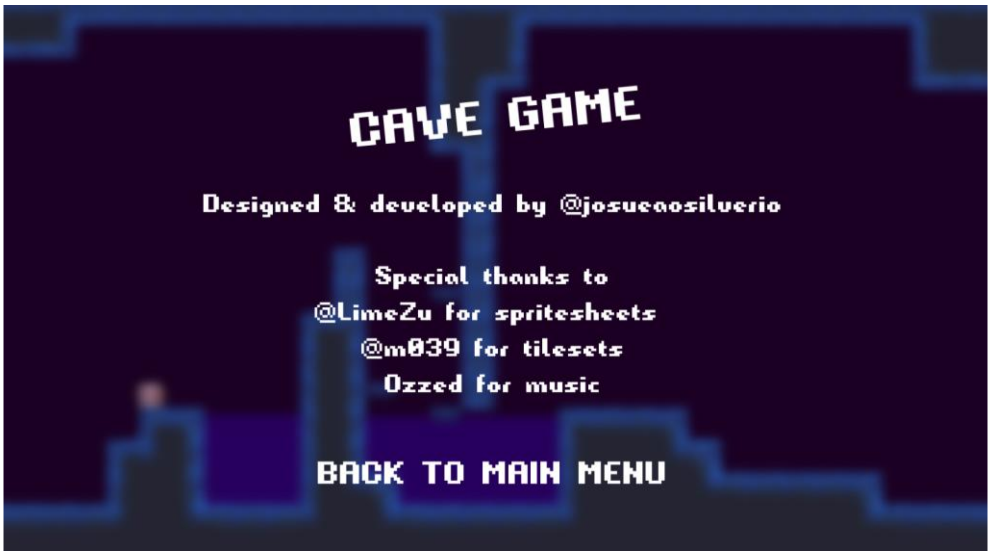


Figure 60 - Credits page

All of these small, but important, tasks mark the end of the design and development process and the end of the development documentation chapter.

Due to the nature of interactive multimedia projects, it is often difficult to understand what these products are with just images and descriptions. With this in mind a small video was created to help the reader understand how some of the features discussed above were implemented (See Appendix **A**).

In the next chapter we will have a critical analysis of the work done and we will discuss problems found during the design and development process and in what way the ED Model influenced the decisions made during the design and development process and bring into light the lessons learned during the design and development process when using the Engagement Design Model, in comparison to a regular design and developmental process without the model.

7. Critical Analysis

In this chapter we will analyse the documentation of the design and development process, described in the previous chapter, with the intention of understanding what went right and what went wrong, bringing into attention what lessons were learned during the design and development process when using the Engagement Design Model when compared to a regular independent game design and development process.

The choice, described in the Methodology chapter, to use the Kanban method to organize and document the work in precision and in a realistic manner aided in the process of distilling all of the reports in the above chapter into simple and understandable lessons, described in the chapter below, and gave us a view of how well the project went, when comparing the state of all features planned and completed.

One of the advantages brought to the process by the ED Model was the possibility to better organize the Kanban boards, described in the Methodology chapter. Using the ED Model made it possible for the boards to be split in three different ones, for each of the streams, instead of having a single board that holds all of the tasks. This meant that a "**divide and conquer**" approach could be taken on the features and helped maintain a single mindset for work for some time instead of having to change the mindset for design and development based on the task at hand, since each stream has its own mindset.

7.1. Progression Stream

As a broader analysis we can state that the design and development done focused on the Progression Stream of the ED Model went the best of the three streams. This was probably due to the fact that most of the design and development done on the Progression Stream serves as a base for the other streams to stand upon, especially when talking about more technical games like platformers. Another factor that contributed to the good outcomes was the fact that this streams' engagement is not very dependent on the scope of the game, unlike the other streams that work better on a project with a bigger scale.

7.1.1. Profile

As stated in previous chapters, **Abstracters are driven by Mastery**. As such, development for the Profile level was based around the technical mastery, which translates to the character movement and the camera movement. Overall, the design and development for the character movement happened as expected. Most of the features planned initially were implemented and the few not implemented mainly happened due to them being deemed not important enough for the game's current state. Features like the main character movement, and quality of life features like **jump corner correction** and **landing correction** that exist to make the player's experience more satisfying were implemented without problems. Other planned features like a **dash**, that made the player move faster in a single direction for a short burst, and another quality-of-life feature, **coyote time**, giving the player some time to jump after falling of a platform, were not implemented due to the changes in the layout, since the final layout ended up being more horizontal and these features work better on a vertical layout.

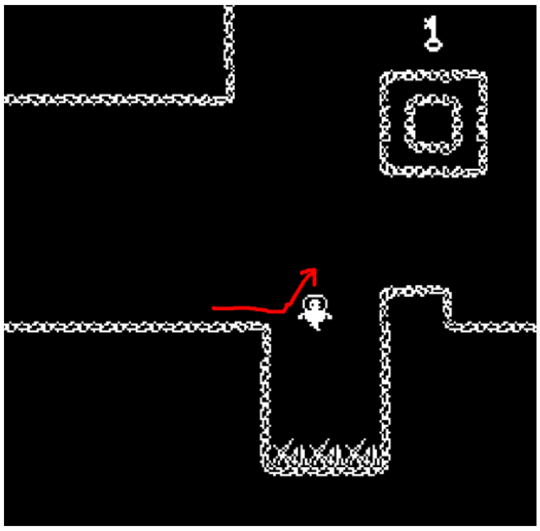


Figure 61 - Coyote Time example

Through the use of the ED Model, it was easy to understand what the players that fit in the Progression Stream tend to like and it was possible to speed up the design and development process. One lesson we can take from this process is: **Knowing your audience and their drive can greatly increase the speed of the design process**.

