João Pedro Curado de Sousa

Licenciado em Ciências da Engenharia Eletrotécnica e de Computadores

Digital Games' Development Model

Dissertação para a obtenção do Grau de Mestre em
Engenharia Eletrotécnica e de Computadores

Orientador: Doutor Tiago Oliveira Machado de Figueiredo Cardoso; Professor Auxiliar, FCT-UNL

Júri:

Presidente: Doutor João Francisco Alves Martins; Professor Auxiliar, FCT-UNL

Arguentes: Doutor Luís Filipe dos Santos Gomes; Professor Associado, FCT-UNL

Vogais: Doutor Tiago Oliveira Machado de Figueiredo Cardoso; Professor Auxiliar, FCT-UNL

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2017

Digital Games' Development Model

Acknowledgements

Firstly, I would like to thank my family for all the support and for never giving up on me throughout all my academic life. To my girlfriend, for the patience and all the lost days motivating me to do even better.

Secondly, to the Universidade Nova de Lisboa and to the Faculdade de Ciências e Tecnologia, for taking me in and allowing me to study and grow as an individual.

Thirdly, to my colleagues Pedro Leandro from the "Falar Pelos Cotovelos" game and Leonardo Espada from the "Bê-à-Bá" game for allowing me to study the results and accompany the progress throughout the development of the games.

Lastly, to Professor Tiago a huge appreciation, due to the patience, the opportunity given and open-mindset displayed during the creation and development of this model.

Abstract

Nowadays technology follows us everywhere. However, and although always present, sometimes technology slips by unnoticed.

One field that could make better use of technology is the field of social causes, namely the care and aid of individuals with disabilities. Currently, the main drivers of this cause are dedicated people, doctors, professors, etc., who spare some of their time to aid and take care of this individuals but generally don't master most of the modern technologies.

As such the initiative, **Social Tech Booster** was created, with the purpose of boosting the use of information systems and/or digital games whose sole objective is aiding individual with disabilities. This initiative is fueled by students on later stages of their master degree, whose final thesis, takes the form of one of these systems/games. Since then, multiple systems and games have been developed, mostly digital games with mixed results, through methods that change from student to student, due to the lack of a more viable methodology.

As such, in this document a new methodology to successfully develop digital games, capable of dealing with the difficulties linked to social causes, is presented. This methodology makes use of several proven development resources to insure the main beneficiaries are the individuals with disabilities, that will feel an improvement in their quality of life.

Keywords: Software, Digital Games, Game Design, Life Cycle

Resumo

A tecnologia, nos dias de hoje acompanha-nos para qualquer lugar. Contudo e apesar de sempre presente, por vezes a tecnologia passa despercebida. Uma das áreas onde a tecnologia podia ser mais proveitosa são as causas sociais, principalmente no auxílio a indivíduos com deficiências. Atualmente, os principais intervenientes nesta causa, são pessoas dedicadas, médicos, professores, etc., que dedicam parte do seu tempo ao auxilio destes indivíduos e geralmente dispõem um conhecimento limitado das novas tecnologias.

Como resultado nasceu a iniciativa **Social Tech Booster**, com o objetivo de impulsionar o uso de sistemas de informação e/ou jogos digitais que promovam o desenvolvimento de indivíduos com deficiências, alimentada por alunos em final de curso, que realizam as suas teses de mestrado sob a forma de um projeto. Desde então, têm sido desenvolvidos vários projetos, a maioria jogos digitais com resultados mistos, através de metodologias que variam de aluno para aluno, devido à escassez de uma metodologia mais viável e capaz.

Como tal, neste documento é apresentada uma metodologia para desenvolver jogos digitais, capaz de lidar com as dificuldades inerentes às causas sociais. Esta metodologia recorre a diversos elementos de desenvolvimento com provas dadas para garantir que os principais beneficiários são indivíduos com deficiências que irão ver uma melhoria na sua qualidade de vida.

Palavras-chave: Software, Jogos Digitais, Design de Jogos, Ciclo de Vida

Contents

List of Figures

List of Tables

1

Introduction

Technology rules over everyday life. Anywhere we go we are surrounded by technology of every shape and sizes. Each piece of technology usually has underlaying software, created to satisfy our needs and whims, so that we benefit the most out of it.

However, despite being surrounded by technology, we aren't always aware of its possibilities. As such, there are some fields, where technology is used but underestimated and dismissed. One example are the social causes, in which either by lack of involvement of the software creators or technology companies, or by lack of incentives, technology is often disregarded and seen as a complication. Nowadays, the treatments and aids provided to individuals with disabilities are given by dedicated people, doctors, psychologists, etc…, whose time is dedicated to the cause. The resources available to this kind of people are limited and consequently the treatments and aid that they are able to give are often very traditional, limited and short. Yet, almost every single one of us has access to a smartphone capable of running a multitude of applications (apps), which could focus on social causes as these, and like such improve the quality of life of countless individuals with disabilities.

As a first approach to minimize and tackle this situation, the **Social Tech Booster** (STB) initiative was created, with the purpose of boosting the usage of technology in social causes. This initiative aim is to conceive information systems and digital games to children, teenagers and young adults with disabilities, to help in their development and growth. **Social Tech Booster** is nourished by Tiago Cardoso and undergraduate students in their last year, that embrace this cause and develop their master thesis under the form of one game or information system[1], [2]. In figure 1.1, the STB vision is presented. It makes use of institutions with social focus to deliver and gather projects, which are then handed to students to develop and deliver a product. After these students graduate, some are hired to continue within the STB initiative, to finally market the product in a wider scale.

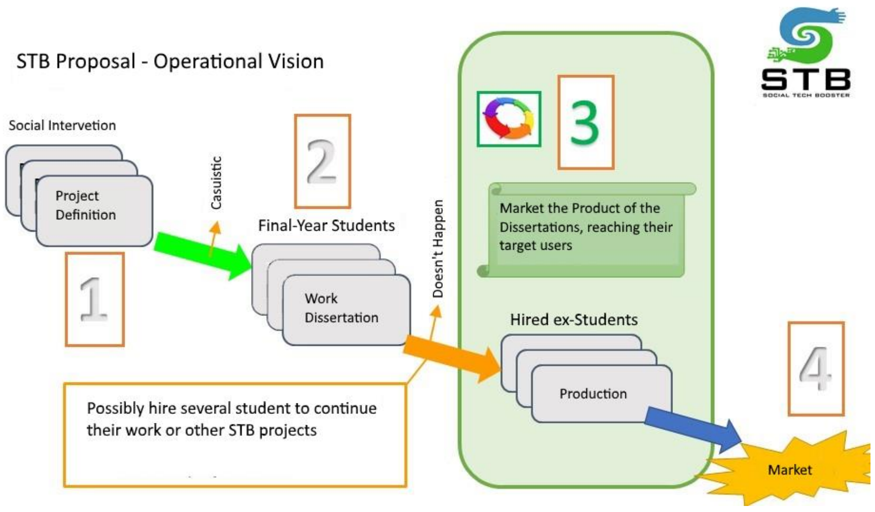

STB Proposal - Operational Vision
STB
SOCIAL TECH BOOSTER
Social Intervetion
Project Definition
1
Casuistic
2
Final-Year Students
Work Dissertation
Doesn't Happen
3
Market the Product of the Dissertations, reaching their target users
Hired ex-Students
Production
4
Market
Possibly hire several student to continue their work or other STB projects

Figure 1.1 – Social Tech Booster's Vision[2]

Throughout the life of this initiative many have been the games and systems developed, some with glimpses of success and others not so lucky. Overall, the impact has been positive but still can be improved upon, further increasing the benefits reaped by the individuals.

Presently, the methodology that supports the development of the projects has changed from student to student, year to year, with some proved success but as opportunities and expectations rise this becomes less tenable. The need for a method capable of supporting the development of these projects, that insures some degree of success in a systematic way, is clear. Most of these projects are digital games, and for this reason, a good model of developing digital games should be adopted.

The Digital Games Industry has been growing at an alarming pace, over the last few years, surpassing most expectations. According to the latest data from *Entertainment Software Association* (ESA), an association composed and formed by multiple famous game developers, like Ubisoft and EA among others, in April 2016, 63% of United States' homes had a regular gamer, who plays at least 3 hours per week. The ESA also claims that in 2015, the digital games industry had a revenue of twenty three and a half billion dollars[3]. Although the data looks great and appealing, digital games and gamification are not a scientific area with plenty of research, it is rather ignored and overlooked. For this reason, there's a huge shortage in information and studies related to this topic and obviously explains the absence of models or methods that support the development of digital games.

Nevertheless, there's one field subject of extensive study and research, very similar to the field of digital games and gamification, the field of software development. Software can be regarded as a set of instructions that allow us to interact and control hardware[4]. Such ample definition lead to the inclusion of numerous types of software, with all kinds of peculiarities. This definition is so broad that includes digital games as just another type of software, so one cannot help but wonder why not use a software development model. The problem lies on the purpose of games, that is to entertain and, in this case that's not all, since this dissertation focus on digital games for social causes. This type of games must entertain while teaching the gamer some type of skill, most times, a somewhat basic skill that will improve the gamer's quality of life.

Like any other form of entertainment, this one can also be seen as a form of art, with room for spontaneity, free or radical thinking and innovations, with leaps of faith. In short, the models and methods proposed until today don't suit the development of digital games, primarily games for social causes, because they are unable to deal with the art component of digital games. As such it must be developed a model capable of dealing with this feature. But although the software development models and methods can't deal with such peculiarity, they still serve as a great resource of guidelines, ideas, methodologies and knowledge.

Another important aspect are the limitations of the parties involved in the development process of these games, especially in the case of games for social causes and for the STB initiative. This process is made of organizations, who provide a service and help individuals with disabilities, and students, whose main commitment is their master thesis and therefore, finishing their degree and studies. One limitation of this sort of organization are the scarce resources they possess, restricting the quality and means of their service. Another obstacle is the knowledge gap between students, whose comfort lies on technology, and service/care providers, who are usually professionals in the health and education industries.

However, not only limitations come from this relation between institution and students. The diversity of the parties involved in the development process of these games is also a positive aspect, as the different parties contribute with separate experiences and feedback, which greatly enriches the development process. All this must be taken into account to avoid the development of an unreliable model.

1.1. Goals

The goals set for this dissertation are:

- Formulation of a model to develop digital games towards social causes.
- Increment the success rate of the games developed within the STB initiative.

1.2. Drive

A new model that enables the systematic development of better games and systems, will bring a new energy to the STB initiative, insuring better grades on the students' dissertations, and better games and systems, therefore letting the students reach a higher sense of personal achievement and improving the quality of the service provided by the organizations to the individuals with disabilities.

The development of a model like this will also attract attention for the cause and for what gaming and modern technologies can do, thus bringing more research, minds and efforts to these topics. This attention could even generate a prosper new business area.

2

State of the art

In this chapter, the basic concepts will be given, discussed and compared, to give the reader a baseline about the state of the art in game development, software development and product's life cycles. This way the reader can stay up to date with these subjects and better understand the reasoning behind the decisions that will be taken in the next chapter.

2.1. Product Life Cycle

In this section, a traditional definition of what is a product's life cycle will be introduced, with the purpose of understanding its stages and comprehend how to extend or mitigate certain stages.

A product goes through four main stages during the course of his life: Introduction, Growth, Maturity and Decline[5]. The recognition and comprehension of this stages is crucial because it enables a better, more accurate and wider perception of the present and a way to predict future events. By knowing the present and understanding the future it is possible to act, in order to obtain the most desirable future or outcome[6]. A typical example of a product's life cycle can be found in figure 2.1. In this case, the product refers to a casual game.

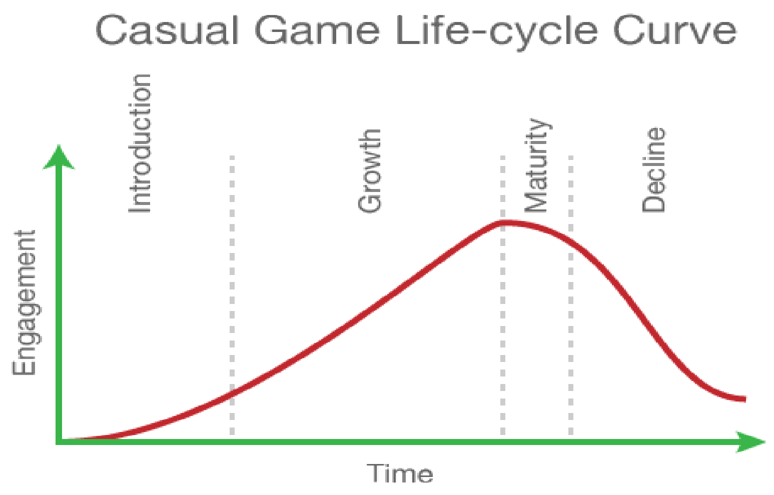


Figure 2.1 – Example of a product's life cycle[7]

2.1.1 Introduction

The first stage by which the product goes through is the Introduction stage. This period is full of uncertainties and unknown risks, all capable of determining the fate of the product, it success or demise. During this phase, it is mandatory that the developers show the need of this product to its future consumers and its target audience, so that it appeals to as many clients as possible. Like any new product, the sales usually are minimal and its productions and supply methods are not optimal.

Beyond the risks and uncertainties, this stage is also the one to ask most from a company or organization, due to the huge investment that a non-optimal product represents. As such some companies opt to systematically wait for the competition to invest and create a market, before investing in their version of the product restraining the amount of risks and the costs.

2.1.2 Growth

The second stage is called Growth. The main characteristic of this stage is its dependency to the products success, if it is successful the product sales will rise, if not they will remain steady and even decline skipping the stage that will be described in the next sub section. Throughout this phase competition rises, mostly because the companies that adopted an anti-pioneer approach now launch their product due to the realization of how healthy the market is.

During this period, the supply and production process is optimized and the profit margins are at an all-time high. The priority shifts from finding customers to keeping the one they have and separate the product from its counterparts. As the profit grows, new more aggressive selling strategies emerge, creating a new stage, with lower prices and less margins that will be discussed next, turning the differentiation between products the make or break detail.

2.1.3 Mature

The next stage is known as Maturity. The first indication that the product reached this stage is the market's saturation, meaning that the sales growth is now very diminished in comparison with the previous stage. By this time most potential targets have already bought a version of the product and so the priority for the company or organization must irrevocably be the differentiation of the product, to retain most of the clients. Above all it is during this phase that the communication between developer and client is most crucial to insure a high standard of client satisfaction and retrieve from the client base suggestions to increase product differentiation, in order to keep up with the competition rate. Although the focus goes to product differentiation, the majority of product are standardized, making even the slightest details seem big differentiation factors.

The duration of this stage can be extremely short, as for example trends, or can be extremely long, lasting through generations, as for example the industry of bolts, that remain widely used since the XIV century.

2.1.4 Decline

The final stage is called Decline. In this phase, consumers run away from the product and sales drop, however, competition remains fierce. Along the way some competitors drop their products and leave the market, meanwhile the remaining companies start to apply gradually more aggressive tactics to revitalize the market.

If they succeed, the market stabilizes and can even show signs of growth and in the best-case scenario it can recede to a scenery that resembles the Growth or Maturity stages. One factor that plays a role in revitalizing a product is its design, style and looks. A corporation who did this very well is Apple, they imbue their products, namely the “Ipod” and “Iphone” family, with modern, clean, different and unique design, with a small differentiating detail, the Apple Button that allows the user to interact with the product in an unusual way than their competitor counterparts. This strategy allowed them to become one of the biggest corporation today.

2.1.5 Extending the life cycle

Comprehending the product’s life stages enables the early planning of the its life. In turn, this planning allows the extension of its life even prior to its launch.

Some actions that extend the life cycle of a product are its promotion towards a more frequent usage by its users, the development of even more ways to use the product, the creation of new users through the expansion of its target market, the discovery of new ways that the users may benefit from the product, the development and increment of new functionalities to newer versions and finally by the continuously update of the product. One example of how this actions influence the life of a product can be found on the article “*Exploit the Product Life Cycle*”, by Theodore Levitt[6], where he studies the nylon case and from where some of this actions come, including figure 2.2.

According to Theodore Levitt[6], planning this extensions creates a set of actives instead of a reactive policy which could demise the product to a less promising future and provides a long-term plan designed to give a new life to the product at the right time, with the exact effort and care required. Lastly, it allows the company a wider control over the product itself.

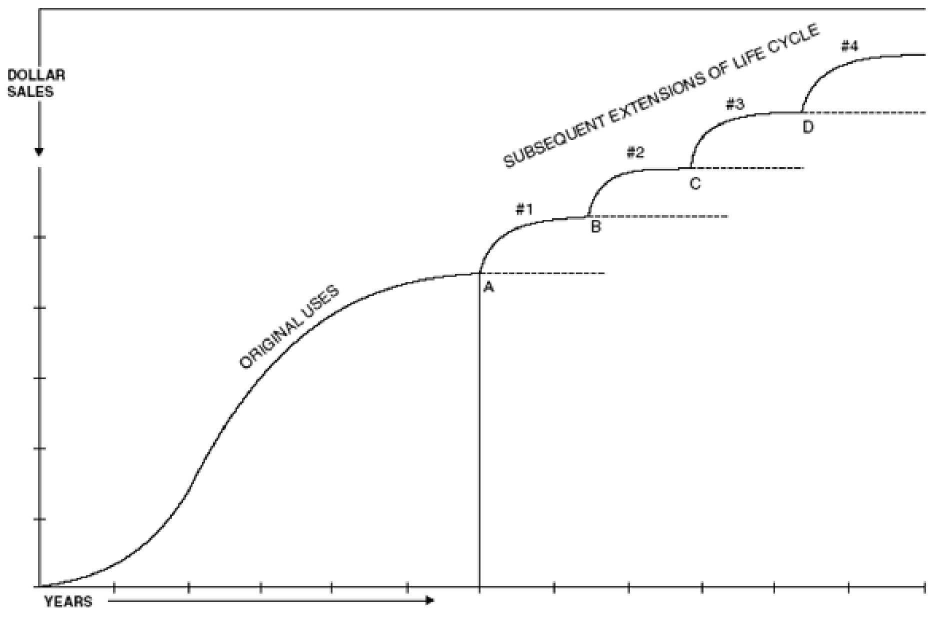


Figure 2.2 – Example of a product's life cycle with extensions actions through incentives[6]

2.2. Software Development

In this section, we will look at the different software development models proposed until today. Each sub section will be composed of a different model, whose ultimate goal is to effectively and efficiently develop software. In the end of each sub section will be an analysis and classification from zero, the lowest value, to five, the highest value, of the model, according to certain key parameters. These parameters have been selected after extensive study of the models presented next, taking into consideration which characteristics the desired model requires and which the present models offer.

The key parameters under which the model will be evaluated are:

- **Documentation** – the quantity of documents generated,
- **Satisfaction** - the probability of the client's satisfaction based on their role in the development process,
- **Agility** – model's capability to deal with unexpected situations,
- **Conclusion** – time needed to finish the product,
- **Functional** – time until achieve a functional product,

- **Quality** – overall quality of the code generated,
- **Evolution** – ease in resuming the project after it is concluded,
- **Application** – ease to apply and enforce.

After classifying every model by all parameters, each model gets an overall rate from zero to five. The rating given to each model is based on the careful review of the strong and weak spots of the model, in conjunction with their advantages and drawbacks present in the literature. A table with all the rating can be found in the last page of this chapter (Table 2.8), for an easier comparison of all model, leading to the conclusion of the weak and strong points of each model.