7.1.2. Context & Artefact

With the information that Users are motivated by gaps in information that can be problem solved through the use of artefacts like puzzles and obstacles, leading the player to a goal, we can analyse the design of the Context and Artefacts done for the Progression Stream of engagement, with a main focus on the level design and layout and the creation of the puzzles present in these levels either as a standalone experience or as puzzles embedded in the levels' layout itself.

We can say that most of the development made in this regard went relatively easily when compared to the design and implementation of other features, mainly on other streams. The design and creation of the level and the different types of blocks used to create the world was not a challenge due to the help given by the ED Model in the design process, and the creation puzzles and water sections of the map was also much easier when comparing to a regular game design and development process without the use of tools like the Engagement Design Model. Due to the choices in layout and the size of rooms, puzzles were a bit simpler than originally imagined, but this did not prove to be a problem, as a good balance between puzzle and action rooms was achieved.

The use of the Engagement Design Model made it possible **to focus on creating new and exciting experiences** for Abstracters instead of having to firstly think about the player base and having to focus time and resources on **discovering what makes more technical players tick** – information already present on the ED Model.

7.2. Expression Stream

Being harder than the Progression Stream to implement, but still easier than the Relation Stream, the Expression Stream proved to be somewhat of a challenge but still doable within the scope of the project and, with help from the development from the Progression Stream, became easier to design to, when seeing what guidelines the ED Model gives us.

7.2.1. Profile

The Engagement Design Model tells us that **Tinkerers are driven by creativity** – the need to create and experiment. The profile of these users, along with the artefacts created by the Progression Stream made it easier to define the overall experience planned for this stream in Cave Game.

After designing and developing a platformer, and knowing the Tinkerers need to create, the decision to create a Level Editor was quickly cemented, as cases for major success in the creation of level editors for platformers are easily found in the industry, being them triple-A like Super Mario Maker (Hino & Hosaka, 2015) or indie like Levelhead (Coster, 2019).

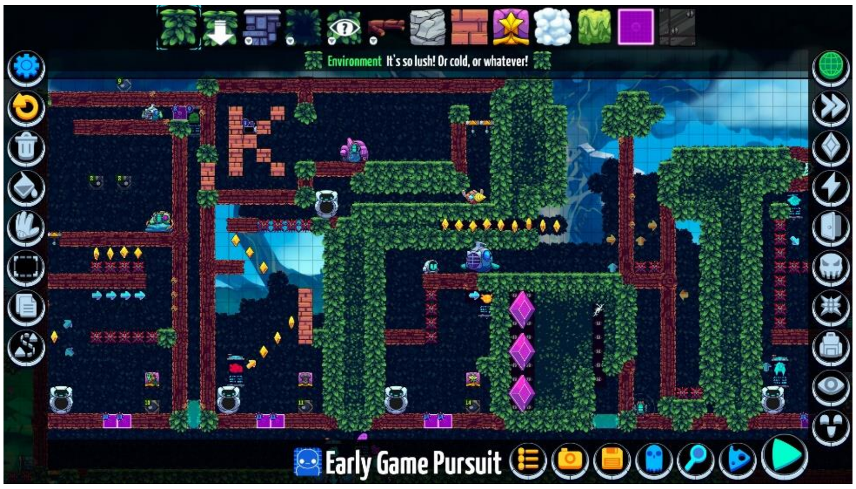


Figure 62 – Levelhead (Coster, 2019)

7.2.2. Context & Artefact

As said above, it only made sense to create a level editor for Cave Game, as that would be the best way to incorporate creative tools in a genre that is, in its essence, so simple and straightforward.

Knowing this, almost all of the features created in the game were designed to be usable by the player in the level editor, creating the feel that the levels players create might have been in the game in the first place. Unfortunately, to keep the player from completely breaking the game and because of the limited size for player created levels, one of the more interesting features, the water bodies, were not implemented in the list of mechanics available in the level editor. After some thought it was decided that letting the players use this mechanic, although having creative potential, was not a good idea, since players could break the systems very easily and the development investment in order to create a potentially game-breaking feature was too great to sustain.

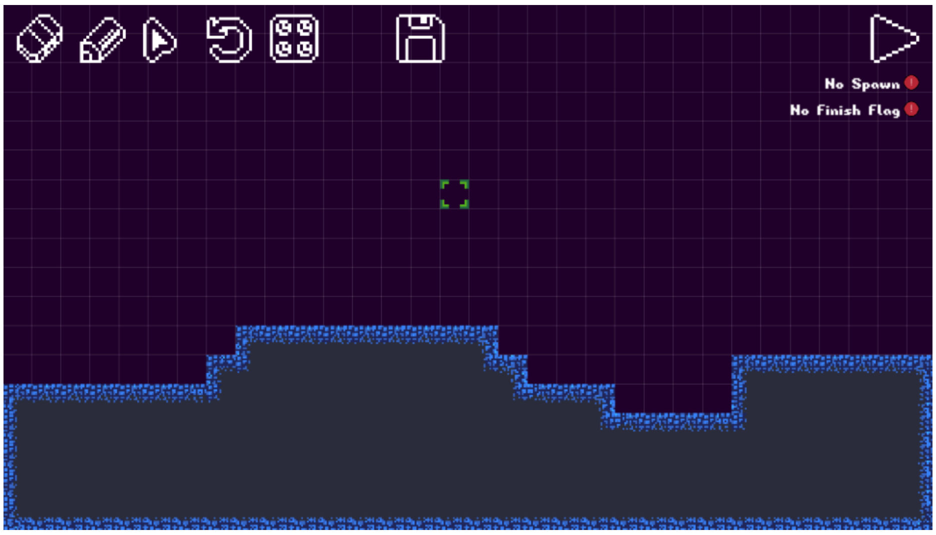


Figure 63 - Level Editor

Even so, and although not having previous experience in designing creative tools in videogames, it can be said that the guidelines present in the Engagement Design Model made it very easy to design and realize the Artefacts defined in the ED Model, the creative tools to make craftworks.