2.2.1 Code-And-Fix Model

The Code-and-Fix model is the simplest of all models, being for this same reason the most used model among newer programmers with less experience. It consists in the simple task of programming, without any regards for planning or any other concerns, only stopping when the project is finished, and facing the problems as they come. This way, it consumes very little time, being the fastest model to produce results.

It gets its name from how it functions, each time some feature its coded (Code), it's tested, if it fails the tests, it's fixed (Fix) and the cycle repeats itself until the product is finished. It is as simple as it can get.

This simplicity is its strong point, but it is also its main weakness, creating numerous obstacles. The main obstacles, as stated by Barry W. Boehm, in his article "*A Spiral Model of Software Development*"[8] are:

a. Easiness in becoming very disorganized and messy code after a few cycles.

b. The commonness that a product gets rejected due to unfulfillment of all requirements of the client, caused by the lack of a planning and preparation stage.

c. Patches and fixes are near impossible due to the lack of documentation and planning, especially after the product was delivered, preventing an update.

Analysis

Analyzing this model, taking special attention to the parameters set in the beginning of this section, it's possible to reach some conclusion and create the following rating (Table 2.1).

The documentation's absence, the weak client involvement, the lack of agility and the lack of planning and preparing towards the future implies that this model rates poorly. All factors combined form a model with a high chance of creating substandard quality code.

However, its simplicity allows for an easy implementation and application, and a very short final delivery date, but not as short as the delivery date of other models that permit the delivery of an uncomplete but functional project.

This model gets the overall value of 1.

	Code and Fix
Documentation	0
Satisfaction	1
Agility	0
Conclusion	5
Functional	2
Evolution	0
Quality	0
Application	5
Overall	1

Table 2.1 - Code and Fix's rating

2.2.2 Waterfall Model

This model was developed in the fifties, as a result of the experience gained during the development of an aerial defense system called SAGE (Semi-Automated Ground Environment)[8]. Although it was developed in the fifties, it only gained popularity in the seventies, and is, nowadays, still a model that is illustrated in most of the literature linked to software engineering and industrial practice.

There are multiple version of this model, but all share the same philosophy and are characterized by three aspects [4], [9]. All waterfall models are linear, broke down into stages and thoroughly documented. That being said, the first step in this model is to break the project into stages, each stage is well documented and has a very defined goal, as seen in figure 2.3.

Its linearity forces the developer to only move onto the next stage or step after finishing the one he is currently on and it only works this way, which means that after finishing a stage you can't go back. The amount of stages and what to do in them varies with the waterfall model adopted. Picking the right waterfall model depends strongly of the kind of software to be developed, but also of the targeted user or client, as some stages only make sense considering the final user. For example, it wouldn't make sense, to perform a market study to develop a very specific software, only to be used within the company itself, however for this same case, it would make sense to have a stage responsible for creating support documentation, with the goal of teaching its users how to function with it, particularly if they aren't experts in software or are unspecialized personnel.

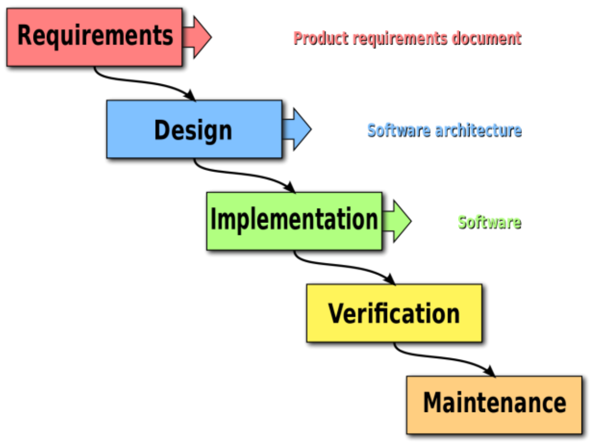


Figure 2.3 – Example of a waterfall model[10]

The contribution from this model was extremely positive to change the software development approach to a more systematic and planned development allowing better results, closer to what the user wants, but it is not perfect. It yet assumes that the process of creation and development is linear and rigid. In reality, it's impossible to follow this model. The linearity of the waterfall model stops any attempt to correct what was previously done in a stage prior to the one the developer currently sits on. For example, if one extra requirement was requested by the client, and the developer was already on a posterior stage to the requirement stage, it would mean that the developer would have to start over. This tiny detail meant that in the best-case scenario, a total redo of the project was required, stage by stage, step by step.

Another problem comes from its rigidity, because the project has to be delivered once without any follow up or future updates, if the result isn't satisfactory due to a missing requirement or a flaw in planning, all the time and effort put in to the project is wasted. It's clear that all the planning and design is made before coding and delivering, so any change made to the first two stages, wouldn't affect the final product. In other words, any requirements that may look secondary or optional at one point can be crucial in the future, turning a successful project into a complete failure. One example would be an accounting software that doesn't allow the addition of a new currency, even if the company doesn't intend to expand to a new market with different currency. If this software was requested by a Portuguese company in 2001, and delivered on January 2nd,2002, this software would be useless because the official currency of Portugal on January 3rd, 2002 would change from being Escudos to Euros. This small detail could pass by unnoticed making the outcome disastrous.

A looped waterfall model was proposed to solve these problems found in figure 2.4, but some problems persevered. The one-time final delivery date, implies that if the client is not pleased with the final product, a new project must be developed. Another problem is that some reasons that led to an unsatisfying product may come from the model itself, due to the small interaction between client and developer causing a lack of feedback throughout the development of the software. This lack of interaction leads the client to miscalculate what requirements are needed, and overlook several important aspects of the software, mostly because of his lack of awareness of the whole project.

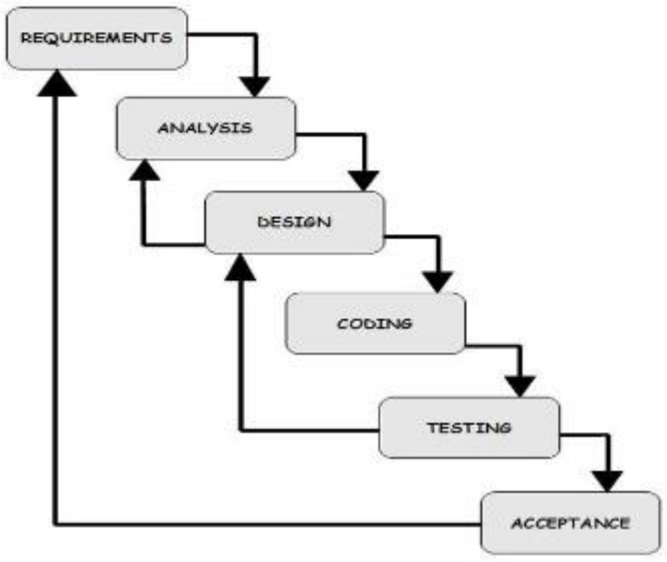


Figure 2.4 - Example of a looped waterfall model[11]

These types of problems motivated the creation and development of new software development models, called evolutionary that try to deal with instabilities within the software development process. The next few models to be discussed will try to solve these problems.

It is also important to note the development of another waterfall variant, called the V-Model. This model was extremely popular because it allowed for a verification/validation to be made before moving on to the next stage, this way reducing the problems of a regular waterfall model. However, this meant a bigger investment of time was needed and an even stricter model with more stages, only suited for larger projects[12].

Analysis

The waterfall model is a lot more structured than the previous model, however its rating is only 2 out of 5 (Table 2.2).

Its stages philosophy and its concern in documenting all aspect of each stage insures a high score in documentation and evolution, due to the ease of resuming and understanding the project, given its documents. Another resulting factor of these two strong points is the slight rise in guaranteeing the clients' satisfaction and the huge bump in code's quality, comparing to the Code-and-Fix model.

Being a more complex and robust than the Code-and-Fix model, the time needed to obtain a final product is a lot higher and obviously so is the time required to obtain a functional product, seen that this model doesn't account for a partial delivery date. Lastly, the lack of agility and the tremendous difficulty in following this model to the letter, is clear. It is inflexible and doesn't withstand any unforeseen obstacles, due to its linearity that would provoke a complete redo of the project.

	Waterfall
Documentation	5
Satisfaction	2
Agility	0
Conclusion	3
Functional	1
Evolution	5
Quality	4
Application	0
Overall	2

Table 2.2 – Waterfall's model rating

2.2.3 Rapid Prototyping Model

This model appeared in the sixties and seventies as a direct answer to Waterfall's model problems. It is frequently used in conjunction with another model, although it can be used alone. The rapid prototyping brings a shift to the software development paradigm, that used to focus mainly in strict stages and a lot of documentation, to a more client focus approach, with lots of interaction between developer and client. To achieve a high rate of interaction with the client, the model makes use of prototypes, more specifically prototypes of fast development, called "rapid prototypes". These prototypes allow the client to know how the system will look, feel and work, and therefore follow the progress of the project, placing him in a better position to express his opinions and actively participate in the development. Each time a prototype is developed, it is delivered to the client, so he may test it and give feedback to the developer. This constant communication is crucial for the model to provide good results, as such the rapid prototyping model is a dynamic model that focus on the client[4], [9], [13]. A typical example of this model cycle can be found in figure 2.5.

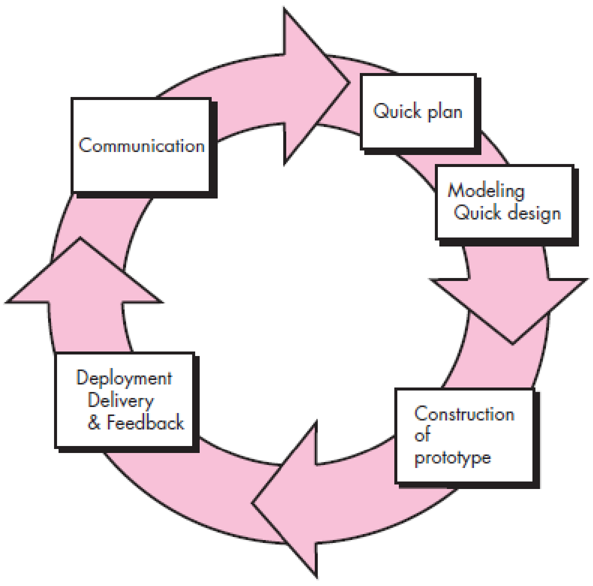


Figure 2.5 - Example of a rapid prototyping model[4]

The shift in the approach brought many advantages, the cut-back on costs, the diminishing of communication related problems, the reduction of wasted time and the production of a product that satisfies the client needs are some of the examples.

It's important to point out that after the process has been concluded and a satisfying prototype achieved, a definitive version of the product must be developed. This version will be based on the last prototype, and not the last prototype itself, because an optimization must occur.

The problems begin in this latter point, being an evolutionary model, numerous prototypes are developed on top of each other until the desired outcome is reached. The key element of these prototypes is their fast development, this quickness means that compromises in terms of efficiency have been made to achieve such short development times. The rapid prototyping's numerous iterations lead to a very inefficiently final prototype, that should be redesigned, optimized and redone completely. In addition to this, during the last iteration if the client wishes to add something else, since the project would have to be remade either way, we would find ourselves in an endless loop of prototypes, never reaching a final product. The numerous iterations of this model also rise new problems, since the final product is based on the last prototype and due to the compromised made on the sum of all prototypes being present in the last prototype, some flaws sometimes slip by unnoticed into the final product, creating a faulty product, with below-average quality.

Moving on, it is important to mention that this model is used primarily in small scale projects, but it can also be used in large scale projects, by breaking it down into smaller projects and applying it to each one individually. If in addition to this model, each small project also apply the Waterfall model, the chances are that we would reach a well-documented project with client's satisfaction insures, although the amount of time required to achieve such result would greatly increase comparing it to the usage standalone of the model[14].

Analysis

The Rapid Prototyping model tries to solve the problems of the Waterfall model at the expense of its strong points (Table 2.3).

In this model, the client grows in relevance and as such the rating in the corresponding parameters increases. Since this model uses prototypes that are essentially early functional versions of the product, it rates the maximum value of 5 as these prototypes are developed extremely fast. It is also a rather simple model, with awareness towards potential future unforeseen events through the usage of prototypes, rising both the grade in agility and application.

However, the increasingly interaction with the clients creates a lack of concern for the documentation, the final delivery date and consequently the potential future of the product after its conclusion. Lastly, the focus on delivering a functional product, as soon as possible, decreases the code's quality, which may be reflected as a below-average product.

This model gets the rating of 3.

	Rapid Prototyping
Documentation	3
Satisfaction	3
Agility	4
Conclusion	2
Functional	5
Evolution	3
Quality	3
Application	4
Overall	3

Table 2.3 – Rapid Prototyping's model rating

2.2.4 Iterative & Incremental Model

This model was created around the same time as the previous one, in the sixties and seventies. Due to its proximity in time to the previous model, it's similar to the Rapid Prototyping model in some respects. The first resemblance is that you can use it in conjunction with another model and how often it happens. The second resemblance lays on the fact that it can also be used multiple times along the same project, by separating the project in smaller sub projects, where this model is applicable.

From the name of this model it's possible to extrapolate its foundations, iteration or repetition and increment or sum. In other words, a project is broke down into blocks, each iteration consists of a block, during that iteration that block is coded and added to the whole project, the more iterations are done, the closer to conclusion the project gets [4], [9]. A typical example can be found in figure 2.6, using the Waterfall model in conjunction with the Iterative and Incremental model.

The first task when following this model is to split the project into subprojects. The second task is to determine the importance and relevancy of each sub project and rate them. After these two tasks are completed, it's time to start coding and implementing the sub projects. Sub projects with a higher rate have priority, therefore are implemented first. Usually, the highest rated sub project is the heart and core of the project.

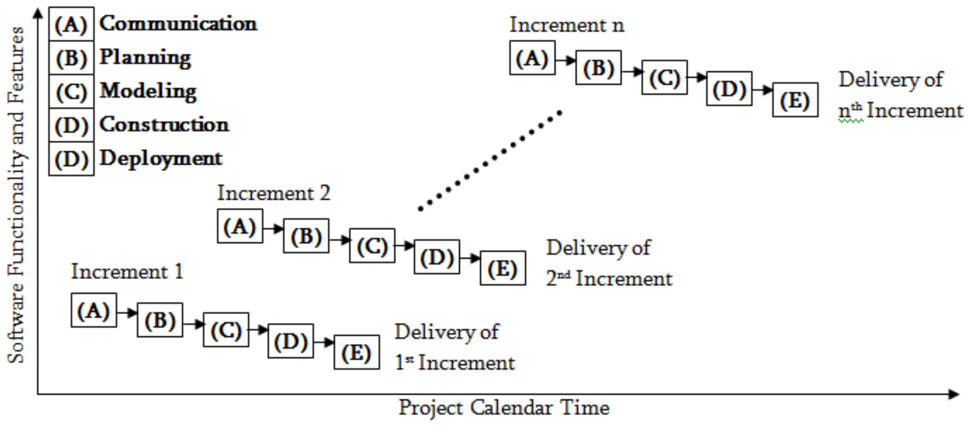


Figure 2.6 - Example of an iterative and incremental model[15]

The constant iterations allow the timely error detection and fix, because each iteration's product is tested prior and after an increment, due to the existence of a functional product from the previous iteration, which allows the results comparison. This functional product, also provides the client a crystal-clear picture of the project’s status enabling him to intervene after each iteration, saving time to both parties.

The Iterative and Incremental model is ideal for big projects, however, due to its nature of breaking a project down into sub projects, it may let some details slip by unnoticed, that might be important when faced with the full project[14].

Using this model simultaneously with a Waterfall model insures that the project is well documented and pleases the client.

Analysis

The Iterative and Incremental model is another approach to solve Waterfall's model and Rapid Prototyping's model problems.

This model is the result of the meetup between a client focused approach and a stricter, documentation and stage-based approach. The frequent client-developer interaction combined with the regular delivery of content grants some agility and insures client's satisfaction. Besides this, the concern for documenting all versions of the project, delivered to the client provides a higher quality code, and also grants that in the future the project can be improved. This way, it's capable of achieving a very positive rating across the board (Table 2.4).

Lastly, it's important to point out that the final delivery date key-parameter loses some of its importance, due to the steady delivery of enhanced and more complete versions of the product through the course of the project, shifting the importance to the other key parameter, the delivery date of a functional product.

	Iterative & Incremental
Documentation	4
Satisfaction	3
Agility	3
Conclusion	3
Functional	4
Evolution	5
Quality	5
Application	4
Overall	4

Table 2.4 - Iterative and Incremental's model rating

2.2.5 Spiral Model

This model was proposed by Barry W. Boehm[8], as an evolutionary model that makes use of others model's key characteristics, the Waterfall model, the Iterative and Incremental model and the Rapid Prototyping model.

This model makes an iterative approach to software development, just like the Iterative and Incremental model, therefore with each iteration the project gets closer to conclusion. During the first iterations, this model makes use of prototypes which are examined by the client, while during latter iterations, a more stable, optimized and final product is presented to the client. In a way, each iteration consists of a Waterfall model, with different stages, that vary from iteration to iteration as seen in figure 2.7.

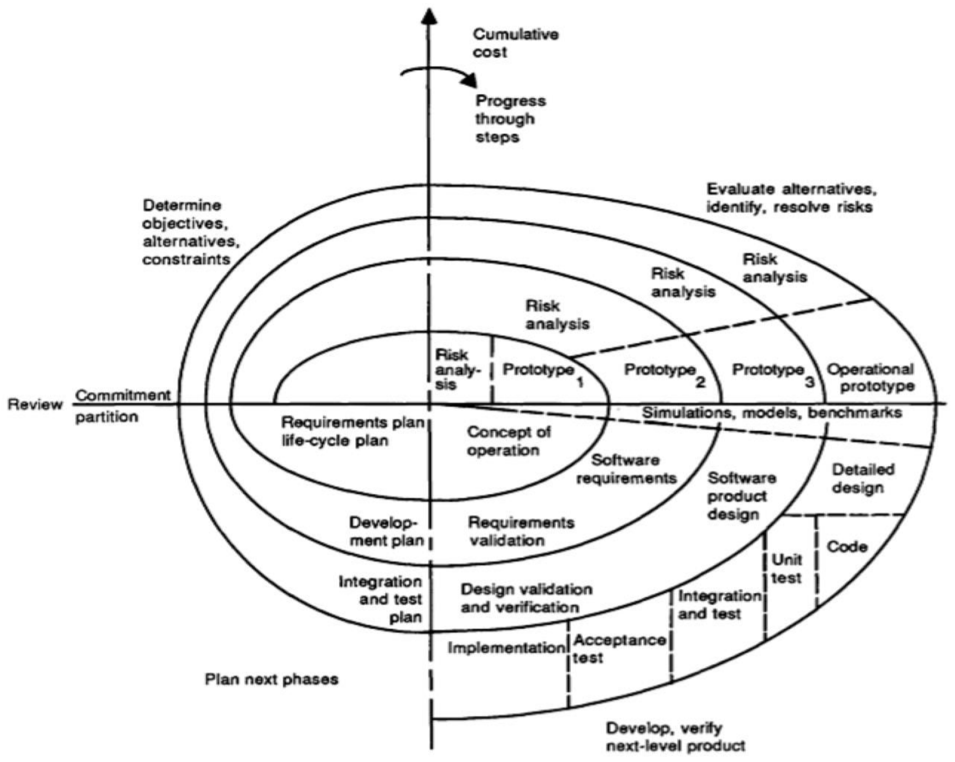


Figure 2.7 – Example of a spiral model as proposed by Barry Boehm[8]

The Spiral model reduces the risk of an unsuccessful project by using prototypes in the first iterations, because of its easy creation with an extremely low development time and its potential to show the client how the project's final product will look like, therefore giving them the chance to feel the product and point out possible flaws, otherwise only detected in latter stages, with a huge negative impact to the project. Furthermore, this model allows to reach the final product with plenty documentation, due to the use of the Waterfall model within each iteration. Lastly, the constant iterations insure an error-free product while also keeping the client updated, simply because he is involved in the process, testing the result of each iteration[4].

This way, the spiral model enables the development of the project with a very high rate to succeed, making use of the strong points of three models. It is also important to note that it's the ideal model for larger projects [14].

Analysis

The Spiral model attempts to solve the problems of the Waterfall model, by combining this with the Rapid Prototyping model and the Iterative and Incremental model.

The strong points of the Waterfall, Rapid Prototyping and Iterative and Incremental models are coupled, originating an incredibly balanced and structured model. However, the weak points of the 3 models are also present in this model, although a lot less visible resulting the rating in Table 2.5.

This model is well documented, prepared for the future, generates high quality code consistently, with insurance of client's satisfaction, but with less than ideal agility, a distant final delivery date and complicated implementation. It's important to emphasize that the final delivery date is of utmost importance to this model because most of the versions delivered to the client are prototypes. Only on latter stages the client is provided with more composed and final versions, closer to the final product.

	Spiral
Documentation	5
Satisfaction	4
Agility	3
Conclusion	2
Functional	4
Evolution	5
Quality	5
Application	3
Overall	4

Table 2.5 – Spiral model's Rating

2.2.6 Agile Processes Model

The Agile Processes Model is the last to be discussed in this sub section. It is the most recent of all the previously described models, being the most widely used model in big software companies, like the European company Spotify[16]. Although this model usually targets teams, its degree of relevance is so high that it must be mention in this document. Being the most recent model dictates that it is also the most controversial one, however the understanding of this model is of extreme importance.

This model has plenty of variants. The most famous one is called eXtreme Programming (XP) and for this reason it's the one we will discuss. All variants follow twelve simple principles, called Principles of Agility[17]. These are:

1. The upmost priority is to satisfy the client through a timely and continuous delivery of high quality software.
2. All changes are welcomed even when the project is on latter stages. The goal is to give the client the competitive edge.
3. Software must be delivered regularly within a short timetable, between two weeks and two months.
4. Both the client and the developers must work together throughout whole the project.
5. Projects are built around motivated individuals. Give them room, support and trust they will deliver.
6. The most efficient and effective way to pass information is through face-to-face meetups.
7. Completely functional software is the main way to measure the progress.
8. This model promotes sustainable development; therefore, the development pace must be able to be kept indefinitely.
9. Good practices and designs that increase agility must be encouraged.
10. Simplicity is key.

11. The best architectures, design and features emerge from self-organizing teams.
12. Regular reflections must occur, in order to optimize all aspects and allow time for adjustments.

Each principle has a different weight depending on the variation of this model used.

In the XP's case, the model is divided into four main activities that follow five simple values. The values are communication, simplicity, feedback, courage and respect. These try to summarize the twelve principles that form this model. Initially, this variant of the agile processes model followed only four values, but in 1999 in the second version of his book "*Extreme Programming Explained*", Kent Beck[18] added respect as the fifth value.

The four main activities are:

- **Planning,**
- **Design,**
- **Coding,**
- **Testing**.

Planning starts with a sharing session, during which the client shares stories that describe the features, requirements, problems, hopes and visions about the software. Each story is rated, the higher the rating, the more priority it has and consequently higher the priority of the feature or requirement it represents. After sharing all stories, these must be grouped in blocks by both the client and developers. Each block is tagged with a time stamp to allow for its implementation. Finally, a plan of action is set considering all block previously created. This way, the clients and developers choose which requirements or features they want to have access to first, as well as the implementation order of all blocks and its final delivery date. As the second principle dictates, it is possible at any stage to change, add or remove stories and by consequence its features and alter the order of implementation and delivery.

The next activity is Design, where the motto is simplicity. During this activity only one block is considered at a time and what is the simplest way to implement it, without considering the following blocks. Just like the previous activity, this one is also subject to a lot of alterations, being often regarded as an activity to perform before and after the activity that will be described next, the coding activity.

After the Design, the Coding stage starts. The first step is to create the set of scenarios/testing units, to enable the next activity, to test the implementation insuring it works as designed. After these units are created, the so-called coding begins. The coding should be done in pairs (*Pair Programming*) and each elements of the pair with a different task, for example one can focus on insuring the quality of the code while the other implements or thinks about a certain detail explicit in the design. As the pairs finish their work, the code must be integrated between all pairs in a daily basis, to reduce problems of compatibility and interface, diminishing repeated segments of code.

The last activity is Testing. This activity is self-explanatory, it consists in running the testing units/scenarios created in the previous activity and delivering the final product for testing to the client.

In the end of this activity another iteration is performed, that goes through the whole model again, until the client is pleased or runs out of stories.

Analysis

The Agile Processes model is the most recent and balanced of all the models discussed, which reflects on its rating (Table 2.6).

This model gives great emphasis to communication and interaction with the client, integrating him in the constructive and development process, this way it achieves the maximum rating for client satisfaction. The same follows for agility, due to tackling the obstacles and problems as soon as they are detected.

This model also rates high in documentation and evolution, due to disposing of complete activities focused in documenting all relevant aspects to the project, as well as, activities that try to reduce the development time while still considering the code's quality.

The only possible negative point of this model is its liability towards the client, since it depends completely of the client's sincerity and availability. For the purpose of this analysis it was assumed that the client is totally cooperative or wouldn't be a client.

	Agile Processes
Documentation	4
Satisfaction	5
Agility	5
Conclusion	4
Functional	4
Evolution	4
Quality	4
Application	5
Overall	4.5

Table 2.6 – Agile Processes' model Rating

2.3. Game Design

The Digital Games Industry has been turning lots of heads due to the new possibilities it can provide and its immense growth over the last years, which continuously exceeded all expectations. One example that shows how much attention videogames have been getting, is the amount of big sports clubs, that are usually linked only to traditional sports, clubs like o Sporting[19], Schalke 04[20], Wolfsburg[21], Besiktas[22], creating E-Sports teams and exchanging or buying and selling digital games' players, just like in traditional sports, to compete at the highest level in leagues created by the game developers.

Despite all the attention, the academics continue to consider digital games development a subject with little to no interest, and consequently there is almost no development of models, similar to the ones described previously for software development, that aid the creation of new digital games. Therefore, most of the knowledge acquired in this subject remains empirical and comes from the experience of few individuals.

However, through the analysis of their experiences, knowledge and ideas, some conclusions can be drawn and similarities found. By compiling all these, it's possible to extract some guidelines that aid in game development.

From three major references in game design eight relevant similarities[23]–[25], where compiled:

- **Goal** – One of the most important aspects to consider when developing a game is what the main goal of the game is. The main goal should be one of the first points to be considered during the design stages. It should be simple, clear and top priority[23]–[25].
- **Theme/Environment** – Theme refers to the context where the game enfolds. It can be set at the same time the goal is set, since both are connected and they must be coherent to avoid inconsistencies and contradictions [23]–[25].
- **Key Elements** – The key elements of a game should be simple and very few, but still relevant. Another important aspect is that all key elements should be equally important and none should be overdevel-

oped when compared with the rest and vice versa. These can be aesthetic, mechanic, technologic or be part of the storyline [23], [25].

- **Meaning** – Actions within the game must have meaning or gamers won't do it, ending up being a waste of feature. The meaning can emerge as a direct result of a player action (immediate result), or from the relation between actions and outcomes when seen from the bigger picture (means to an end). Both types are important when creating a game [23]–[25].
- **Balance** – Within the game there are different mechanics and resources, each with distinct values, if for any reason certain mechanics or resources are clearly superior to others, the remaining will be rendered useless and forgotten. This reduces the depth of the game, as well as, the possibilities given to the player and diminishes the meaning of some actions and the game in general. The lack of balance shows the player clear design problems, and should be treated with extreme care [23]–[25].
- **Iteration** – The process of development and creation of a game must be an iterative process. This process should be used especially for testing the game and continually deploy new prototypes. Only through continuous development and appraisal of the game it is possible to insure it serves its purpose. As such, it's mandatory to develop multiple versions of the game, meaning, various iterations are required to fully create and develop a game [23]–[25].
- **Test** – A testing stage is essential to the development of a game. During its development, a game should be tested numerous times, either by the development team or clients and even members of the targeted audience. These tests can be performed in prototype, during the first stages of the project to speed up the development process. The crucial point is that all flaws and errors are detected early [24], [25].
- **Documentation** – Throughout the development it's important to keep documenting all details of the process. The documents should target both developers as well as gamers. The creation of this documents al-

lows for a better implementation, and allows for an early detection and correction of error or flaws [24], [25].

2.4. Challenges in Game Development

From the previous chapters, the idea that was passed on was that there was a lack of game development models and due to the complexity and unique characteristic of games, software development models weren't capable of providing constant results. In this section will be presented the main problems with game development, as well as provide reasons to adopt some strategies present in software development models.

The challenges present in game development are plenty and the results are disastrous, but many have been solved already by the software industry. To solve problems, as with most issues, these must be recognized and understood so that a solution can be found. The main challengers, as stated by Christopher M. Kanode and Hisham M. Haddad,[26] are:

- **Diverse Assets** – Games are a result of integrating many different expertise. Handling all these poses a challenge as the project grows.
- **Scope** – Lack of a plausible, viable design and planning, means that the project is constantly increasing as features are added. Evermore the addition of feature without a care thought, may lead to the addition of unrealistic features.
- **Publishing** – Bringing the game to the industry can be a challenge due to the lack of investment or outdating technologies, since the game industry is a very competitive and fast-paced industry.
- **Management** – Dealing with some many assets and keeping the project on the right track requires great communication between all members and an excellent oversight.
- **New Technologies/Third party** – The constant competition of the gaming industry leads to a never-ending development of new exciting technologies. Coping with this can prove hard if the wrong technology is chosen.

- **Team Organization** – Keeping all team members in check, thinking the same and working to the same common goal is a challenging task.
- **Development Model** – Choosing the right model can determine the success of the project. Understanding the process is also crucial.

The challenges are present in all types of games and they cause immense amount of problems to the game industry. According to Pretillo [27] in a recent data gathering, only 16% of project in the gaming industry are completed on time and on budget. This research featured twenty games from a wide variety of different styles of games, with distinct developers and methodologies.

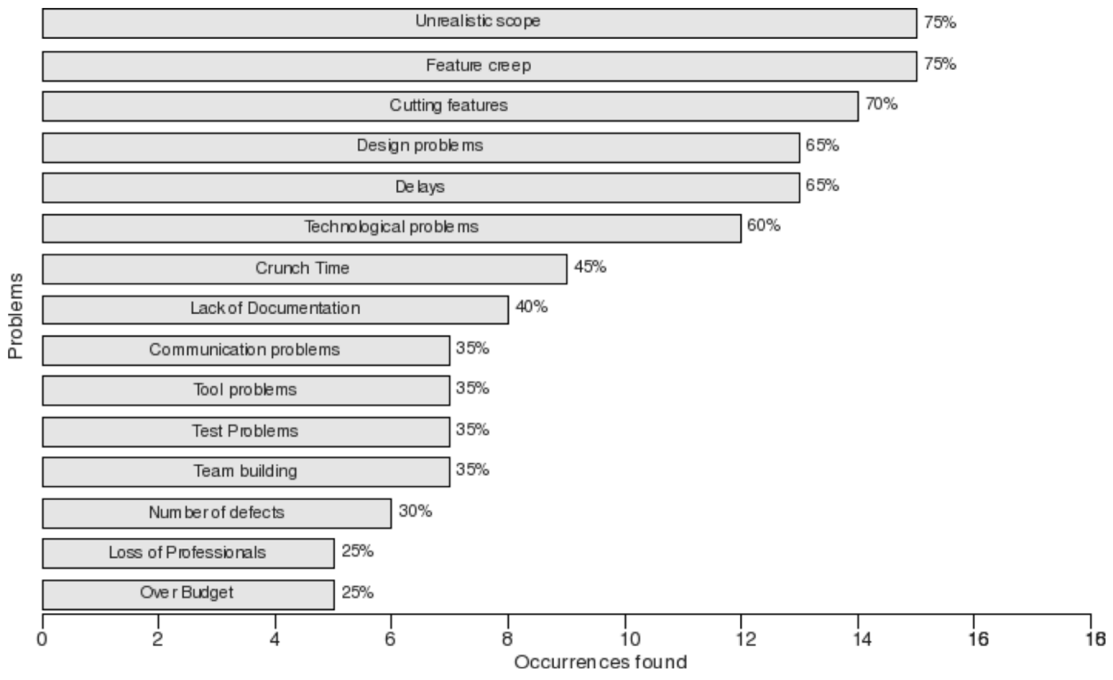


Figure 2.8 - Occurrences found of each problem type in research and analysis performed[27]

It is also possible to observe from the data gathered, which is displayed in figure 2.8, that the problems with greater occurrences (over 50%), are due to bad project management and poor requirement gathering. Avoiding these problems through the usage of a model that prevents these, can prove to be extremely valuable.

From the study of the software development methodologies Christopher M. Kanode and Hisham M. Haddad [26] learned how to solve these problems. The solution to each problem can be found in the table presented below, Table 2.7.

Challenges	Software Engineering Practices
Diverse Assets	Optimize tools and pipeline for integrating assets into the game.
Project Scope	Apply requirements engineering and risk management when translating the GDD to the project scope. Consult with the teams involved so that the project scope is realistic. Consider time needed for game exploration and feature creep.
Game Publishing	Develop deeper communications between the publisher and the development house. Publishers need to be clear with their requirements. Developers need to keep the publisher informed of project progress.
Project Management	Invest in managerial training with an emphasis on project management practices.
Team Organization	Evaluate potential process methods based on team organization and corporate culture. Encourage an attitude of the team as a whole and less importance on individuals.
Development Process	Understand current process and the problems with it. Identify processes that will benefit the project.
Third-Party Technologies	Apply risk management to selection of third-party technology in order to identify which, if any, components would work best for the current project, and for future projects.

Table 2.7 - Solutions found in software development to game development problems

Although most solution are too high-end and target big organizations and teams, they can still contribute toward developing a game development model for the STB initiative via other similar but low-end solutions.

	Documentation	Satisfaction	Agility	Conclusion	Functional	Evolution	Quality	Application	Overall
Agile Processes	4	5	5	4	4	4	4	5	4.5
Spiral	5	4	3	2	4	5	5	3	4
Iterative & Incremental	4	3	3	3	4	5	5	4	4
Rapid Prototyping	3	3	4	2	5	3	3	4	3
Waterfall	5	2	0	3	1	5	4	0	3
Code-and-Fix	0	1	0	5	2	0	0	5	1

Table 2.8 – Rating of all models

3

Digital Games' Development Model

3.1. Problematic

The lack of tools used in therapy sessions for individuals with disabilities is an alarming problem. Most of these tools try to address the needs of the individuals but unfortunately, a few fall short and can even enhance the problems. To minimize this, the STB initiative was created. However, the STB reach has been limited in part due to its development approach, leading to a shortage of individuals that have been able to profit from the tools developed by the STB initiative, simply because they aren´t aware of its existence.

Both problems point to the same conclusion, the need for a structured, well-thought development model capable of deploying tools, in this case mostly games, to the desired targets. This development process must be mastered and integrated within the STB initiative, so that it can run as smooth as possible providing high quality games, to pose as tools, to the one's that need them.

So far, the development processes within the STB initiative have changed year to year, semester to semester, student to student and it reflects on the results. Although with an overall positive outcome, the games developed up until now have had mixed results, while the students successfully finished their degree, there are still games that either don't reach the desired target or the institutions, that asked for them don't put them to use, due to numerous factors, most of those associated to the lack of follow-up or design failures. As such this can't linger much more, a game development method must be adopted, in order to achieve a higher rate of satisfaction from the institutions and deliver high quality games.

But as previously discussed there is a lack of proved game development methods and processes capable of repeatedly provide results in different scenarios and circumstances and particularly in the way that the STB initiative functions with students and institutions.

The few proved methods found were software development processes, however not only these methods cannot cope with the STB inner working as also fail to provide reliable results when it comes to games, because these cannot be viewed as a product of pure engineering and therefore, can't be developed through an engineering straightforward process, like a software development process, although they can be considered as one of the many types of software existent nowadays.

This leads to the conclusion that a new game development process must be created and documented, designed to suite the STB initiative.

3.1.1 Requirements

At the beginning of this project some requirements were laid down, to better suit the STB's initiative inner workings and therefore, achieve a higher rate of success. These requirements pose as foundations to the project, guidelines to obey and fulfill.

They are:

- Incorporate interactions between Institutions and Students
- Agility to deal with sudden problems
- Allow supervision over the project
- Provide relevant and useful games systematically
- Capacity to deal with Students' and Institution's limitations
- Easy implementation and understanding

3.2. Solution

Taking in consideration all the points listed in the previous section and the information in chapter 2, this model will have to be composed of several essential characteristics.

The proposed model will focus heavily in documentation, with multiple instances of interaction with the institution/client, making use of prototypes and several moments for tests through a systematic and iterative approach.

The focus on documentation will provide good foundations to the project while allowing the supervision of the project. The use of prototypes will reduce the amount of time needed to fully document and develop the game, while the constant interaction with the institution will keep them on the loop, insuring the project meets the expectations, which will be validated by the tests performed on later stages. Finally, the systematic and iterative approach provides stability and agility to deal with unexpected problems (figure 3.1).