A major lesson that can be taken from the design and development of this streams artefacts is that: **When creating a videogame that employs both Progression and Expression engagement types, the Artefacts created for the Progression Stream can heavily influence the Context of the Expression Stream.**

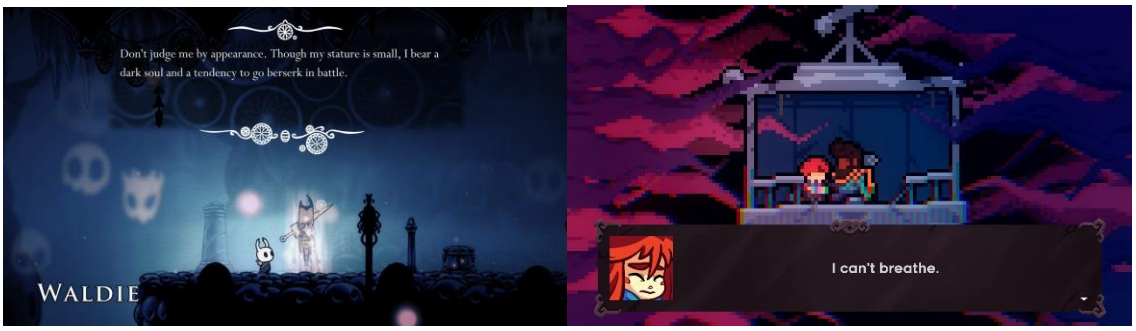
7.3. Relation Stream

As a broader analysis we can say that the Relation Stream had, at the same time, the hardest process to design, but the easiest to implement, in a technical sense. More than the Progression and Expression, the Relation Stream depends heavily on the scope of the project, due to the nature of this stream. The Expression Stream stems from human emotion and, more often than not, human emotion acts like a process and not like an action, per say. Most of the emotions we feel, and especially the feeling of connection that fuels the Relation Stream, take time to mature and very rarely do we feel a connection with someone or some character in little to no time.

This stream proved to be the most difficult to design, due to the small scope of the game and the nature of this stream. The Relation Stream leans heavily on human emotion and empathy, things that need time and depth to develop, especially in virtual environments where there is a there is the need for a suspension of disbelief for players to create relations with the universe's characters. The small scope that comes with a playable demo makes this task much more difficult as the whole of the demo only plants the seeds for this empathy, that would have time to grow on a longer game but cannot grow in this small game, thus making this experience a bit incomplete. Still, the implementation, in a technical sense, was one of the easiest to implement, due to the abundance of tools to aid in the development of characters and stories, stemming from a large user base.

7.3.1. Profile

As stated above, Dramatists, this streams users, are moved by the need of affinity – the need to connect with others. Due to the nature of Cave Game, a more technical experience as platformers tend to be, the design of a story with believable characters proved to be difficult, as most platformers tend to have an experience more based around gameplay, as seen on games like Super Mario Bros.. Only more recently, with games like Celeste (Thorson & Berry, 2018) and Hollow Knight (Gibson et al., 2017), do we see platformers and metroidvania with compelling story and characters we can connect to.


Don't judge me by appearance. Though my stature is small, I bear a dark soul and a tendency to go berserk in battle.

I can't breathe.

Figure 64 - Hollow Knight (Gibson et al., 2017) *& Celeste* (Thorson & Berry, 2018)

7.3.2. Context & Artefact

The design and development of mimetic characters was at the same time a simple yet hard task. In the beginning of development, the idea for characters was much simpler, without branching paths and with different ideas in mind. This led to the development of an in-editor tool to create dialogs which ended up being scrapped and substituted by Yarn Spinner, the tool used to create and manage dialogs in the game.

This proved to be the best choice and let the game have much deeper characters and dialogs with a fraction of the work when in comparison to the old system. This came from a bad understanding of the Relation Stream and resulted in a month's work ultimately being scrapped – problem that would have not happened if more time were taken when analysing the guidelines present in the Engagement Design Model. Although unfortunate, this oversight gives us insight in what ways the development without the proper use of the Engagement Design Model leads us.

After resolving this problem, the design and development went on without greater problems besides the smaller scope of the game, as referred in the previous chapter, which made it difficult to create empathic characters.

It is also important to note that, when designing and developing for the Relation Stream there was a fine balance between too much action brought by the Progression Stream and between too much talking, and exposition brought by the Relation Stream. Although this problem was brought to light by the ED Model, since when in comparison to a regular design and development process we may not divide the work into these three streams, it may happen in every development process, and the use of the ED Model brought awareness to this problem early on and prevented the game from becoming too one-sided, in regards of the Streams.

After this analysis it was possible to summarize the knowledge created in three lessons, found below:

a) If not careful, it is easy to overshadow the engagement created by the Progression Stream with the one by the Relation Stream; and vice-versa.
b) Engagement based on the Relation Stream, while possible in smaller projects, will work much better on a greater scope, where connections have space to deepen.

c) A better understanding of the Engagement Design Model in the beginning can save a lot of time in the long run.