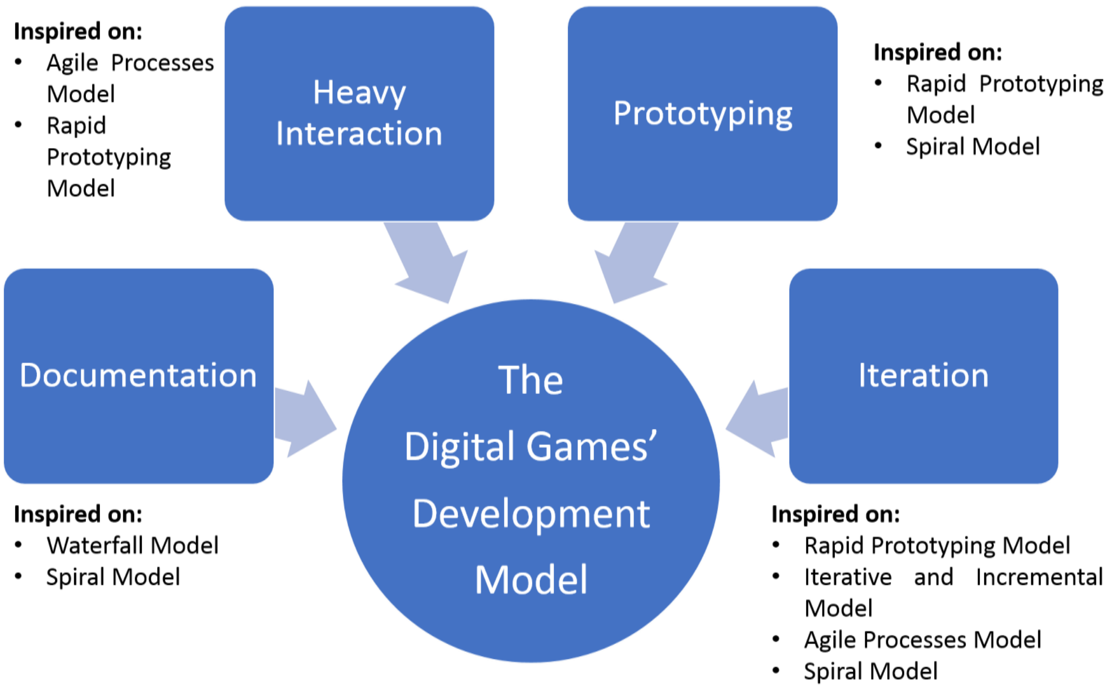


Figure 3.1 - Elements incorporated in the proposed model

3.2.1 Structure

The proposed model is called **The Digital Games' Development Model (DGDM)**, and consist of five distinct stages. Each stage is composed of different activities, targeting various aspects of game development. The five stages are:

- **Planning** – The first stage prepares all the essentials to start a successful project.
- **Design** – This stage builds on top of the foundations laid by the previous stage, grabbing the remaining details and starting the so-called development.
- **Development** – This stage consists of pure engineering, ending with a complete game, with only needing to be validated and inspected.
- **Evaluation** - This stage consists on a set of tests, that aim at validating and confirming that the game is ready for the last stage.
- **Deployment** – The final stage terminates the project and releases the games, trying to maximize their reach.

3.2.2 Planning

The first stage is Planning. Its goal is to prepare, set-up and arrange some of the essentials to the project and lay down foundations to a healthy working environment, insuring less problems on latter stages and in general.

In this stage, students get to know the project and the institution which they will be working with. The idea is to shorten the gap between both parties, eliminate communication problems and settle down the student within the institution. Aside from this, the institution also get the chance to explain the goal of the project and what they foresee and hope, giving an opportunity to get feedback from the student point of view, and this way stimulate the working relationship.

By the end of the stage, the student should have a rough concept of the project, its features, functionalities and some details as scenery or storyline. Therefore, leaving room to schedule the work, separating each task into blocks of 2/3 weeks according to the complexity and importance of the task, where the blocks referring to core features or functionalities have priority.

This stage is divided in three main activities as seen in figure 3.2:

- **Brainstorming,**
- **Meetup,**
- **Set-up.**

These reflect what was described earlier in this sub section.

The first activity is Brainstorming. It consists of a simple, straightforward explanation of the project, followed by an early inspection of the market and brainstorming/creativity exercises.

The second activity is Meetup. It consists of the first set of meetups or re-unions between student and institution where there's an exchange of points of view, opinions and ideas regarding the project. From these, an early design, with a set of features and functionalities, will arise. On which the student and institution must agree on.

Lastly, the Set-up stage consists of the previously described schedule. To schedule the work, the student should follow the same approach used on an Agile Processes model during its planning activity. The features and functionalities should be grouped in blocks, each block should have the duration of 2 to 3 weeks and should be accompanied by a rating reflecting its relevancy. The more relevant, crucial and core blocks have priority and should be scheduled earlier.

It is important to note that all these activities should be documented throughout all its duration, and all documents kept in a portfolio to allow for an easy supervision and overview of the whole project.

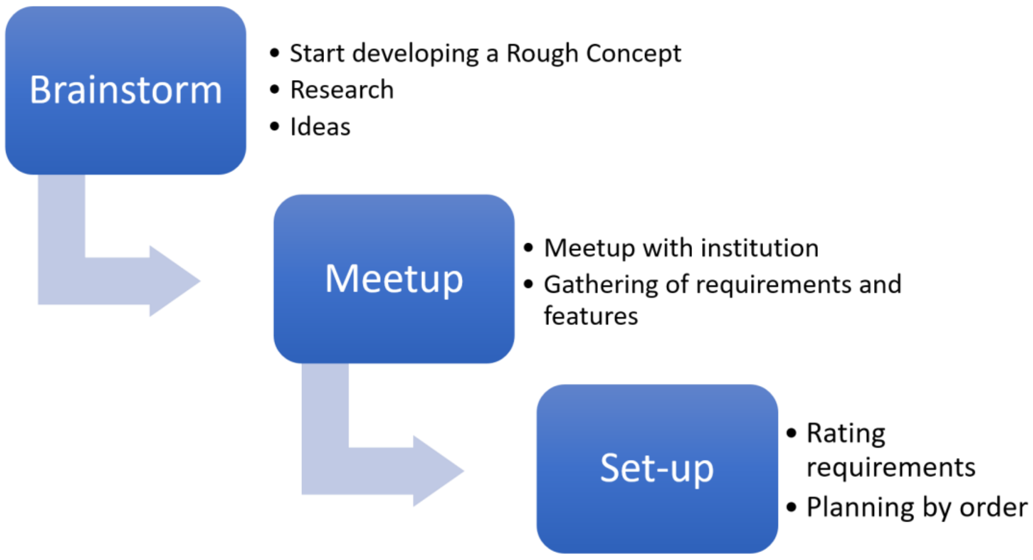


Figure 3.2 -Planning Stage activities

3.2.3 Design

The second stage is Design. The goal of this stage is to create and review the Game Design Documents (GDD), that will serve as guidelines to the project.

Like the last stage, there are three main activities, two of which to support the creation of the game design documents and one to document and review the documents itself as showed in figure 3.3. These documents, like all documents generated, must be kept with the portfolio.

By having these activities, some problems, especially communication related and requirement gathering ones, are prevented and the project is required to get a green pass from both the institution and the supervisor. These activities insure there's enough to start developing the project, while preventing future problems, like the ones mentioned above, all by interacting heavily with the client and supervisor, and reflecting on several crucial aspects.

The three activities are:

- **Appointment,**
- **Initial Prototype,**
- **Re-evaluation.**

Appointment is the first activity and tackles the gap between institution and students, diminishing the knowledge difference and preventing communication failures while also enabling the student to gather valuable information. It consists of one or more go-along visits to the institution, where students follow the members of the institution on their normal day and gather information concerning the resources available to the institution and understand the limitations of their target audience. This way the students get a broader knowledge of the institution limitations, that are not always obvious or explained during the previous stage, and therefore, can design a better suiting project, by knowing how the institution can profit the most from it. These visits can also involve patients and/or their daily caretakers, giving that they consent to it. All information gathered and the conclusions made, must also be preserved in the portfolio.

The second activity is called Initial Prototype. It is a simple activity and as the name suggests its final product is the first prototype. This prototype should be composed by all the instances, features and functionalities that are considered relevant by the student, taking into consideration all the information present in the portfolio. In other words, the prototype should reflect the whole project or game and for this reason, it should be made on paper or in an equally fast way, to pose only as an early draft.

If during the creation of this prototype, any flaws are found and corrected or extra features added, these should be documented and kept in the portfolio. An example of a support document to design this prototype is displayed in annex 1.

The third and final activity is Re-evaluation. This activity consists in a revaluation of all the documents generated, the completion of the Game Design Documents (GDD) and their evaluation by the supervisor and institution. The GDD are a set of sheets made by Tiago Cardoso that go over all aspects essentials to a game. These are completed with the information gathered so far, and with the design decisions made to this point.

The sheets are divided in five core parts:

- **Art,**
- **User Interface,**

- **Game,**
- **Level,**
- **Lead.**

This division allows the project manager and developer to take their time to address all the major game design details stated in section 3 of the previous chapter 2.

The Art section focus on detailing the game theme and environment, by documenting all the style related aspects of the game, since lighting to color palette. It is also made an approach to the key elements of the game, like characters, user interface elements and game mechanisms.

In the User Interface section, all the elements at the user disposal are presented and thoroughly explained. The objective is to get a clear image of how the user will interact with the game and determine which elements are required to implement all the necessary game mechanics and scenes.

In the Game section, the focus is to understand how the game will function overall. To do so, the different gameplay modes, their mechanics, challenges and actions, are addressed. The goal of doing so, is to avoid lack of meaning in one of the gameplay modes and insure the goal of the game is well determined. After this, the focus shifts to balancing this same gameplay modes through the instantiation of the game's internal economy, further enhancing the avoidance of design failure. Finally, it is made an early approach to level design to help incorporate the previous gameplay modes into the game.

The Level section continues the work done in the Game section, further detailing level design. The goal is to document every aspect of each level, including starting, winning and losing conditions, the key elements present and even some style related aspects.

Lastly, the Lead section tackles at two separate design details. The first is the design from a wider viewpoint, taking the task as whole. This way, the designer is asked to summarize all the features and instantiate the game concept. The second is the storyline that supports the game.

All these documents can be found in the annexes 3 to 7.

These GDD accompanied by the initial prototype are extremely valuable to the project, as they represent the game in development and so offer opportunities to detect flaws, inconsistencies and communication errors due to the wider vision they provide. After the GDD are completed, they must be presented to the supervisor and the institution, to insure nothing goes unnoticed and everything is as desired and the institution kept at par. These GDD should also be kept in the portfolio.

If either the institution or supervisor are not pleased, changes can be made and the student should go through both the second and third activity, and therefore generate a new prototype and GDD.

Similarly, if any flaw was detected and corrected or any extra feature added either on this activity or the last, these should be reflected in the GDD. If necessary, adjustments to the planning can also be made in this activity.

This grants some agility and gives an opportunity for the institution to reflect on what asked for.

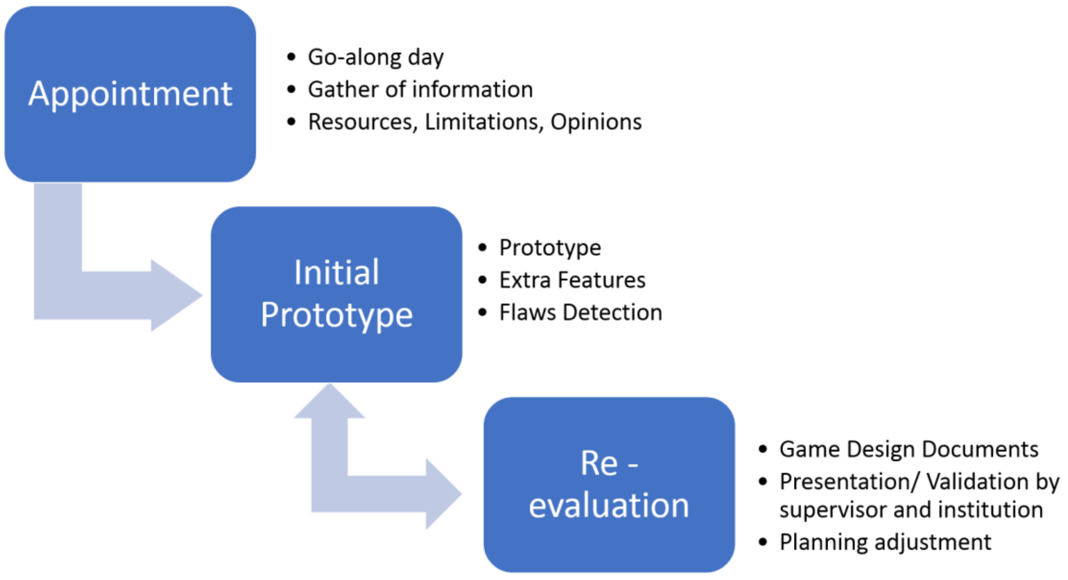


Figure 3.3 – Design Stage activities

3.2.4 Development

The next stage is the Development. The Development stage revolves around the creation of the game.

This stage applies an iterative approach to game development as presented in figure 3.4. Each iteration, represents the development of one block or feature and goes according to the plan made in the Planning stage and perfected in the Design stage.

An iteration consists in three main activities,

- **Production,**
- **Test,**
- **Inspection.**

The Production activity corresponds to the sheer act of programing and developing the block or feature and its addition, after conclusion, to the already fully developed blocks.

The Test activity correspond to the set of tests that the block goes through to insure integrity. The tests, at this stage, are performed by the development team, student and close personnel, and should only reflect the functional aspect, not the fun aspect of the game. The objective is to tune down the occurrence of "bugs" and improve the overall quality of the code from a technical perspective. An example of a support document for these tests can be found in annex 2.

The Inspection activity is the final activity of this stage. It consists of an inspection to the implementation, performed by the supervisor and, in cases that may be necessary, by the institution as well. This inspection lets the institution know how the development is going on and lets them intervene in the production process.

Like the previous stage, if either the institution or supervisor are not fully satisfied, the development goes back to the first activity and the block is improved.

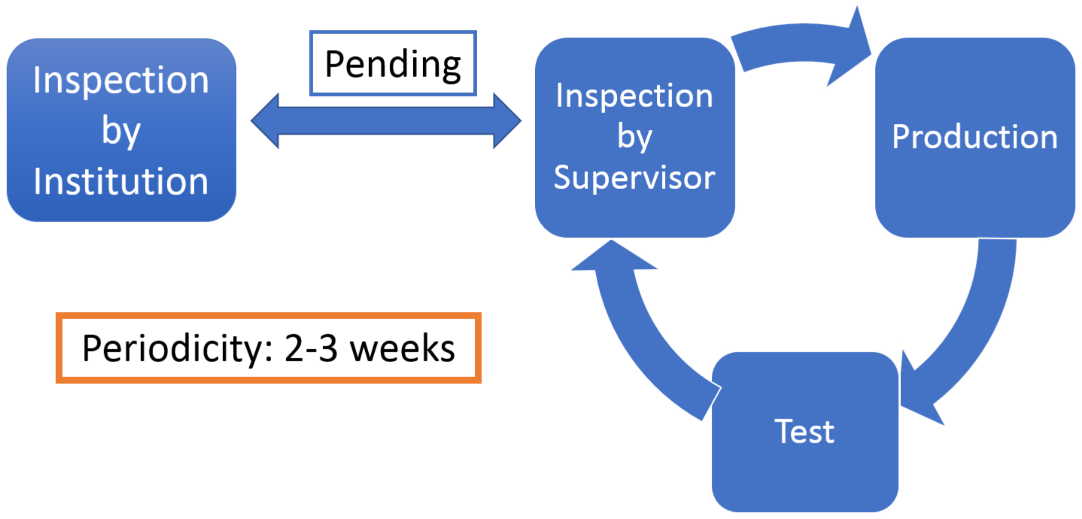


Figure 3.4 - Development Stage activities

3.2.5 Evaluation

The evaluation stage aims at insuring that the game performs as designed, both in the functional aspects as in the fun context. This stage consists on a series of simple tests, where both functional and fun aspects are evaluated.

The tests are composed of play session, where the subjects experience and play the full game and a follow-up form, where the developer asks questions regarding overall performance, existence of bugs, design flaws and other general errors. The goal is to get a clear idea of the subject's opinion and validate the game.

These tests are made in three distinct phases as shown in figure 3.5, the first involves only the development team and members that know all the aspects of the project, the second phase involves members that hardly know or don't know the project and lastly the third phase involves people that fit inside the desired target, members of the institution that don't know the project and other health professionals.

By performing tests with separate groups of individuals with distinct grades of knowledge, the development team is capable of detect flaws, otherwise undetectable. Certain aspects that matter to a group may not matter to others, but may still be relevant to the project. These tests provide an opportunity to fail proof the game.

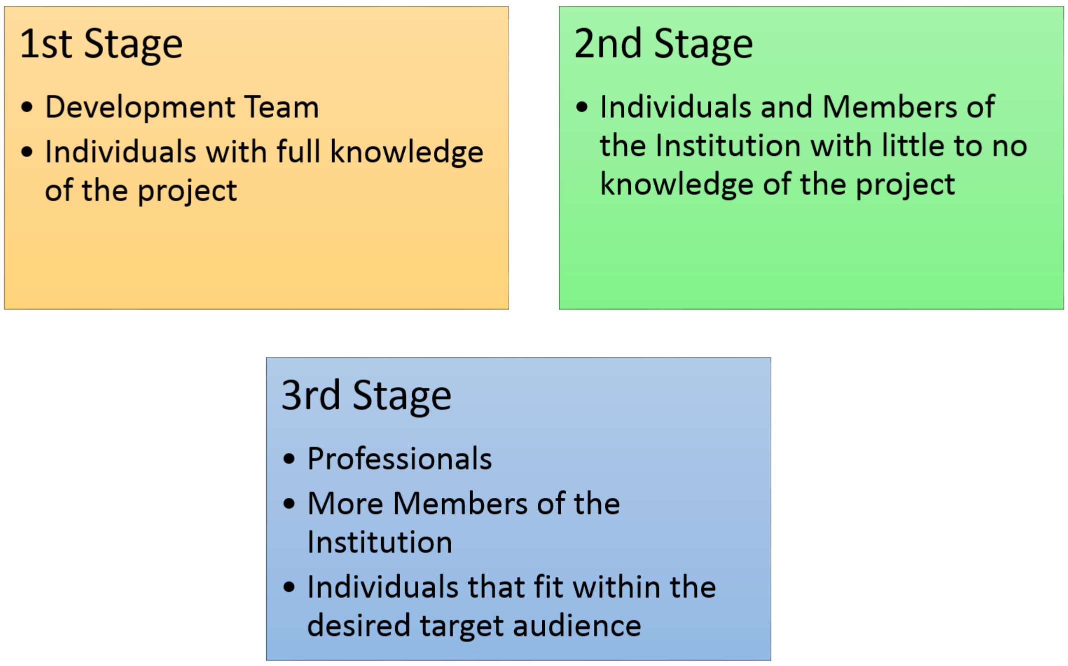

1st Stage
• Development Team
• Individuals with full knowledge of the project
2nd Stage
• Individuals and Members of the Institution with little to no knowledge of the project
3rd Stage
• Professionals
• More Members of the Institution
• Individuals that fit within the desired target audience

Figure 3.5 - Evaluation Stages

3.2.6 Deployment

The final stage is Deployment and is responsible for the deployment and release to the institution and the masses. This stage has two separate activities.

The first and most important activity is the deployment to the institution. The method of deployment must be ideal to the institution and should be one of the more relevant points to observe and discuss during the first activity of the design stage, Appointment.

The second activity is the deployment to the masses and public. This deployment should be done through platforms that are widely used and well known to the public, for example Google Play Store and iTunes Store, to reach as much people as possible.

Although these games target people with disabilities, the STB initiative needs to survive. Therefore, some of the more generic/less focused games can have two versions, one to be handed to the institution and another to be released to the public. The first must be a complete and totally free version, while the second can have a freemium or publicity based business model. The development of the last shouldn't be of the student's responsibility, being totally optional, due to the low academic value.

4

Implementation and Validation

4.1. Validations Methods

In this section, the methods used to validate and support the proposed model will be presented. After developing and creating the DGDM, it had to be validated to insure it did what it was developed for. To do so an application, to support its implementation, was developed, in addition to the development of two games using this model and a quiz to verify the acceptance of the DGDM within the STB initiative.

The goal of the application is to help implement and follow the proposed model, offering the project manager and developer crucial tools that aid in the process of game developing. The application is organized in a certain manner to be coherent with the DGDM. This way, the user is induced to use the proposed model, a systematic development methodology, benefiting both the user, as the DGDM, giving the model popularity, and the user a useful tool. In the next sections, more information of the application will be given, as well as a thorough explanation of its design and functionalities.

Through the development of two games using the proposed model, it's possible to extract numerous information about the DGDM. By analyzing this valuable information, it's attained the possibility of characterizing the proposed model. The main goal of this, is to validate DGDM's efficiency and effectiveness on developing successful games within the STB initiative. In this sense, various data of distinct aspects related to performance of the developed games will be presented and analyzed in the section 3 of this chapter.

4.2. Support App

The support app is called **"Game Development Support App" (GDSA)**. The tool used to develop this App was the Android Studio, the official tool recommended by Google, the creators of Android.

The decision of using this tool limits the number of devices it targets, since it can only develop applications for Android Devices. Still Android devices dominate the market, composing up to 81.7 % of the market, according to a research performed by Gartner[28], one of the leading research companies. This decision was based on the experience acquired during the Master's Degree of Electrical and Computer Engineering and the fact that being the recommended tool supported by the company that develops the world's leading operating system (OS) means it will be the tool with the biggest community and support. In addition to this, the vast amounts of experience working with JAVA programming language, the main language used to develop Android application, lead to the obvious choice of opting for this tool.

The GDSA is an Android App that follows the proposed model, supporting it by offering a convenient way to access most of the tools required to implement the DGDM. The GDSA provides tools for four of the five stages present on the model, as presented in figure 4.1.

They are:

- **Planning,**
- **Design,**
- **Development,**
- **Evaluation.**

The last stage of the model, Deployment, was left out, due to the lack of tools that the GDSA could offer, being an activity to be done exclusively with an unlimited variety of physical or external tools.

For each stage a set of useful tools, that would aid in its activities, was determined. These tools target most of the documentation and organizational aspects of the DGDM. The idea is not to replace the formal generation of these elements but to help developing them on the fly in a convenient and fast way, to latter re-use in its formal creation.

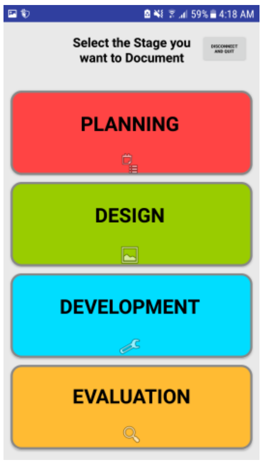


Figure 4.1 - Main menu, separated in each of the model's stages and a disconnect button

The first stage is **Planning**. As previously described this stage has three different activities: Brainstorm, Meetup and Set-Up. Each activity can make use of different tools.

For the Brainstorm activity, it is useful to be able to write down ideas, as well as alter and view them. For the Meetup activity, it is useful to schedule an appointment and afterwards write the Minutes of the Meeting. As for the Set-Up activity, it is important to carefully perform it, and therefore it shouldn't be done on the fly through an App.

The second stage contemplated in the model is **Design**. Similarly, to the previous stage, it also has three different activities: Appointment, Prototype and Re-evaluation.

The Appointment activity requires a method to schedule the appointment as well as a way to take notes to document some interesting and relevant details that may emerge from it. The Prototype activity, as the name mentions, needs to document the prototype, after this and a way to write down ideas. The Re-evaluation activity consist in the creation of the GDD. These documents are composed of the sheets created by Tiago Cardoso, so it’s extremely valuable to offer the user an accessible way to fill these out.

The next stage is **Development**. This stage is composed of mainly coding and testing, and therefore doesn’t require much from a support perspective. Although when using the proposed model, it would be pleasant to be able to schedule appointment and afterwards be able to write down the Minutes of the Meeting. The ability to create and document tests is also extremely valuable.

The final stage is **Evaluation**. In this stage, all the actions revolve around testing. To efficiently test some type of software, a couple of things are required. In this case, three essential actions are required: the ability to schedule and invite individuals to perform the tests, the ability to generate forms and the ability to write down additional feedback.

4.2.1 Requirements

From the previous section, it is possible to extract what is required for each stage. The requirements are presented in the table below, Table 4.1.

Stage	Functionality
Planning	• Add Idea • View/Edit Idea • Add Appointment • Add Minutes of Meeting • View/Edit Minutes of Meeting
Design	• Create GDD • Add Prototype Content • Add Note • Add Appointment
Development	• Add Appointment • Add Minutes of Meeting • Add Test • View/Edit Test • View/Edit Minutes of Meeting
Evaluation	• Add Note • Generate Form • Add Appointment

Table 4.1 - Functionalities required per stage present in the GDSA

4.2.2 Functionalities

After collecting all the requirements, it is time to determine what functionalities the GDSA needs to have to implement all those requirements.

For the table presented on the sub section above, the following functionalities emerged:

- **Logins** – To deal with various users;
- **Save Data** – To create and add Minutes of Meeting, Ideas, Notes, GDD and Prototype Content;
- **Read Data** – To view and edit Minutes of Meeting, Ideas, Notes and GDD;
- **Calendar** – To create and add Appointments;
- **Forms** – To generate forms accessible from an external source.

Logins

Nowadays the ability to deal with different users and keep track of all them is a must. This happens for a multitude of reasons, for instance a user can have multiple devices and keep everything connected and updated or two users can use the same device. Forcing a user to choose only one device and to back-up regularly is unthinkable in the current days. For this reason, it was decided that the GDSA should be able to run both online as offline, but store and manage all its activities online. After this decision, the following ones were obvious and its direct result.

Since the OS was already decided, it made sense to make use of the online solutions provided by the creators of the OS, Google, especially developed for it. Therefore, each user is linked to a Google account.

To login a user only has to insert its Google account credentials and accept all the permissions required to the GDSA run. All the remaining actions required to login are managed by Google's API.

After the first login, the application stores the information about the user signaling that Google account as the default account, so the user doesn't have to insert the Google account credentials every time the app is started. If for any reason, the user wants to swap account, the GDSA offers the option to log out and choose another account. A flowchart of this can be found in figure 4.2.

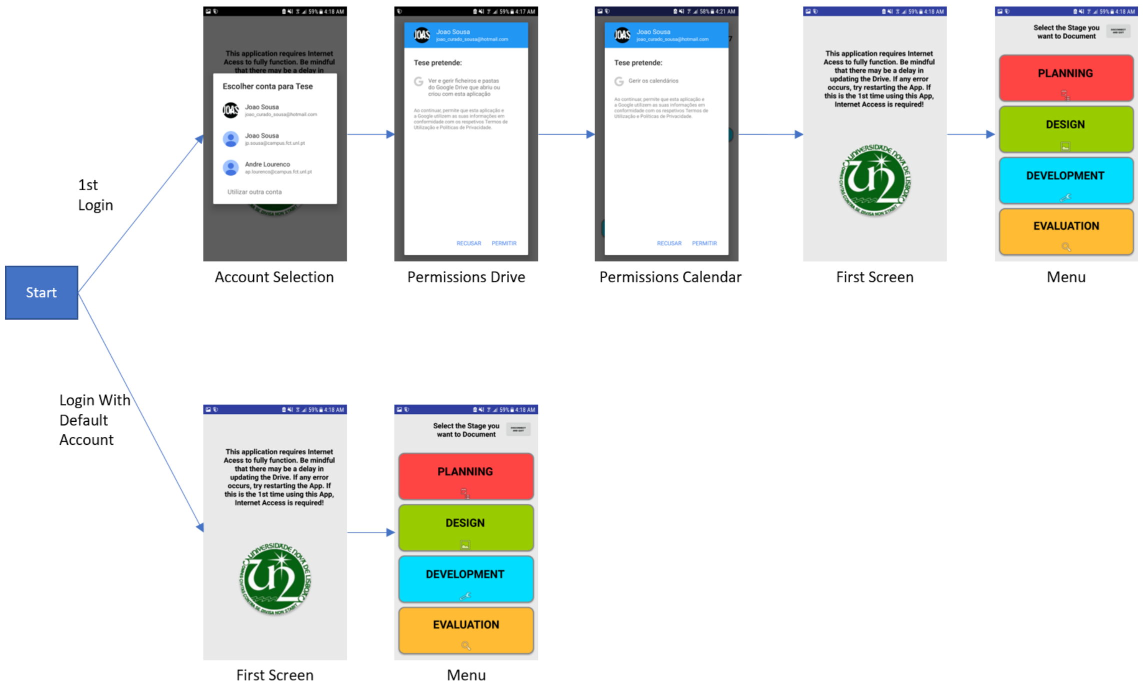

This application requires Internet Acess to fully function. Be mindful that there may be a delay in updating the Drive. If any error occurs, try restarting the App. If this is the 1st time using this App, Internet Access is required!

Figure 4.2 - Flowchart of 1st login and remaining logins with already defaulted account

The only two times the GDSA requires internet access is when the user swaps to a never used account or whose credentials aren't stored in the device and when the application needs to sync with the Google account. The user is informed of these details in the first screen of the application.

Save Data

The ability to store data is crucial to the GDSA, without it wouldn't be possible to store the files created by the application and the GDSA would be rendered useless.

The data is stored using another of Google's online solutions, the Google Drive. Since each user is linked to a Google's account, it is fairly easy to access the Drive and directly store all the content that was created by the user through the GDSA. This way, after the creation or alteration of any content the application uploads to the Drive, so that the user can remotely access everything from everywhere. One might think that, for this reason, the GDSA requires internet access at all times, but due to the fact that all Android devices come with Google's Drive App pre-installed, the GDSA doesn't require it, since every content created by the GDSA is stored directly in the Google's Drive App folder, so that when the user finally has access to the internet, the Google Drive is updated with the new content. It is important to note that inside the Google Drive it's created a new folder that refers exclusively to content created by the app and that this folder is only visible by the user, to keep the Drive clean and all documentation stored together.

The GDSA can create and store two types of data: text documents and Jpeg photographs. A flowchart related to the storage of data is presented in the figure 4.3.

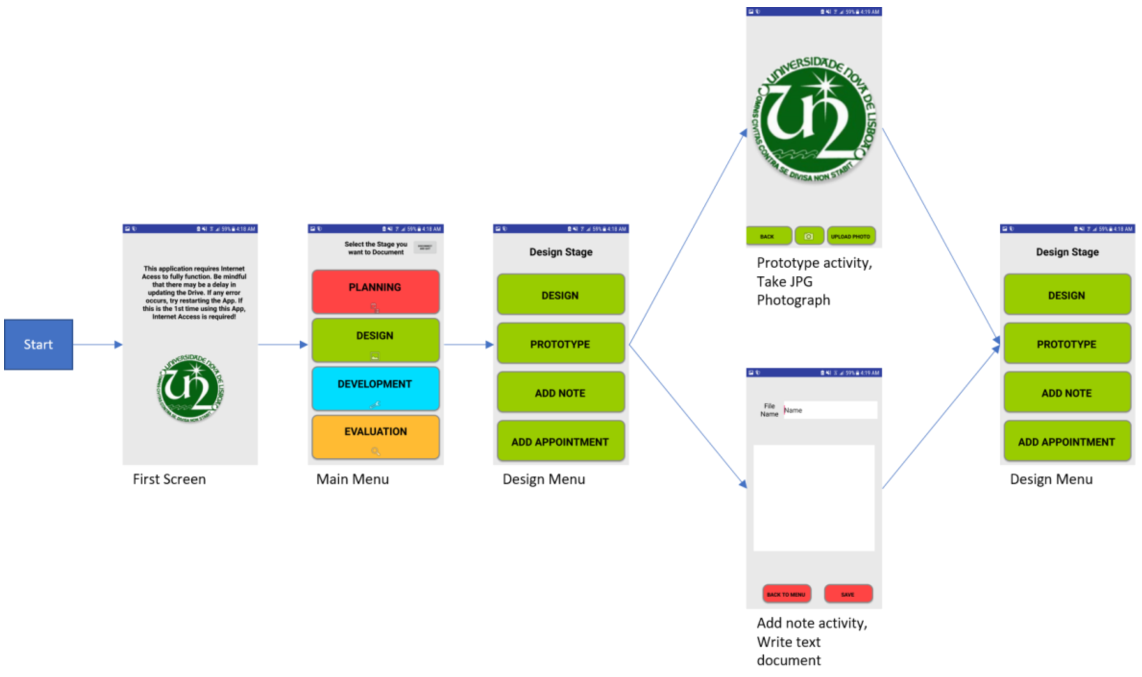
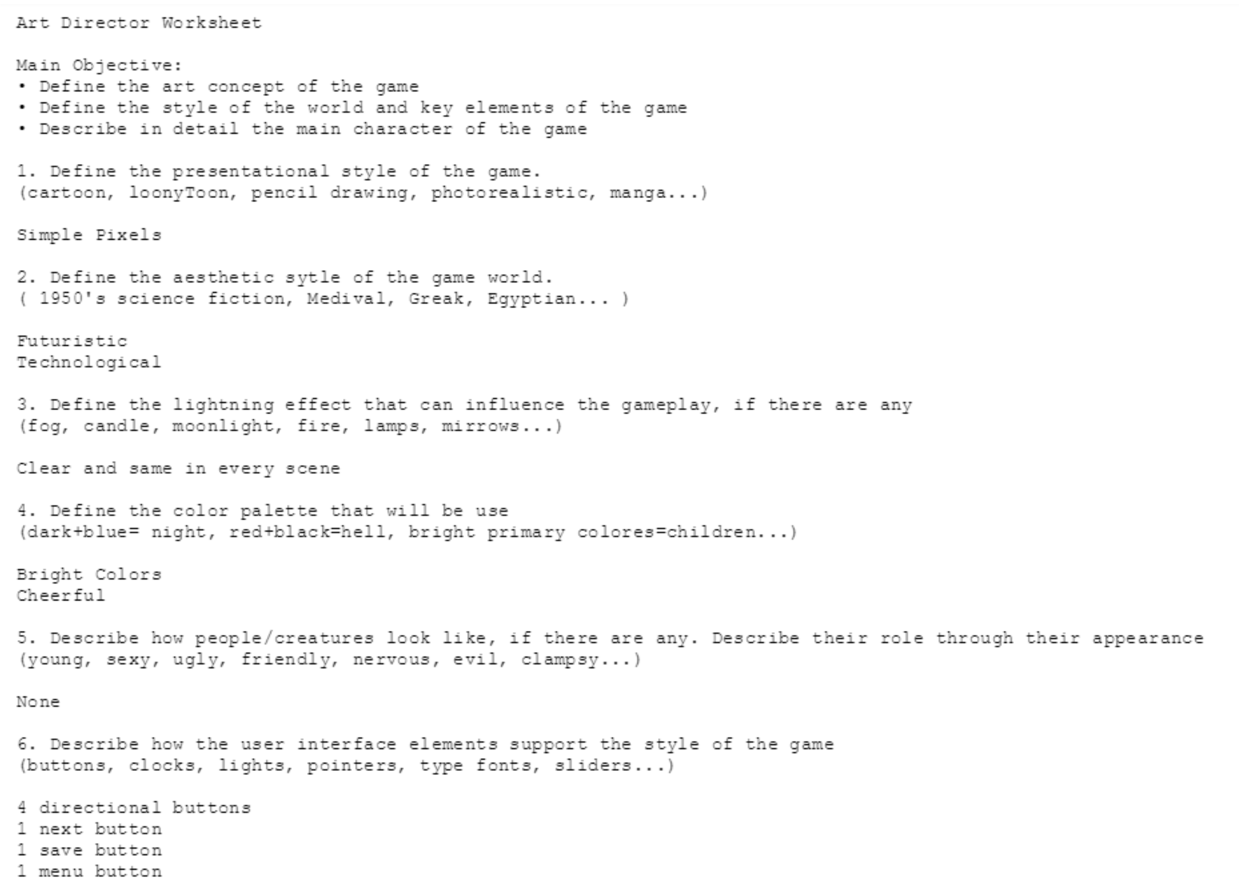

This application requires Internet Acess to fully function. Be mindful that there may be a delay in updating the Drive. If any error occurs, try restarting the App. If this is the 1st time using this App, Internet Access is required!

Figure 4.3 - Flowchart of navigation through Design to save data (text and photo)

The photographs are product of the Prototype activity inside the Design stage. The option of storing photographs as a product of this activity comes from the lack of a better solution, since the other option was to force the user to use a complex, very limited and unintuitive interface to create sketches of a prototype, something that can easily be achieved by using the super-tech solution of pen and paper whose result can also be easily stored through a simple photograph, since memory usage is not a problem when working in a cloud-based app.

The other type of data are text documents, which are the Notes, the Ideas, the Minutes of Meeting and the GDD. Although, they all are stored in this simple format, their structure varies from document to document, which results in completely different documents, with different goals. To create different structures, the GDSA grabs information from a XML file that teaches it how to structure the text document, by adding and removing strings. An example result can be found in figure 4.4. The decision of using text documents emerged from the ease of working with them, unlike Word documents or Pdf documents, that require a huge library to create them.



```
Art Director Worksheet

Main Objective:
• Define the art concept of the game
• Define the style of the world and key elements of the game
• Describe in detail the main character of the game

1. Define the presentational style of the game.
(cartoon, loonyToon, pencil drawing, photorealistic, manga...)

Simple Pixels

2. Define the aesthetic sytle of the game world.
( 1950's science fiction, Medival, Greak, Egyptian... )

Futuristic
Technological

3. Define the lightning effect that can influence the gameplay, if there are any
(fog, candle, moonlight, fire, lamps, mirrows...)

Clear and same in every scene

4. Define the color palette that will be use
(dark+blue= night, red+black=hell, bright primary colores=children...)

Bright Colors
Cheerful

5. Describe how people/creatures look like, if there are any. Describe their role through their appearance
(young, sexy, ugly, friendly, nervous, evil, clampsy...)

None

6. Describe how the user interface elements support the style of the game
(buttons, clocks, lights, pointers, type fonts, sliders...)

4 directional buttons
1 next button
1 save button
1 menu button
```

Figure 4.4 – Example of a GDD generated by the GDSA – Art Section

Read Data

Another crucial functionality is the ability to read data, especially the data generated by this application in past situations. This gives the user the opportunity to review and reshape what was written, providing extreme agility, that is much needed in game development.

Although the GDSA can create two types of data, it can only read one of those types, text documents. The reason for this, comes from the lack of need for reading photographs, as when consulting these, the user should want to do it in a relax and composed manner to carefully think what is the next step.

Reading text documents is easy because it doesn't require libraries, unlike Pdf and Word documents. The difficulty lays in the presentation of all the information to the user. The reason for this is the number of different documents that the GDSA allows to create, from the simplest note to the more complex GDD.

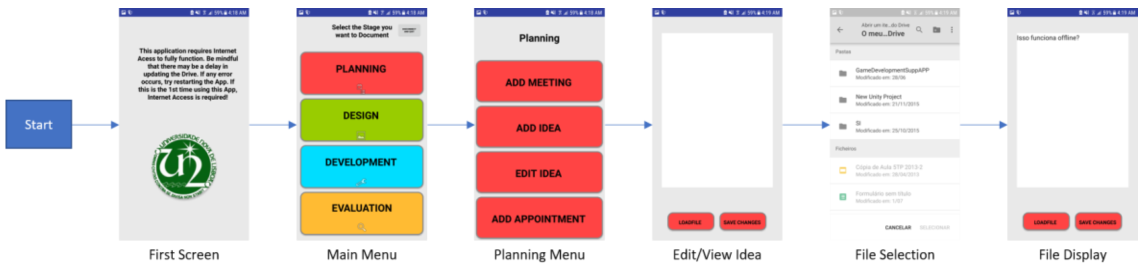


Figure 4.5 – Flowchart of navigation through the Planning stage to read data (Idea)

So, the GDSA reads the document and presents it in the simplest way possible, showing the documents in a window as it is stored. From this window, the user can view, edit or increment any text file, generated by the GDSA, from notes to GDD's. However, the user can also alter the structure of the file and for this reason, this must be done with care. A flowchart of the reading functionality can be found above in figure 4.5.