7.4. Problems & Solutions

Every development project has issues and problems that can occur due to various factors, previously expected or not. In this chapter we will discuss the different problems, technical or not, that arose during the design and development project and how these problems where overcome.

Firstly, we will discuss problems that came from the size and requirements of the projects when in comparison to the size and skills of the team.

Every game needs, to different degrees, a developer, a designer, an artist, a musician, and many other different positions in the team to evolve from an idea to a fully-fledged product. Naturally, on indie teams most members must put on different hats and fill different jobs due to the small nature of these teams. In some cases, a single person must wear every hat and must have radically different skills, like programming, creating art and composing music, which, although difficult, is completely possible with enough time.

Cave Game's team was composed by a single member, with skills mainly focused on programming and game design, and severely lacking in art and music creation skills. Since, due to the project's nature, time is limited and, even when having these skills, the project's magnitude is relatively grand when considering the available time and due to these problems there was no possibility to create most of the art and music for the game, nor to learn these skills in in a timely manner.

To solve the problem above there was a need to use free to use assets found online, credited in the previous chapter. Every asset was hand-picked to fit the game's team and to make sure that the usage of the assets was completely permitted for all kinds of uses. Some art was then changed to better fit the theming and styles and a bit com simpler complementary art was created to make a better experience.

As for the music and sounds, every song in the game was from a free game music library and chosen to fit the themes and every sound, like the jumping sound or the sound for getting hurt, was created using a free software that randomly generates game sounds for different purposes, like the aforementioned ones.

The next problems to be discussed will be technical problems that arose from the complexity of the task. When designing and developing a game that must have all types of engagements described in the ED model one must be very careful when designing ideas to make sure of the feasibility of each feature. Gunpoint's (Francis, 2013) developer, Tom Francis, in his GDC talk "Lessons Learned Making Gunpoint Quickly Without Going Mad" (Francis, 2016) gives a formula he uses when deciding what features. Francis makes his decision based on efficiency and defines it as the relation between the value the features has for a player and the work the developer. With this formula the developer could gauge if the features planned are viable for development without spending time to developing the feature only to scrap it after.

Firstly, some of the features planned had to be simplified. One of these features was the movement of the character, that initially has dashes and wall grabbing/jumping planned, but due to the complexity of these tasks the movement ended up being simplified to the most important parts, and the levels were changed to fit these changes. Another feature that ended up being simplified was the level creator, that has planned more blocks for the player to use and features like letting the player change some blocks settings, like the force of the Spring Block.

Some other features were not deemed important or efficient enough, so they wound up being scrapped entirely. Some of these features were a Speedrun-like clock that counted the time the player spent in the levels and the ability to save levels online with the Level Editor, so players could share and play each other’s levels without the need to share codes through means outside the game. Both these features were ultimately scrapped because the value they would bring would not cover the cost the demanded, both in time and in other means like the cost of maintaining a server for level hosting.

8. Conclusion

As a conclusion we can analyse the project goals stated in a previous chapter and see if and how these goals were achieved:

a) Designing a videogame with three different engagement types, mirroring the three different streams present in the Engagement Design Model.

Overall, we can say that this objective was achieved, having created three different facets to the game with all three different engagement types, to different degrees, described in the Engagement Design Model.

b) Documenting the design and development process, connecting the decisions with the Engagement Design Model.

As seen in previous chapters we can say that the documentation of the process was well achieved, describing all the technical work in the development as well as the design choices while justifying them using the levels of the streams in the Engagement Design Model.

c) Determine what lessons can be taken for indie developers so they can open their game up for more people, without having to compromise their creative vision, based on the documentation of the process.

In the previous chapter we discussed what was learned during the design and development process while using the ED Model when in comparison to a design and development process without using a pre-existing design system, condensing the knowledge discovered in lessons that we will state below:

1. Knowing your audience and their drive can greatly increase the speed of the design process.
2. When creating a videogame that employs both Progression and Expression engagement types, the Artefacts created for the Progression Stream can heavily influence the Context of the Expression Stream.
3. If not careful, it is easy to overshadow the engagement created by the Progression Stream with the one by the Relation Stream; and vice-versa.
4. Engagement based on the Relation Stream, while possible in smaller projects, will work much better on a greater scope, where connections have space to deepen.
5. A better understanding of the Engagement Design Model in the beginning can save a lot of time in the long run.

The completion of these goals gives us an answer to our research question:

What lessons can we take from the Engagement Design Model when designing the same experiential universe for different types of players?

The design, development and documentation of the work process gave us insight into all of the choices made with the Engagement Design Model in mind and made it possible for us to determine and create lessons to help guide the design process when using the ED Model, so that other designers and developers have a better experience and save time during the process, and possibly creating new lessons based on the ones found during this project.

As a closing statement we can say that this process was very interesting and made it possible to learn about new subjects pertaining to Game Design and Development. Although very difficult we can say that, besides the lessons stated above, a lot of technical abilities were studied, and all of the process let the author evolve and grow as a game design and development professional.

In the next chapter we will discuss alternative routes that would be interesting to explore in future projects, giving reason as to why these studies would be valuable.

8.1. Future work

Every project of this nature, with limited resources and a tight schedule, must be simplified and will, more often than not, be limited to a specific topic to make sure that the project is finished in the allotted time. That being said, every project will have some paths that would be interesting to explore in a future work. In this chapter we will discuss other ways this project could have gone and why they would be important to research in a future work.