Similarly, to the previous functionality, it isn't necessary to have an internet connection to read documents, for the same reasons.

Calendar

Like the previous functionalities, the ability to interact with a calendar by scheduling events, allowing the user to organize its time is fundamental to the GDSA.

Google also has a solution for this, Google Calendar. Just as every Google account is linked to a Google Drive, the same happens with Google Calendar. The usage of Google Calendar is extremely valuable because it syncs with the calendar present on the device since they are essentially the same tool, due to both the OS and Calendar being developed by Google, as long the same account is logged inside the GDSA and the OS.

Through the GDSA it's possible to create events with multiple attendees as shown in the figure 4.6, that are visible by consulting the device calendar if using the device default account or Google Calendar through the associated account. In addition to this, a reminder is created to alert the user twenty-four hours before the event starts, which is automatically sent under the form of a push notification, to the device logged in with that account.

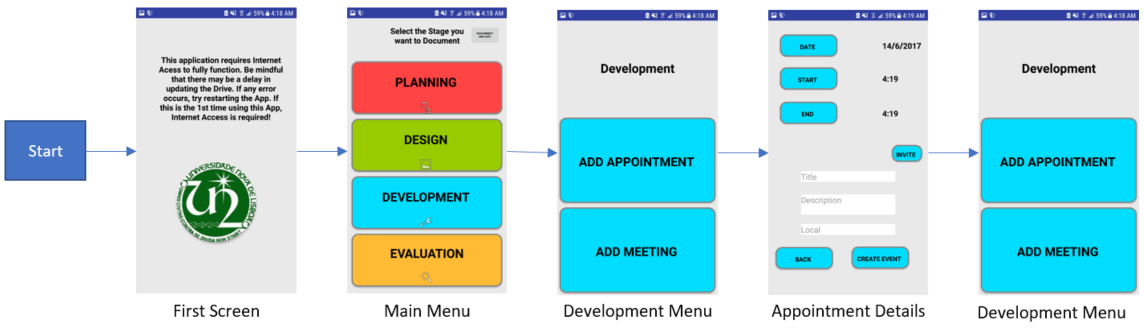

This application requires Internet Acess to fully function. Be mindful that there may be a delay in updating the Drive. If any error occurs, try restarting the App. If this is the 1st time using this App, Internet Access is required!

Figure 4.6 - Flowchart of navigation through Development to add an appointment

When operating without internet connection, the GDSA behaves in a different manner warning the user, that without it, is unable to access the calendar and thus, perform any action relative to it. This happens because the Google Calendar App functions in a distinct way, when compared to the Google Drive App and Google Login.

Forms

The ability to generate forms is fundamental to the GDSA, due to the importance of testing in game development.

Unlike the other functionalities, Google doesn't offer a complete solution, only a half-solution, the Google Forms. Although this solution is very similar to the previous ones, it is not prepared to deal with Android OS and therefore, no Android application can easily communicate directly with it. Luckily, the Google Forms is accessible through any browser.

Just as every Google account is linked to a Google Drive or Google Calendar, the same happens with Google Forms. The Google Forms allows for users to create their forms while also providing a platform to deliver the form and process its results.

In order to the GDSA to interact with Google Forms, it prompts the browser to open a window on its website as presented in figure 4.7. In this window, the user is able to manage all the forms ever created by its account, create new forms from scratch and see the results from each form.

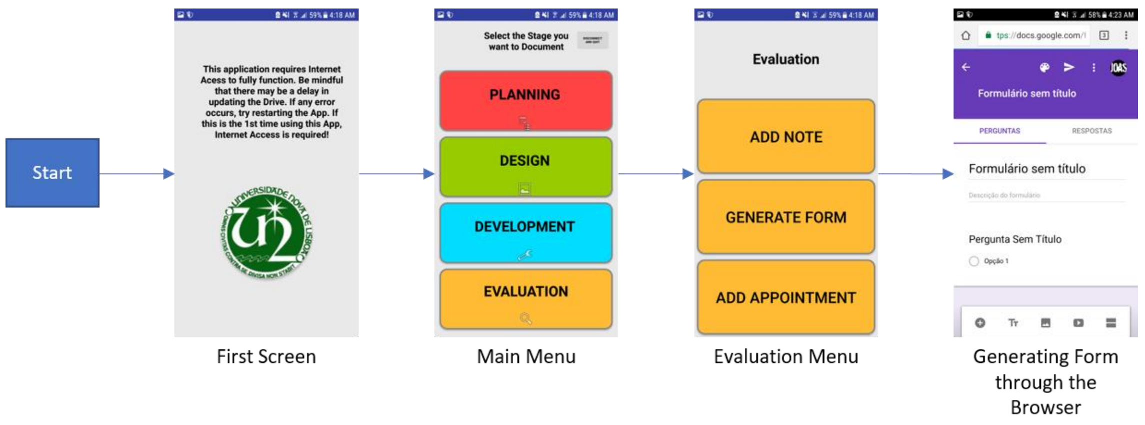

This application requires Internet Acess to fully function. Be mindful that there may be a delay in updating the Drive. If any error occurs, try restarting the App. If this is the 1st time using this App, Internet Access is required!

Figure 4.7 - Flowchart of navigation through the Evaluation stage to generate a form

Since this functionality is dependent of the browser, it requires Internet access at all times.

4.3. Games

In this section, the two games will be presented and discussed. The details of their development will also be examined, by following their progress throughout the model with the aid of their developers and documentation created during this process. This documentation is a direct result of implementing the proposed model, and are a product of the supervision allowed by it, since they were created during and after the various meetups of the DGDM. These documents not only supported the development of the DGDM but also help to validate the proposed model. The reason for this comes from the fact that both games were developed at the same time as the DGDM, therefore, its development was influenced by this model as vice-versa.

The first game developed using this model is called "Bê-à-Bá", a game aimed at aiding in the early stage of learning how to read, for children between 3 and 6 years of age.

The second game developed using this model is called "Falar Pelos Cotovelos", a game aimed at children from 2 to 5 years old, who struggle to say a few words, often because they don't know its meaning (semantics), leading to difficulties in socialization. These difficulties are usually associated with disturbances in the development, commonly called as autism, and must be tackled early to mitigate its effects.

4.3.1 "Bê-à-Bá"

"Bê-à-Bá" was developed by Leonardo Espada[29], with the participation of "Centro Diferenças - Centro de Desenvolvimento Infantil" - a Portuguese leading child-care center, during this academic year. As previously described, this game aims at aiding in the early stage of learning how to read, for children between 3 and 6 years of age. The development of the proposed model was accompanied by the development of "Bê-à-Bá", leading to each decision made towards the DGDM being instantly tested and verified by its implementation within the development of "Bê-à-Bá".

That being said, "Bê-à-Bá" went through the five stages of the Digital Games' Development Model:

- **Planning,**
- **Design,**
- **Development,**
- **Evaluation,**
- **Deployment.**

As previously mentioned, the **Planning** stage consists of three activities, Brainstorming, Meetup and Set-Up.

The Brainstorming activity provided promising ideas but raised the concern that the market could be saturated, due to the vast variety of games already available.

This concern was quickly forgotten when during the Meetup activity, the doctors from Centro Diferenças promptly explained their ideas and visions. The game envisioned was extremely focused in teaching how to read through specific techniques already widely used within Centro Diferenças. Although the game was focused, there was still room to implement extra features, so from the previous activity, some small ideas where selected to be implemented within the game. This way, the game would find its place in the market by targeting a specific niche, that implemented the same techniques as the institution, but with this features a wide enough niche to be viable outside of the institution.

The clear ideas from Centro Diferenças made the last activity from the **Planning** stage, Set-Up, very straightforward, resulting in a fast activity and in a simple, straightforward plan.

Summing up, the three activities in the **Planning** stage drafted enjoyable game, that tried to teach children reading techniques through simple play mechanics direct to the point.

The next stage, **Design**, had its life easy, since the ideas obtained from Centro Diferenças were so clear and simple. The first of the three activities that forms this stage is the Appointment activity. In the "Bê-à-Bá" case, this activity was short, due to the lack of time from both student and institution, but nonetheless the institution introduced its installations, resources and members, giving the chance to gather most of the relevant information to continue the development process.

The second activity was Prototyping, which, due to the clear visions of the doctors, was promptly and swiftly done with only one or two minor difficulties.

The final activity was the Re-evaluation activity. This activity started by filling out the GDD's, which highlighted some weak spots in the design. Through the GDD's and the Prototype, it was possible to detect some lack of meaning within the game, since the player was poorly rewarded for performing well. These soft spots were verified by the institution, however the decision was to continue the development and keep special attention to these weaknesses, because the game still did its main job, of teaching how to read, well.

Entering in the **Development** stage, exceptional care was given to the weakness previously pointed out. During each iteration, particularly on inspections with Centro Diferenças, these weak spots were discussed and analyzed. Each time these would grow increasingly worrying, posing a negative trend.

The following stage was **Evaluation**, and the decision was to thoroughly test the game during the first two phases and determine if the game was ready for the next phase. By the end of the two phases, it was obvious the need to restructure the game and deal with its soft spots.

Therefore, the game went through another **Design**, **Development** and **Evaluation** stages. Thanks to the GDD's and prototype during the **Design** stage, the inspections during the **Development** stage and the two first phases of the **Evaluation** stage, the problem with the game was completely identified, making the new rundown of these stages easier.

The returning to the **Design** stage, meant that not everything had to be redone, in this case, only certain aspects had to be re-imagined. Since the problem laid in a design aspect and not in any limitation associated with resources or installations, the Appointment activity morphed into a more discussion centered activity, in other words a meetup. From this, came the idea to add more depth to the game, by adding extra features that would provide an increase positive feedback when the player performed well. Moving on to the Prototype activity, a new feature began to taken form. This feature would be an external element of the game that allowed the player to unlock content by beating new levels of difficulty with a certain degree of proficiency, the higher the proficiency the better content was unlocked. This way, the player would get an incentive to play without being private of improving, a "two birds for one stone" situation. Reaching the Re-evaluation activity, the game seemed and felt like a different game, and the new game was approved by both professor as institution and the GDD were updated.

On the next stage, **Development**, a couple of iterations were required to fully implement this feature, due to its sheer complexity. In the last iteration, the game had a new touch to it and was ready to be evaluated on the next stage.

During the next stage, **Evaluation**, the game aced both two phases. Leaving for last the third phase as a final seal of approval, before deploying the game.

In the last phase, several tests were performed with individuals that fit the target audience. These tests took place during some of the sessions with these same individuals. The results of these tests are sent directly into an email address which stores all the information regarding the player, the answers, the questions, the success ratio and to which stage, phase, level and sub-level those stats correspond to. By storing this information in an email address, all this data is online and accessible by everyone, everywhere with the email account details and more specifically the institution.

After analyzing the results, one session with one particular subject stood out. This session focused on one stage of the game, with ten different levels, each with three separate phases. From this session, the tables 4.2 to 4.6 emerged. These tables illustrate all the levels where the subject made mistakes.

Table 4.2 – Results from level 0

Phase	Level	Animal	Expected	Answer
1	0	Sapo	s	s
1	0	Sapo	a	a
1	0	Sapo	p	p
1	0	Sapo	o	o
2	0	Sapo	sa	sa
2	0	Sapo	po	po
3	0	Sapo	s	s
3	0	Sapo	a	a
3	0	Sapo	p	o
3	0	Sapo	p	p
3	0	Sapo	o	o

Table 4.3 – Results from level 1

Phase	Level	Animal	Expected	Answer
1	1	Javali	j	j
1	1	Javali	a	a
1	1	Javali	v	v
1	1	Javali	a	l
1	1	Javali	a	a
1	1	Javali	l	l
1	1	Javali	i	i
2	1	Javali	ja	ja
2	1	Javali	va	va
2	1	Javali	li	li
3	1	Javali	j	j
3	1	Javali	a	a
3	1	Javali	v	v
3	1	Javali	a	a
3	1	Javali	l	l
3	1	Javali	i	i

Table 4.4 – Results from level 3

Phase	Level	Animal	Expected	Answer
1	3	bode	b	b
1	3	bode	o	o
1	3	bode	d	e
1	3	bode	d	e
1	3	bode	d	d
1	3	bode	e	e
2	3	bode	bo	bo
2	3	bode	de	de
3	3	bode	b	b
3	3	bode	o	o
3	3	bode	d	d
3	3	bode	e	e

Table 4.5 – Results from level 7

Phase	Level	Animal	Expected	Answer
1	7	Mula	m	m
1	7	Mula	u	u
1	7	mula	l	l
1	7	mula	a	a
2	7	mula	mu	mu
2	7	mula	la	la
3	7	mula	m	m
3	7	mula	u	l
3	7	mula	u	l
3	7	mula	u	l
3	7	mula	u	u
3	7	mula	l	a
3	7	mula	l	l
3	7	mula	a	a

Table 4.6 – Results from level 9

Phase	Level	Animal	Expected	Answer
1	9	búfalo	b	b
1	9	búfalo	ú	ú
1	9	búfalo	f	f
1	9	búfalo	a	a
1	9	búfalo	l	l
1	9	búfalo	o	o
2	9	búfalo	bú	bú
2	9	búfalo	fa	fa
2	9	búfalo	lo	lo
3	9	búfalo	b	b
3	9	búfalo	ú	ú
3	9	búfalo	f	f
3	9	búfalo	a	a
3	9	búfalo	l	o
3	9	búfalo	l	l
3	9	búfalo	o	o

From these results, it's possible to take some conclusions. Starting from the number of tables presented is clear that the subject, made mistakes in 5 of the 10 levels, therefore obtaining a ratio of 50%. This fact suggests the game is well designed as the balance is as good it can get.

To further confirm this, an analysis to each phase was performed, culminating in figure 4.8.

From this figure, it is easily observed that the phase that causes the most trouble is the last phase, corresponding to 67% of the total errors. This lines up with the expected, as phases 1 and 3 are considered harder by tackling the task of spelling with and without entropy elements, while phase 2 only deals with syllables.

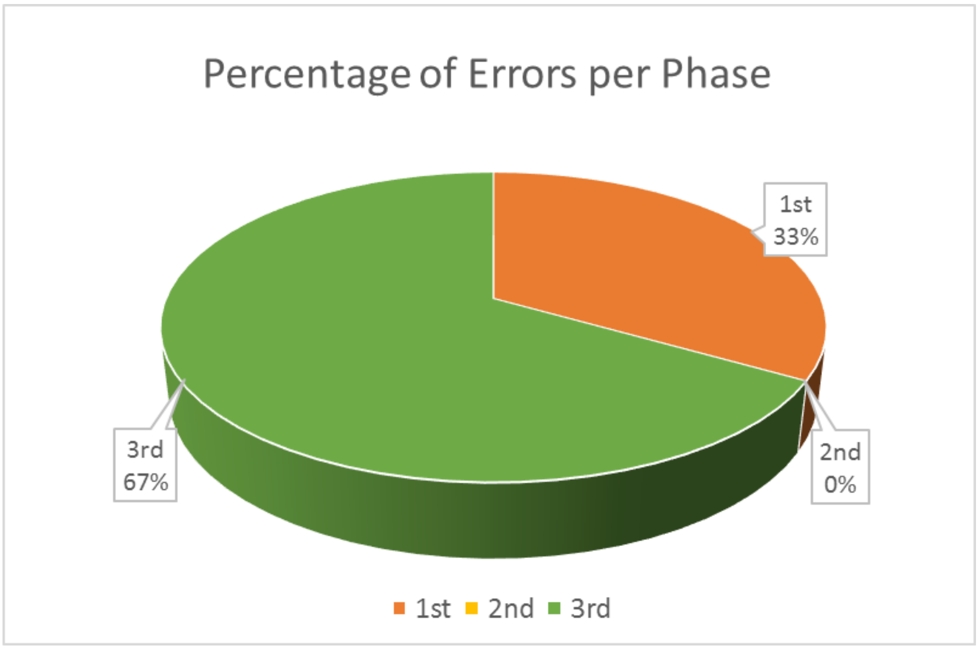


Figure 4.8 – Percentage of Errors per Phase

Lastly, from the tables it is also possible to gather other type of information. This is true not only for this subject but for all subjects, as this information is stored inside an email account as previously stated. The level with the highest numbers of errors, the type/characteristics of words that cause the most problems or misunderstood letters/syllables are examples of other useful information easily obtained via the game. In this case, the subject reveals problems with the word "mula".

These results confirm the game performs as designed and the main goal is reached of aiding in challenging task of the teaching of how to read, leaving one last stage to be performed, **Deployment**.

After verifying the game performs well, it was deployed to the institution under the physic form of a flash drive with a apk file, installable in Android devices. But this is not the only way the game was deployed, as it is currently available in the STB official website (http://stb.uninova.pt), under the section attributed to the "Bê-à-Bá" project. In addition to this, the game is close to be deployed into the Google Play Store, to insure the game the maximum reach possible.

4.3.2 "Falar Pelos Cotovelos"

"Falar pelos Cotovelos" was developed by Pedro Leandro[30], with the participation of "Centro Diferenças - Centro de Desenvolvimento Infantil" - a Portuguese leading child-care center, during this academic year, similarly with "Bê-à-Bá". This game aims at children with disturbances in their development, visible by their difficulty with words, mostly with their semantic. Since their difficulties are visible, "Falar Pelos Cotovelos" tackles these same problems through what makes them visible, words. Just like the game previously described, "Falar Pelos Cotovelos" was also developed in conjunction with the proposed model, therefore, each decision made towards the DGDM was instantly tested and verified by its implementation within the development of "Falar Pelos Cotovelos".

"Falar Pelos Cotovelos" also went through the five stages of the Digital Games' Development Model:

- **Planning,**
- **Design,**
- **Development,**
- **Evaluation,**
- **Deployment.**

The **Planning** stage consists of three separate activities. On the Brainstorming activity, lots of ideas emerged but just like the "Bê-à-Bá" game, the market seemed saturated which raised some concerns.

These concerns were wiped away during the Meetup activity, due to the visions, hopes and ideas provided by the doctors from Centro Diferenças. The vision for the game was crystal clear, it would be a simple, straightforward game, implementing specific techniques used in Centro Diferenças during the treatments and appointments. Contrary to the other game, there was no room for extra features, due to the target audience. Children with autism see the world differently, having huge trouble with the most basic of task, often due to lack concentration and focus. Developing a game for these children meant that the game would inherit some limitations, due to their condition. This fact was heavily emphasized and for this reason, the indications, ideas and vision of the doctors from Centro Diferenças were followed almost to the tiniest detail.

On the last activity of the **Planning** stage, Set-Up, no difficulties were found, leading to the creating of a simple clear-cut plan, mostly because so much work was done on the previous activity by Centro Diferenças.