Firstly, it would be very interesting to see the outcomes of the research in a very similar project but with different game genres and with different interpretations of what could be designed and developed for each of the Engagement Streams. This way we could see how the Engagement Design Model could influence the design and development process in games like racing games, MMORPGs or fighting games, and see if different game genres might give us different lessons.

Also, still within the indie scope, it would be very interesting to apply the Engagement Design Model in the design and development of a game with a larger, but still limited, team. The lessons taken from this project were based on a ‘team’ comprised of a single person, and would apply to a very small team, but it would be very interesting to see in what way a bigger team might influence the lessons taken from the process and even what new lessons specific to a team environment might be learned in this variation of the process.

Another project that would be very interesting to see the outcomes of, but very unlikely to happen, would be the usage of the ED Model in a triple-A game design and development process. The design and development process of a game with a giant team and much more resources than in indie team is radically different from the indie game development process, and the usage of a system like the Engagement Design Model might have a noteworthy impact and might give us completely different lessons.

Finally, another type of game that would be very important to research is the Serious Game. Abt (1970) defines serious games as games that “have an explicit and carefully though-out educational purpose and are not intended to be played primarily for amusement.”

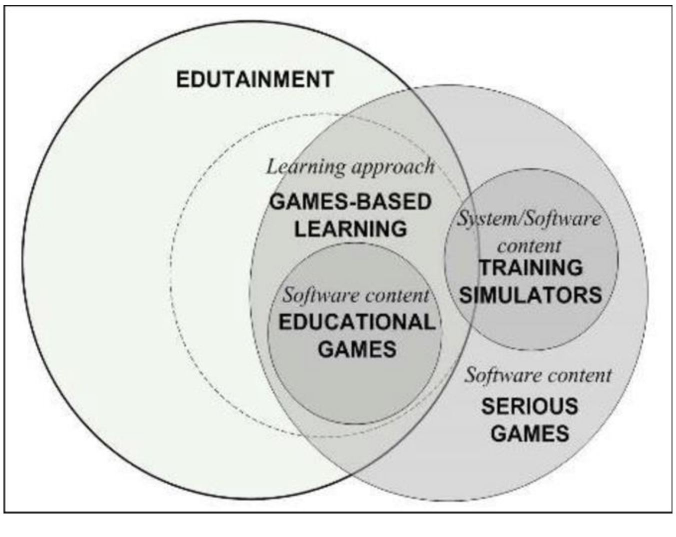


Figure 65 - Serious Games (Tang, Hanneghan, & El Rhalibi, 2009)

As said in previous chapters, the focus of this project was to see what lessons might be learned in the development of a game using the Engagement Design Model and implementing all three engagement streams present in the model. This design and development of all three of the streams means that, in theory, the game would be engaging for most players and no type of game need this more than Serious Games. These games are made for a lot of different areas, like medicine and advertisement, but are more used in education and training – setting where games should be engaging to all students and trainee.

The use of the Engagement Design Model in the design and development of Serious Games might have an important impact in this type of games, paving the way for a future where schools and companies are able to reach all of their students and employees and give them information through the use of videogames, reforming the way we see learning, transforming into something we do with because of intrinsic motivation and not because of obligation. We can see this improvement in education in cases like Variant (Texas A&M University, n.d.), a game that fuses mathematics and calculus with gameplay, used in the Texas A&M University, where they found out that students tended to play around 10 time a week for about two hours, which resulted in a success rate for students to rise from 80% to 100% (Thomas, 2018).

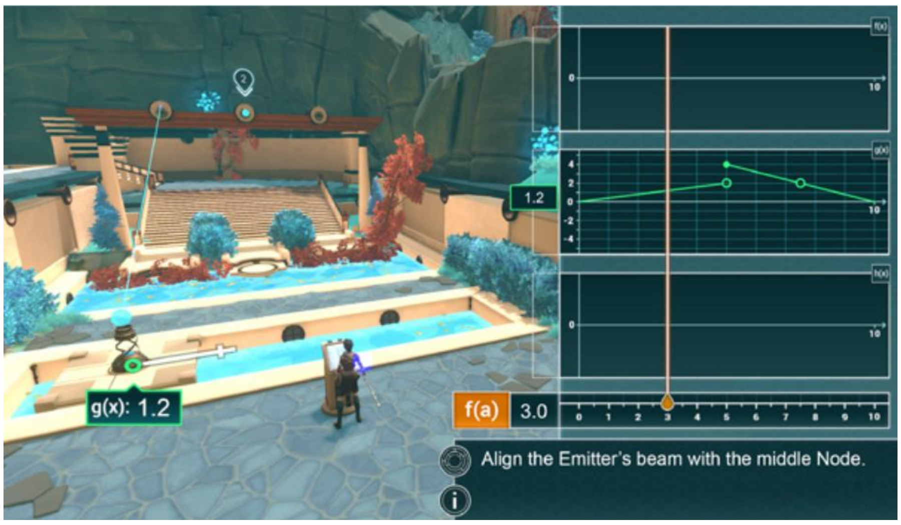


Figure 66 – Variant (Texas A&M University, n.d.)

References

Abt, C. C. (1970). Serious Games. *American Behavioral Scientist*, *14*(1), 129–129. https://doi.org/10.1177/000276427001400113

Amaya, D. (2004). *Cave Story*. Studio Pixel.

Anderson, D. J. (2010). *Kanban: Successful Evolutionary Change for Your Technology Business*. Sequim, WA: Blue Whole Press.