Moving on to the next stage, **Design**, almost no difficulties were found, in part due to most of the path being already set by Centro Diferenças. During the Appointment activity, the first of the three activities that form this stage, a lot of information was gathered from both the institution and the target audience, thanks to the opportunity to know and somewhat follow children that would profit from the use of this game. From this activity, one particular detail emerged has extremely interesting, the existence of one game which was already being tested in the sessions but due to its high price and complexity in certain aspects, it didn't fulfill all the requirements and therefore, was inefficient.

The Prototyping activity is the second activity of this stage. This activity had most of the work already done by Centro Diferenças, due to how clear the ideas and visions were when presented on the Meetup activity of the previous stage. Leaving only room for documenting, sketching a few resources and filling the gaps. After terminating this activity, the only thing left to do in this stage, was the Re-evaluation activity, consisting in the generation of the GDD's and the presentation of these and the prototype to the professor and Centro Diferenças.

By the end of this activity, all the parties involved showed elevated levels of motivation and excitement when faced with the prospects of this game and how well the entire process was being unfold.

Reaching the **Development** stage with this level of enthusiasm was an extra push, but still several iterations were needed to fully code the game as expected.

The following stage was the **Evaluation** stage. All the enthusiasm involving the game was justified in this stage. The game passed all three phases with flying colors, due to its clear design and objective focus.

Focusing on the last phase, several tests were made with individuals within the desired target audience. These tests involved many sessions, with three different individuals. From the results of these tests, it's possible to draw some conclusions about the success of the game. These results were gathered from a file, created while the individual was playing the game. This file contained information about the time spent to answer the question, what were the answers and actions performed, what was the question and difficulty and which individual was playing. It's important to note that this file is constantly updated so that this data is available to the institution.

From the analysis of the results, five tables were created which then, originated a success ratio graphic and two comparison graphics. By using these resources a few conclusions were made.

The success ratio graphic is presented below, on the figure 4.9. This graphic was created with the help of three tables representing performance of each individual in the lower difficulty level.

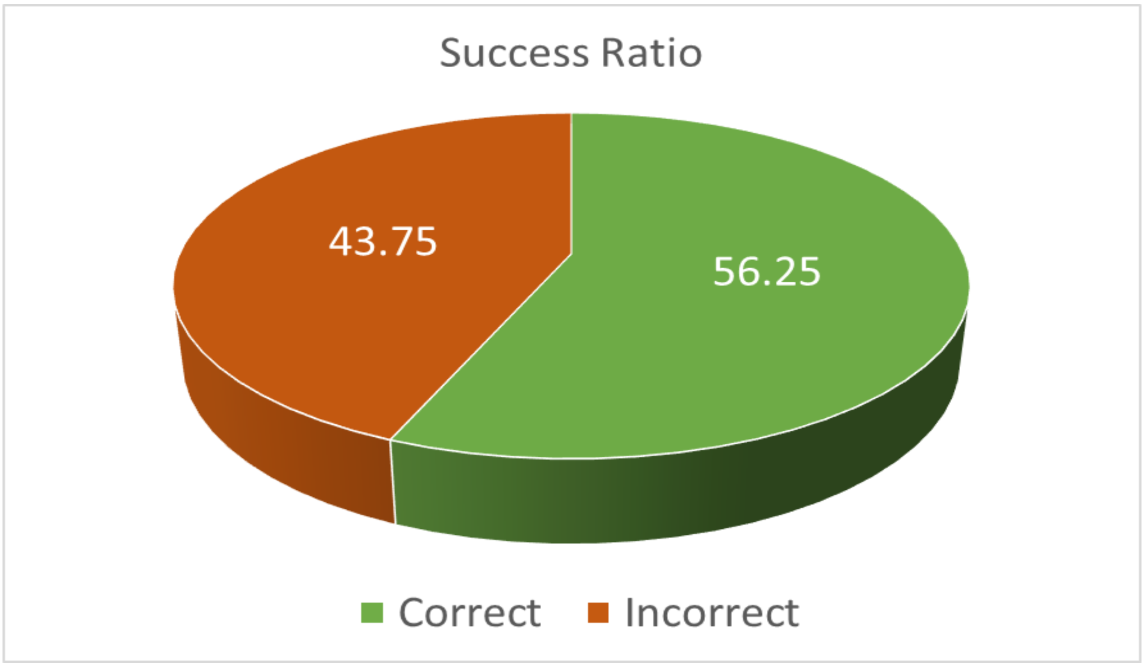


Figure 4.9 - Success ration graphic of the three subjects within the same difficulty level

It is easily verified that the ratio is positive, with 56.25% correct answers against 43.75% wrong answers. Being the lowest difficulty level means, it's important that the success rate is positive by a certain margin, but not that easy that would lead to a lack of interest from the players. From this figure, it's confirmed that the game lowest difficulty is suitable for its target audience.

In addition to the previous graphic, one more were created, presented in figure 4.10. The graphic was created from the data presented on tables 4.7 and 4.8. This data was gathered from two sessions of two separate individual which focused only on one part of the game.

Table 4.7 - Results obtained by Subject 3 in Difficulty 1 Location Section

LEVEL	QUESTION	ANSWERS	TIME (seconds)
1	BELOW	BELOW	4.954
2	UP	INSIDE	6.327
2	UP	SIDEWAYS	3.534
2	UP	INSIDE	5.833
2	UP	INSIDE	2.701
2	UP	INSIDE	2.9
2	UP	INSIDE	3.1
2	UP	INSIDE	2.033
2	UP	INSIDE	2.4
2	UP	INSIDE	2.3
2	UP	SIDEWAYS	5.034
2	UP	UP	2.3
3	SIDEWAYS	SIDEWAYS	10.254
4	INSIDE	INSIDE	7.387
5	UP	UP	5.987
6	SIDEWAYS	UP	5.753
6	SIDEWAYS	SIDEWAYS	3.1
Total	6	17	75.897

Table 4.8 - Results obtained by Subject 2 in Difficulty 1 Location Section

LEVEL	QUESTION	ANSWERS	TIME (seconds)
1	UP	SIDE	15.086
1	UP	SIDE	2.934
1	UP	SIDE	1.566
1	UP	UP	2.467
2	DOWN	INSIDE	7.719
2	DOWN	DOWN	2.8
3	UP	DOWN	5.554
3	UP	INSIDE	5.633
3	UP	INSIDE	5.167
3	UP	UP	3.838
4	SIDE	SIDE	5.587
5	UP	UP	5.153
6	DOWN	DOWN	16.32
Total	6	13	79.824

From the tables 4.7 and 4.8, some observations can be made. Firstly, the Subject 2 took more time than Subject 3 to conclude the stage but needed less tries. This fact leads to figure 4.10.

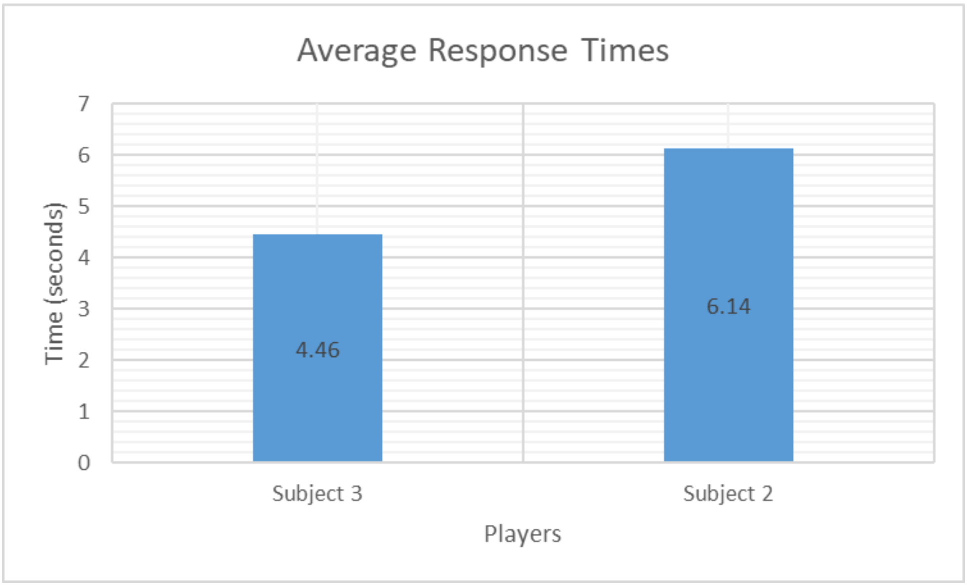


Figure 4.10 - Average Response Times of Subject 2 and 3

In figure 4.10, it's showcased the average response times for both players. As previously concluded the average response time is higher for Subject 2 since more time was spent while providing less answers. This can hint to some problems in the implementation of the game but after a close analysis to both tables, it's clear that the problem lays elsewhere. From table 4.7 it's obvious that Subject 3 has a misconception with the word "UP", leading to multiple incorrect answers since all the answers were given in a reasonable amount of time (over 2 seconds), meaning they weren't the product of a random try, without any thought or reason.

One important detail to mention, is the fact that from the tables created from the file, it's possible to determine which elements cause the most problems to the player. This type of information about the player is extremely valuable for the therapists and doctors. As previously stated this information is all stored inside a file so that it can be obtained by these same therapists and doctors.

In short, the results were excellent, proving the game is well designed, as the ratios are positive and the players remain interested during the play session.

In addition, the game provides relevant and crucial information for the therapists and doctors.

Besides providing excellent results, it also contributed with one of the most unique, special and enriching experiences. It happened during the last phase of the **Evaluation** stage, due to one of the purest reactions expressed by one of the children that tested the game. This child smiled and laughed extensively when performing well in-game and felt every failure just as passionately. This event happened numerous times, leaving a feeling of joy to fill the room, and showing how successful the game really was.

After verifying the game performs well, it was deployed to the institution under the physic form of a flash drive with a apk file, installable in Android devices and by their installation in some of their tablets/smart devices. Similarly to the "Bê-à-Bá"game, this game also is available in the official website (http://stb.uninova.pt), under the section attributed to the "Falar Pelos Cotovelos" project and its deployment to the Google Play Store is also in the works, to insure that more rooms can be filled with joy.

4.4. Quiz

All the effort put into developing and creating this model would be in vain, if the main target developers would not put it into practice. For this reason, understanding what these developers think of the DGDM is crucial to insure the continuation and implementation of this model. If the DGDM isn't well received it won't be put into practice and therefore, has virtually no value.

The solution found to assess this issue was to develop a quiz. The questions within this quiz aimed at knowing what was the overall opinion on the model and which could be the vulnerabilities of model.

After the development, the quiz was presented to the developers. In this case, the developers, the main targets of these model, are the all members of the STB initiative. Almost all members of the STB were subject to this quiz, at a total of 7 persons, this way the quiz approached various members, in distinct stages of their project development, both at the beginning and end, therefore, reflecting the opinion of developers in the both ends of the spectrum, unexperienced and veterans, as well as, gathering the most answers possible. Before presenting the quiz, the DGDM was explained in detail in a ten to twenty minutes session, to guaranty that all members were aware and reminded of all the intricacies of the model.

The quiz is composed of four main questions, one sub-question and one suggestions section. The reason for the existence of a suggestions section is clear, however the same doesn't apply to the sub-question. From the answers gathered from the quiz, the following four charts were created.

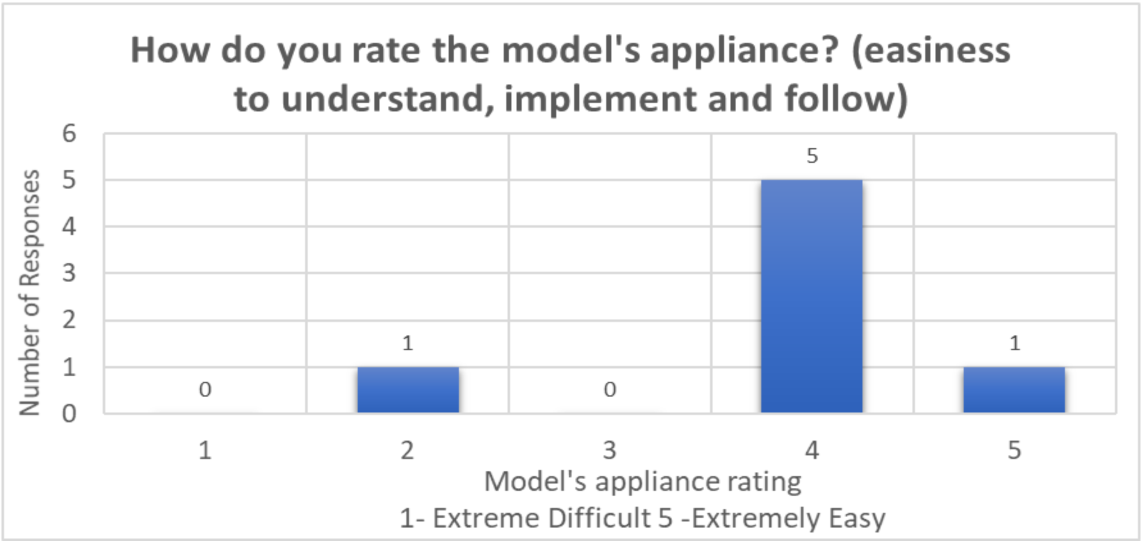


Figure 4.11 - Answers gathered from the first main question of the quiz

In the figure 4.11, presented above it's possible to see both the question as the answers gathered. The subjects were asked to rate with a maximum of 5 values and a minimum of 1 value, how easy they think it is to understand, implement and follow the DGDM, the higher the rating the easiest it would be. From the figure is also possible to conclude that it is regarded as easy to understand, implement and follow but not too easy so that it can be used without concerns, laying in the balance as desired.

The results obtained for the second question are presented in the figure 4.12. This time, the subjects were asked how willing they were to follow the model, being the maximum value of 5, absolutely willingly and 1 the minimum value, never willingly. As observed in the figure, the results were split into the two top values, which suggests that the DGDM is seen as an asset but it can still be improved upon.

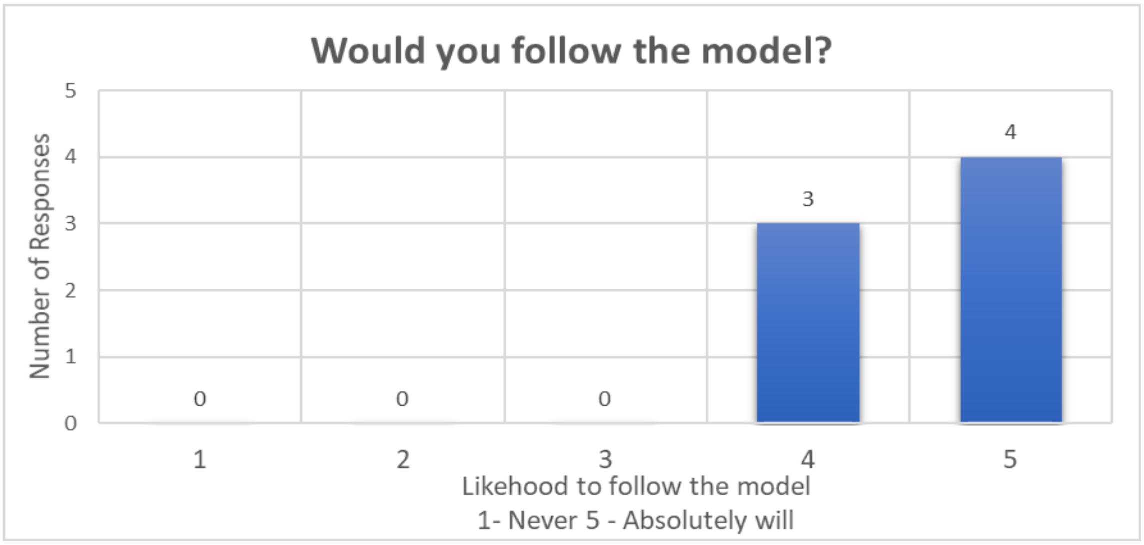


Figure 4.12 – Answers gathered from the second main question of the quiz

The third question of the quiz is accompanied by a sub-question. This sub-question is a result of a positive answer to this third main question and aims at grasping a better understanding of the decisions that led to this positive answer. The main question in cause is "Would you change/add any stage or activity?". If a positive answer is received the subject is asked to follow-up on that answer through the sub-question, by providing information why and how it would proceed. From the results to the third question, showed in the figure 4.13 below, it's possible to verify that this sub-question proved futile as no positive answer was given. The absence of positive results, in addition to the results obtained in the previous question, leads to the conclusions that either the minor improvements that can potentially be performed to the model, aren't simple and/or of immediate implementation, thus requiring a more indepth study or that there aren't any improvements to be made. Both conclusions imply that the DGDM has immense value, and should be implemented and subject to study.

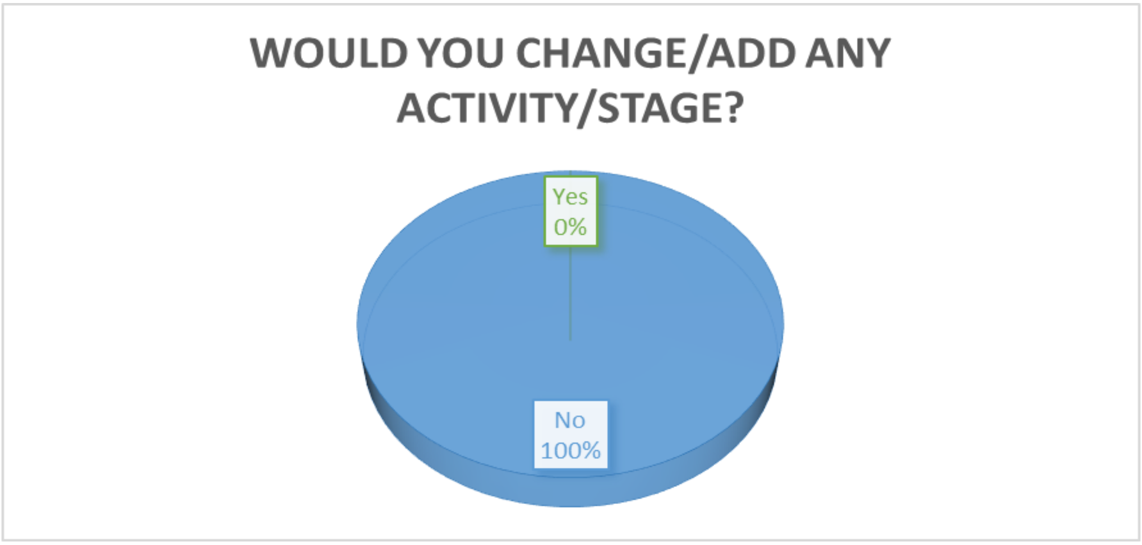


Figure 4.13 - Answers gathered from the third main question of the quiz

Finally, the results of the last question can be observed in the final figure 4.14. The last question approached the subjects, in order to obtain a rating for the DGDM. From the results gathered, it's clear that it is seen with great value since most answers are on the two highest rating values, further confirming the results obtained to the previous question. Similarly, to the previous question it's also possible to verify that there might be some minor improvements left to perform, due to the most frequent answer being the rating 4 and not the highest value.