Apperley, T. H. (2006). Genre and game studies: Toward a critical approach to video game genres. *Simulation and Gaming*. https://doi.org/10.1177/1046878105282278

Armstrong, M. (2009). *Borderlands*. USA: 2K Games.

Bartle, R. (1996). HEARTS, CLUBS, DIAMONDS, SPADES: PLAYERS WHO SUIT MUDS. *Journal of MUD Research*.

Bartle, R. (2004). Designing Virtual Worlds. *New Riders Publishing In*. https://doi.org/10.1093/carcin/bgs054

Baszucki, D., & Cassel, E. (2006). *Roblox*. Roblox Corporation.

Bateman, C. (n.d.-a). BrainHex: What does my BrainHex class icon mean? Retrieved December 29, 2019, from https://blog.brainhex.com/what-does-my-brainhex-icon-mean.html

Bateman, C. (n.d.-b). BrainHex. Retrieved December 29, 2019, from https://blog.brainhex.com/

Bateman, C. (n.d.-c). International Hobo - DGD2 Questionnaire - Intro. Retrieved December 29, 2019, from http://survey.ihobo.com/DGD2/intro.php

Bateman, C., & Boon, R. (2005). *21st Century Game Design (Game Development Series)*. USA: Charles River Media, Inc.

Bateman, C., & Boon, R. (2006). 21st Century Game Design. In *21st Century Game Design*.

Bélanger, D. (2014). *Watch Dogs*. Ubisoft.

Berbece, N. (2016). *Move or Die*. Romania: Those Awesome Guys.

Berger, B. M., Caillois, R., & Barash, M. (1963). Man, Play, and Games. *American Sociological Review*. https://doi.org/10.2307/2090095

Bithell, M. (2012). *Thomas Was Alone*.

Blow, J. (2008). *Braid*. Number None.

Boon, E., & Tobias, J. (1992). *Mortal Kombat*. Midway Games.

Bottomley, P. (2014). *Ether One*. White Paper Games.

Braben, D., & Bell, I. (1984). *Elite*.

Brevik, D., Schaefer, E., Schaefer, M., Sexton, E., & Williams, K. (1997). *Diablo*. Blizzard Entertainment.

Castelnérac, F. (2003). *Trackmania*. Nadeo.

Cavanagh, T., & Foddy, B. (2010). *VVVVVV*. Nicalis.

Cave Game Example - YouTube. (n.d.). Retrieved December 22, 2020, from https://www.youtube.com/watch?v=d9HPQ6MGsjY

Cazzaro, M., & Débant, A. (2014). *Child of Light*. Ubisoft.

Christensen, C. M. (2012). Encyclopedia of Human-Computer Interaction. In *Encyclopedia of Human-Computer Interaction*. https://doi.org/10.5121/ijfcst.2014.4403

Coster, S. (2019). *Levelhead*. Butterscotch Shenanigans.

Costikyan, G. (2006). I Have No Words & I Must Design (1994). In *The game design reader : a rules of play anthology*.

Craig, J., Mercer, S., & Elliott, M. (2016). *Overwatch*. Blizzard Entertainment.

Crawford, C., Peabody, S., Art, T., Web, W. W., & Loper, D. (2003). The Art of Computer Game Design. In *Computer*.

Crooks, D. (2016). *Enter the Gungeon*. Devolver Digital.

Delay, C. (2005). *Darwinia*. Introversion Software.

Désilets, P., Raymond, J., & May, C. (2007). *Assassin's Creed*. Ubisoft.

Dixon, D. (2012). *Universe Sandbox*. Giant Army.

EA Romania. (2019). *Fifa 20*. EA Sports.

Fish, P. (2012). *Fez*. Trapdoor.

Flury, M., & Gibson, B. (2016). *Thumper*.

Foddy, B. (2014). State of the Union. Retrieved January 13, 2020, from https://www.youtube.com/watch?v=7XfCT3jhEC0

Fox, T. (2015). *Undertale*.

Gaynor, S. (2013). *Gone Home*. The Fullbright Company.

Gibson, A., Pellen, W., & Kazi, D. (2017). *Hollow Knight*. Team Cherry.

Greenberg, A. C., & Woodhead, R. (1981). *Wizardry*.

Gulliksen Stray, V., Moe, N. B., & Dingsøyr, T. (2011). Challenges to teamwork: A multiple case study of two agile teams. *Lecture Notes in Business Information Processing*. https://doi.org/10.1007/978-3-642-20677-1_11

Healey, M. (2005). *Ragdoll Kung Fu*. Merscom.

Hidemaro, F., & Aonuma, E. (2017). *The Legend of Zelda: Breath of the Wild*. Japan: Nintendo.

Hino, S., & Hosaka, A. (2015). *Super Mario Maker*. Nintendo.

Hoffman, M. (2012, August 14). Make a Splash With Dynamic 2D Water Effects. Retrieved September 17, 2020, from https://gamedevelopment.tutsplus.com/tutorials/make-a-splash-with-dynamic-2d-water-effects--gamedev-236

Holowka, A., Benson, S., & Hockenberry, B. (2017). *Night in the Woods*. Finji.

Hooke, R. (1660). Hooke’s law | Description & Equation | Britannica. Retrieved September 17, 2020, from https://www.britannica.com/science/Hookes-law

Hunicke, R., Clark, N., Singh, B., & Bell, C. (2012). *Journey*. Sony Computer Entertainment.

Iisalo, J. (2009). *Angry Birds*. Rovio Entertainment.

Ikonen, M., Pirinen, E., Fagerholm, F., Kettunen, P., & Abrahamsson, P. (2011). On the impact of Kanban on software project work: An empirical case study investigation. *Proceedings - 2011 16th IEEE International Conference on Engineering of Complex Computer Systems, ICECCS 2011*. https://doi.org/10.1109/ICECCS.2011.37

Ishiwatari, D., & Yamanaka, T. (2017). *Guilty Gear Xrd REV 2*. Arc System Works.

Juul, J. (2002). The game, the player, the world: Looking for a heart of gameness. *Proceedings at the Level Up: Digital Games Research Conference*. https://doi.org/10.3200/JOEE.39.2.47-58

Kamiya, H. (2006). *Okami*. Capcom.

Katsunori, O. (1998). *Dance Dance Revolution*. Konami.

Kaufman, R., Pinney, J., & Casillas, A. (2013). *The Wolf Among Us*. Telltale Games.

Kimura, H., Hino, S., Oshino, Y., Tezuka, T., & Yamamura, Y. (2019). *Super Mario Maker 2*. Nintendo.

Kitano, M. (2005). *Trauma Center: Under the Knife*. Atlus.

Kitchen, G. (1985). *Garry Kitchen’s Gamemaker*.

Koster, R. (2004). A Theory of Fun.

Laidlaw, M. (1998). *Half-Life*. Sierra Entertainment.

Laidlaw, M. (2004). *Half-Life 2*. USA: Valve Corporation.

Le, M., & Cliffe, J. (1999). *Counter-Strike*. Valve Corporation.

Lee, J. H., Karlova, N., Clarke, R. I., Thornton, K., & Perti, A. (2014). *Facet Analysis of Video Game Genres*. https://doi.org/10.9776/14057

Levine, K., & Finley, A. (2007). *Bioshock*. 2K Games.

LimeZu. (2019). Animated mini characters 3 [Platform] [FREE] by LimeZu. Retrieved October 20, 2020, from https://limezu.itch.io/animated-mini-characters-3-platform-free

Logg, E., & Rains, L. (1979). *Asteroids*. Atari.

Ma, J., & Davis, M. (2012). *Faster Than Light*. Subset Games.

Ma, J., & Davis, M. (2018). *Into The Breach*. Subset Games.

Marutani, T., Saito, Y., & Terasawa, Y. (2010). *Danganronpa: Trigger Happy Havoc*. Spike.

Matsunaga, H. (2007). *Wii Fit*. Nintendo.

Matsushita, T., & Kanno, K. (2012). *Brain Age: Concentration Training*. Nintendo.

McCandlish, M., & Alderman, T. (2014). *Titanfall*. Electronic Arts.

McMillen, E., & Refenes, T. (2010). *Super Meat Boy*. Team Meat.

Meier, S., & Shelley, B. (1991). *Civilization*. MicroProse.

Metzen, C., & Phinney, J. (1998). *StarCraft*. Blizzard Entertainment.

Middleton, P., & Joyce, D. (2012). Lean software management: BBC worldwide case study. *IEEE Transactions on Engineering Management*. https://doi.org/10.1109/TEM.2010.2081675

Miller, R., & Miller, R. (1993). *Myst*. Brøderbund.

Miyamoto, S., Aonuma, E., Koizumi, Y., & Oyama, Y. (2002). *The Legend of Zelda: Wind Waker*. Nintendo.

Miyamoto, S., & Tezuka, T. (1985). *Super Mario Bros.* Japan: Nintendo.

Miyamoto, S., Tezuka, T., Kondo, K., Eguchi, K., Hino, S., & Konno, H. (1990). *Super Mario World*. Nintendo.

Mozgin, D. (2019). Space Cave Tileset by Dmitry Mozgin. Retrieved October 20, 2020, from https://m039.itch.io/blue-space-cave-tileset

Myers, I., & Myers, P. (1980). Gifts Differing: Understanding Personality Type. In *Nicholas Brealey Publishing*. https://doi.org/10.1525/jung.1.1981.2.3.18

Nagata, A., & Akamatsu, H. (1986). *Castlevania*. Konami.

Neuhaus, I. (n.d.). Bfxr by izzy neuhaus. Retrieved October 20, 2020, from https://iznaut.itch.io/bfxr

Newton, I. (1687). Newton's laws of motion | Definition, Examples, & History | Britannica. Retrieved September 17, 2020, from https://www.britannica.com/science/Newtons-laws-of-motion

Norman, D. (2016). The Design of Everyday Things. In *The Design of Everyday Things*. https://doi.org/10.15358/9783800648108

O'Brien, H. L., & Toms, E. G. (2008). What is user engagement? A conceptual framework for defining user engagement with technology. *Journal of the American Society for Information Science and Technology*. https://doi.org/10.1002/asi.20801

Ohno, T. (1988). Toyota Production System Beyond Large -Scale Production Taiichi Ohno. In *Production*.

Overmars, M. (1999). *Game Maker*.

Ozzed. (n.d.). Dunes at Night - An 8-bit and Chiptune album - Ozzed.net. Retrieved October 20, 2020, from https://ozzed.net/music/dunes-at-night.shtml

Pajitnov, A. (1984). *Tetris*. Alexey Pajitnov.

Pajot, L., & Swirsky, J. (2012). *Indie Game: The Movie*. Canada. Retrieved from https://www.imdb.com/title/tt1942884/

Pardo, R., Kaplan, J., & Chilton, T. (2004). *World of Warcraft*. Blizzard Entertainment.

Persson, M., & Bergensten, J. (2011). *Minecraft*. Sweden: Mojang.

Pinchbeck, D. (2012). *Dear Esther*. The Chinese Room.

Polk, R. (2011). Agile & Kanban in coordination. *Proceedings - 2011 Agile Conference, Agile 2011*. https://doi.org/10.1109/AGILE.2011.10

Quivy & Campenhoudt. (2008). Quivy e Campenhoudt - Manual de Investigacao em Ciencias Sociais. *Gradiva*, Vol. 1, pp. 51–57.

Rao, A. (2011). *Bastion*. Supergiant Games.

Rieke, Z., & Fukuda, S. (2003). *Call of Duty*. Activision.

Rodkin, J., Vanaman, S., Anderson, N., Moss, O., & Remo, C. (2016). *Firewatch*. Campo Santo.

Romero, J., Petersen, S., McGee, A., Green, S., & Shanbell, D. (2016). *Doom*. USA: Bethesda Softworks.

Rose, A. (2014, January 18). Creating Dynamic 2D Water Effects in Unity. Retrieved September 17, 2020, from https://gamedevelopment.tutsplus.com/tutorials/creating-dynamic-2d-water-effects-in-unity--gamedev-14143

Rosedale, P. (2003). *Second Life*. Linden Lab.

Russell, S., Graetz, M., Samson, P., & Witaenem, W. (1962). *Spacewar!*

Salen, K., & Zimmerman, E. (2003). *Rules of Play: Game Design Fundamentals*. The MIT Press.

Schell, J. (2008). *The art of game design : a book of lenses*. Elsevier/Morgan Kaufmann.

Senapathi, M., Middleton, P., & Evans, G. (2011). Factors affecting effectiveness of agile usage - Insights from the BBC worldwide case study. *Lecture Notes in Business Information Processing*. https://doi.org/10.1007/978-3-642-20677-1_10

Shin, A. (2019). *Superliminal*. Pillow Castle.

Simpson, M. (2000). *Shogun: Total War*. Electronic Arts.

Smith, D. E. (1983). *Lode Runner*. Brøderbund.

Söderström, J., & Wedin, D. (2012). *Hotline Miami*. Devolver Digital.

Sugg, D. (2017). *Fortnite*. Epic Games.

Swift, K. (2007). *Portal*. Valve Corporation.

Taipale, M. (2010). Huitale - A story of a Finnish lean startup. *Lecture Notes in Business Information Processing*. https://doi.org/10.1007/978-3-642-16416-3_16

Tajiri, S., Masuda, J., & Sugimori, K. (2000). *Pokémon Crystal*. Japan: Nintendo.

Takumi, S., Inaba, A., & Matsukawa, M. (2001). *Phoenix Wright: Ace Attorney*. Capcom.

Tanaka, F. (1994). *Ridge Racer*. Japan: Namco.

Tang, S., Hanneghan, M., & El Rhalibi, A. (2009). Introduction to Games-Based Learning. In *Games-Based Learning Advancements for Multi-Sensory Human Computer Interfaces* (pp. 1–17). IGI Global. https://doi.org/10.4018/978-1-60566-360-9.ch001

Texas A&M University. (n.d.). *Variant*.

Tezuka, T., Miyamoto, S., & Terui, K. (1986). *The Legend of Zelda*. Nintendo.

Thomas, A. (2018). The Effective Use of Game-Based Learning in Education | Andre Thomas | TEDxTAMU - YouTube. Retrieved December 14, 2020, from https://www.youtube.com/watch?v=-X1m7tf9cRQ

Thorson, M., & Berry, N. (2018). *Celeste*. Matt Makes Games.

Tikkanen, H. (2019). *Stormdivers*. Housemarque.

Toy, M., Wichman, G., Arnold, K., & Lane, J. (1980). *Rogue*.

Vanaman, S., Rodkin, J., & Darin, M. (2012). *The Walking Dead*. Telltale Games.

Von Ahn, L., & Dabbish, L. (2008). Designing games with a purpose. *Communications of the ACM*. https://doi.org/10.1145/1378704.1378719

Wolf, M. J. P. (2002). Genre and the video game. In *The Medium of the Video Game*.

Womack, J. P., & Jones, D. T. (1996). Lean Thinking: Banish Waste and Create Wealth in Your Corporation, Revised and Updated. *Simon & Schuster*.

Wong, K. (2018). *Florence*. Annapurna Interactive.

Wreden, D., & Pugh, W. (2013). *The Stanley Parable*. Galactic Cafe.

Wright, W., Hutchinson, A., Chalmers, J., Gingold, C., Librande, S., & Johnson, S. (2008). *Spore*. Electronic Arts.

Yee, N. (n.d.-a). Our Gaming Motivation Data Distilled into a 20-Minute Talk. Retrieved December 29, 2019, from https://quanticfoundry.com/2016/04/07/gdc-talk/

Yee, N. (n.d.-b). The Gamer Motivation Model in Handy Reference Chart and Slides - Quantic Foundry. Retrieved December 29, 2019, from https://quanticfoundry.com/2015/12/15/handy-reference/

Yee, N. (2015). *GAMER MOTIVATION MODEL OVERVIEW & DESCRIPTIONS*.

Yokoi, G., & Okada, S. (1986). *Metroid*. Nintendo.

Yu, D. (2008). *Spelunky*. Mossmouth, LLC.

Zagalo, N. (2020). *Engagement Design: Designing for Interaction Motivations*. Berlin: Springer Nature. https://doi.org/10.1007/978-3-030-37085-5

Appendices

Appendix A

Video example of each streams implementation

https://www.youtube.com/watch?v=d9HPQ6MGsjY (“Cave Game Example - YouTube,” n.d.)