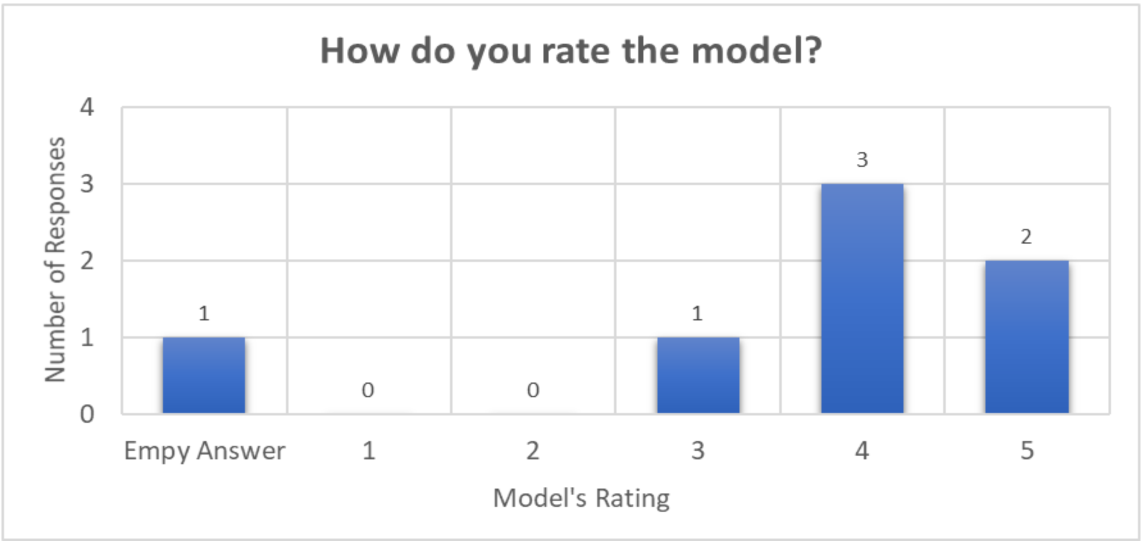


Figure 4.14 - Answers gathered from the forth main question of the quiz

The last section of the quiz was a suggestion box that provided almost no results, only one answer. The suggestion was the development of a manual or some guidelines for the development. These would be very generic as to serve the purpose of reminders of good practices and some band aids for each activity. This suggestion has immense value, as it could be the remaining factor to push the DGDM forward.

The analysis of this quiz and its results, leads to the conclusion, that the main targets of this model, the developers within the STB initiative regard the DGDM as an asset, to be implemented willingly, that will help in the development of games, but further study of the model, only achievable from its implementation into more projects, is also required.

5

Conclusion

5.1. Conclusion

The development of this project allowed the observation of how scarce the usage of digital games towards social causes is. The huge industry developers focus only in developing digital games for entertainment purposes, leaving only a few to develop games for more social causes. This brands a bad image for digital games, in the scientific community eyes, which by consequence creates a lack of development and knowledge within this area of expertise. This lack of content was the first experienced difficulty during the project. The rarity of similar models for game development meant that the research approach had to change radically to focus on a more developed but similar area of expertise. This similar area offered immense contributions, due to its advanced development and great background, leading to the incorporation of multiple solutions into the proposed model.

Another difficulty was the conciliation of all the limitations, from both institutions and students. The model needed to be fast, simple and efficient, but also detailed, agile and flexible. Finding the balance required multiple versions of the model and the selection of which activities were core and indispensable.

The developed model (DGDM) impacts positively the development of digital games towards social causes, providing the project managers and developers a useful tool under the form of a model, which leads the development away from possible bad practices. By following the DGDM, the developers are offered a guiding hand which prevents bad practices, promotes the development of meaningful games and provides solutions to avoid hitting a full stop in the development.

The development of a support application greatly increased the potential of the DGDM. The Game Development Support App facilitates the development of digital games, while diminishing the difficulty of implementing and following the model, by providing tools to ease up two of the most important characteristics of developed model, documentation and supervision. The development of the GDSA didn't pose much trouble overall, being the only difficulties worth noting, the struggle with authenticating the user into the Google API and the lack of design ideas for a friendly and intuitive user-interface.

Finally, the objectives set in chapter 3 were achieved. However, this wasn't done effortless as some concessions have been made. For example, to allow supervision and insurances the model required vast amounts of documentation. This is costly in terms of time and resources, while also being a burden to developers and supervisors, which could be solved by developing standardized resources to aid in its supervision but this adds layers of complexity to the model, breaking another of the DGDM's requirements. The result of this concessions is visible through the quiz, which hints into the need for a few adjustments depending on the project in hands and its developers, some might require a stronger component while others don't. Overall, the proposed model successfully achieves its goals.

5.2. Contributions

Firstly, I would like to point out all the contributions given by Centro Diferenças towards the development of both "Bê-à-Bá" and "Falar Pelos Cotovelos".

Secondly, this dissertation is also accompanied by a scientific article called "Digital Games' Development Model"[31], published in European Alliance for Innovation Endorsed Transactions on Serious Games journal.

Finally, this document is complemented by a user guide[32], containing direct guidelines and suggestions on how to follow, implement and apply the DGDM to further success.

5.3. Future Works

As pointed out in the 5.1 section, it's possible to improve both the model as the application in several aspects with multiple goals.

Starting with DGDM, a multitude of improvements can be made. The creation of resources targeting developers with information, guidelines and suggestions regarding each of the stages and activity would be a great asset for both the STB initiative as well as the model itself. Since the developers would gain a valuable tool from which they could get the necessary support, learn more about the model, clarify some doubts and even share/trade experiences contributing to both the STB initiative and the model. This resource could be shared between model and application, due to their clear relation. For instance, it could take the form of an official website or forum or PowerPoint presentation, or something more physical like flyers or posters. An early version of this resource was made, as previously stated but a lot of improvements can still be made.

In addition to the improvement already stated, a detailed study about marketing, online business models and deployment tools would also contribute plenty, particularly for the deployment stage of the DGDM. From the study, it would be possible to extract recommendations for the developer, regarding how to market the game as well as how to deploy and where. These recommendations would be an asset not only for the STB initiative but also for developers outside of this initiative.

Similarly, to the previous suggestion, the creation/research of several guidelines for some of the activities or stages within the model, would also be very benefic.

Moving on to the application, several improvements can also be performed. The development of an iOS counter-part of the already existing Android application would increase the amount of people that could profit from the application.

An improvement can also be made in the overall presentation of the game, turning it more appealing and intuitive to use.

Lastly, the number of options provided to the user is very limited, regarding which tools to use within the application. What this means is that the user is forced to use both the Google Account, Drive and Calendar. This can be a big turndown when a user doesn’t have access or simply prefers to use similar tools. Allowing the user to choose which tools they want to work with would be an immense value that few applications currently possess.

6

References

[1] "Social Tech Booster." [Online]. Available: http://stb.uninova.pt/. [Accessed: 22-Feb-2017].

[2] C. Santos, T. Cardoso, V. Santos, and J. Barata, "Transferência de Tecnologia para Causas Sociais através de VideoJogos," in *SciTecIN'15 / EPCG2015*, 2015.

[3] ESA, "Essential facts about computer and video game industry," vol. 2016, pp. 1–3, 2016.

[4] R. S. Pressman, *Software Engineering A Practitioner's Approach 7th Ed - Roger S. Pressman*. 2009.

[5] D. Hofstrand, "Product Life Cycle," no. August, p. 1, 2007.

[6] T. Levitt, "EXPLOIT THE PRODUCT LIFE CYCLE," *Harvard Bussiness Rev.*, vol. 43, pp. 81–94, 1965.

[7] CasualKIT, "Casual Game's Life Cycle." [Online]. Available: https://www.casualkit.com/blog/88666256397. [Accessed: 22-Feb-2017].

[8] B. Boehm, "A Spiral Model of Software Development," *IEEE Comput.*, no. 1, 1988.

[9] C. Ghezzi, M. Jazayeri, and D. Mandrioli, *Fundamentals of Software Engineering*, vol. 5961. 2002.

[10] P. Kemp and P. Smith, "Waterfall Model," 2013. [Online]. Available: https://en.wikipedia.org/wiki/Waterfall_model. [Accessed: 22-Feb-2017].

[11] J. Partogi, "Information Systems Engineering," 2015. [Online]. Available: https://informationsystemsengineering.wordpress.com/. [Accessed: 22-Feb-2017].

[12] S. Balaji, "Waterfall vs v-model vs agile : A comparative study on SDLC," *WATERFALL Vs V-MODEL Vs Agil. A Comp. STUDY SDLC*, vol. 2, no. 1, pp. 26–30, 2012.

[13] S. D. Tripp and B. Bichelmeyer, "Rapid protoyping: An alternative instructional design strategy," *Educ. Technol. Res. Dev.*, vol. 38, no. 1, p. 31=44, 1990.

[14] Centers for Medicare & Medicaid Services, "Selecting a development approach," *Centers Medicare Medicaid Serv.*, pp. 1–10, 2008.

[15] Basicofy, "Incremental Model." [Online]. Available: http://www.basicofy.com/wp-content/uploads/2016/09/Untitled362.png. [Accessed: 10-Feb-2017].

[16] H. Kniberg, "Spotify engineering culture | Labs," 2014. [Online]. Available: https://labs.spotify.com/2014/03/27/spotify-engineering-culture-part-1/. [Accessed: 13-Jan-2017].

[17] R. C. Martin, "Agile Software Development, Principles, Patterns, and Practices," *Book*. p. 529, 2002.

[18] K. Beck, *Extreme Programming Explained , Second Edition*. 2004.

[19] "Inscrições abertas para torneio eSports | Site oficial do Sporting Clube de Portugal." [Online]. Available: http://www.sporting.pt/pt/noticias/clube/noticias/2016-09-07/inscricoes-abertas-para-torneio-esports. [Accessed: 23-Feb-2017].

[20] "FC Schalke 04 takes over eSports team 'Elements' - News - Schalke04.de." [Online]. Available: http://www.schalke04.de/en/news/160516/page/2234--86-86-.html. [Accessed: 23-Feb-2017].

[21] "VfL Wolfsburg | E-Sport." [Online]. Available: https://www.vfl-wolfsburg.de/en/info/news/digital-services/e-sport.html. [Accessed: 23-Feb-2017].

[22] "Beşiktaş Gymnastic Club." [Online]. Available: http://www.bjk.com.tr/tr/haber/61136/. [Accessed: 23-Feb-2017].

[23] K. Salen and E. Zimmerman, "Rules of Play: Game Design Fundamentals," *Nihon Ronen Igakkai Zasshi.*, p. 672, 2004.

[24] J. Schell, *The Art of Game Design, Second Edition*. 2014.

[25] C. Crawford, *The Art of Computer Game Design: Reflections of a Master Game Designer*. 1984.

[26] C. M. Kanode and H. M. Haddad, "Software Engineering Challenges in Game Development," pp. 260–265, 2009.

[27] F. Petrillo and M. Pimenta, "What went wrong ? A survey of problems in

game development What Went Wrong ? A Survey of Problems in Game Development," no. February, 2009.

[28] L. Goasduff and A. Ann Forni, "Gartner Says Worldwide Sales of Smartphones Grew 7 Percent in the Fourth Quarter of 2016." [Online]. Available: http://www.gartner.com/newsroom/id/3609817. [Accessed: 12-Jul-2017].

[29] L. Espada, "Bê-à-Bá," Universidade Nova de Lisboa - Faculdade de Ciências e Tecnologia, 2017.

[30] P. Leandro, "Falar Pelos Cotovelos," Universidade Nova de Lisboa - Faculdade de Ciências e Tecnologia, 2017.

[31] T. Cardoso, J. Sousa, and J. Barata, "Digital Games' Development Model," *EAI Endorsed Trans. Game-Based Learn.*, vol. 4, no. 12, p. 153399, Dec. 2017.

[32] J. Sousa, "Digital Games' Development Model," 2018. [Online]. Available: https://drive.google.com/file/d/1RayCOOp-wZ_15zaANAb3BXZ5VptoIaIj/view?usp=sharing. [Accessed: 01-Jan-2018].

Annexes

Annex 1

The Digital Games' Development Model

Prototype Design Form – Design Stage

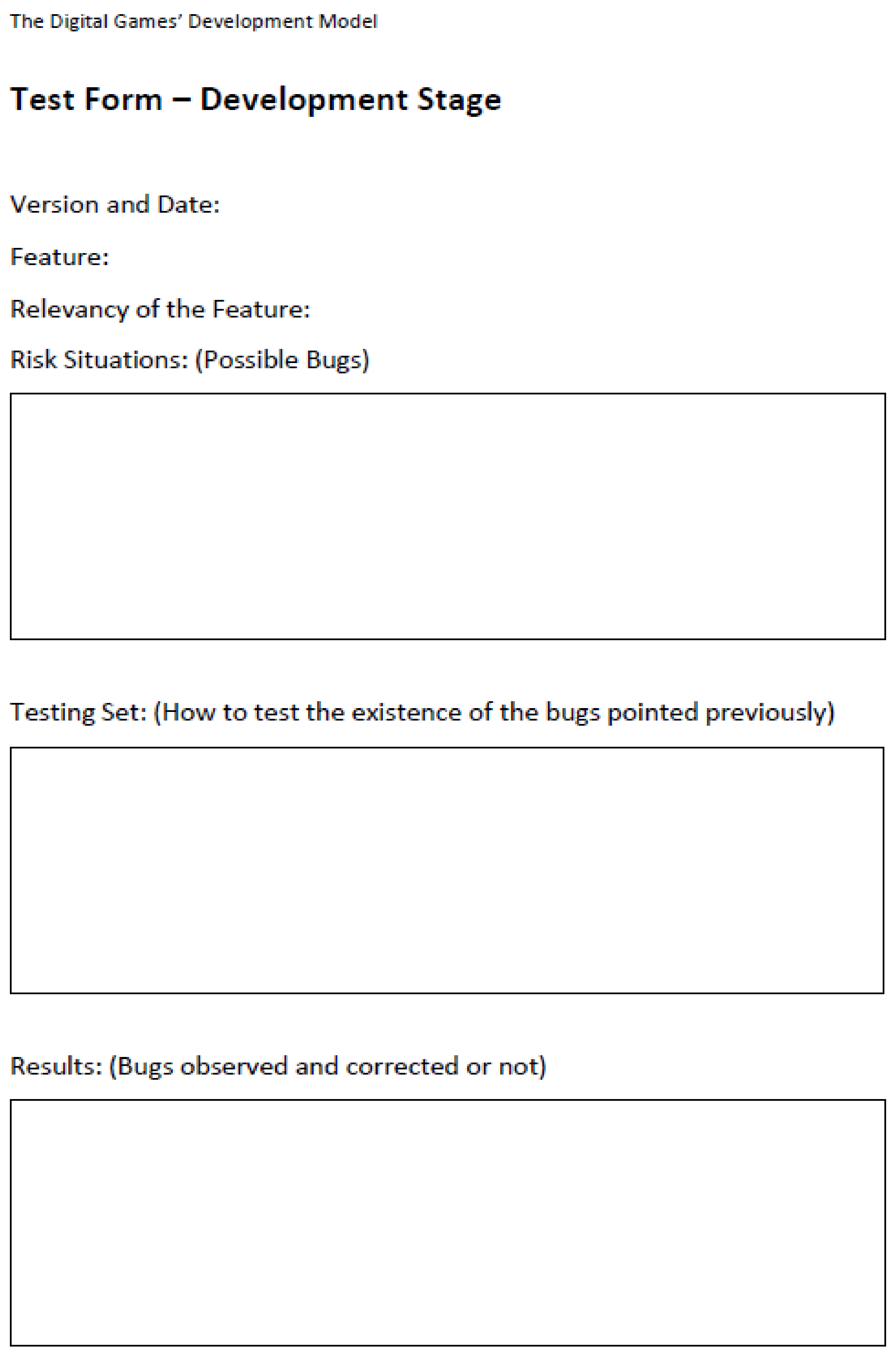
Version and Date:

Screen Name:

Screen Layout:

Functionalities:(Before and After Screens, Buttons, etc...)

Annex 2



The Digital Games' Development Model

Test Form – Development Stage

Version and Date:

Feature:

Relevancy of the Feature:

Risk Situations: (Possible Bugs)

Testing Set: (How to test the existence of the bugs pointed previously)

Results: (Bugs observed and corrected or not)

Annex 3

Art Director Worksheet

Main Objective:

- **Define the art concept of the game**
- **Define the style of the world and key elements of the game**
- **Describe in detail the main character of the game**

1. Define the presentational style of the game.
(cartoon, loonyToon, pencil drawing, photorealistic, manga...)

---

2. Define the aesthetic sytle of the game world.
(1950's science fiction, Medival, Greak, Egyptian...)

---

3. Define the lightning effect that can influence the gameplay, if there are any
(fog, candle, moonlight, fire, lamps, mirrows...)

---

4. Define the color palette that will be use
(dark+blue= night, red+black=hell, bright primary colores=children...)

---

5. Describe how people/creatures look like, if there are any. Describe their role through their appearance
(young, sexy, ugly, friendly, nervous, evil, clampsy...)

---

6. Describe how the user interface elements support the style of the game
(buttons, clocks, lights, pointers, type fonts, sliders...)

---

7. Draw the box of the game as it will look like in a shop

8. Drawing Section
Make sketches of key elements. (player avatar if is visible, enemies, scenario, buildings, weapons, tools, health kids...)

Annex 4

Game Designer Worksheet

Main Objective:
Describe the Gameplay Modes

A. Primary Gameplay Mode.
Place where the player will spend most of the time

1. Setting

2. Perspective (eg. 1st person, 3rd person)

3. Interaction Model (eg. Avatar, omnipresent)

4. Challenges

5. Actions

6. Mechanics

B. Internal Economy

1 Define the main resources of the game, if there are any (points, health, money, time, ...)

____

____

____

Complete the following list with the elements of the internal economy

Resource	**How it is produced**	**How it is consumed**

2. How would you balance the resources to change the difficulty

____

____

____

3. Describe how is the positive feedback controlled, if there is any.

____

____

____

C. Game Level

Describe a level that represents in the best way your game.

1. **Work with the level designer so he/she can complete the level worksheet**
   . Analyze the level layout
   . Describe the starting condition of the level
   . Does the game has a single victory or loss condition ?

Describe the victory or loss condition, if there is one

____

____

____

Describe the main challenges and actions

____

____

____

____

Annex 5

Lead Designer's Section

Game Concept:
Write down the game concept, maximum 5 lines

Story & Narrative

Backstory (if necessary)
Describe what happens before the game starts

Beginning stage of the game

Middle stage of the game

Ending stage of the game

Epilogue stage of the game

Player Description

Gameplay Killer Features

Annex 6

Level Designer Worksheet

Main Objective:

- **Fluid communication with the Game Designer and User Interface Designer**
- **Create different escenarios that can challenge the player in different ways**

LEVEL DEFINITION

1. Level Name ____

2. Level Description
(In one phrase describe the level. Ex: an industrial factory)

____

____

3. Victory condition, if any

____

4. Loss Condition, if any

____

5. Starting conditions

____

6. Key elements that makes this level different from others

____

____

____

7. Level description
(describe and position the different elements and NPC that makes part of the gameplay)

____

____

____

8. Draw the Layout of the level , elements and NPC

Annex 7

User Interface Designer Worksheet

Main Objective:

- **Define the main actions the player has to do and map the to the main input control the game will use (Joysitck, mouse, keyboard,...)**
- **Determine the lay out of the screen of the primary gameplay mode**

A. Input Control

Complete the section that best match your game control
(keyboard, mouse, gamepad, other)

1. Mouse

Define what moving around the mouse does.

____

Define what does the Left Button do.

____

Define what does the Right Button do.

____

2. Gampad

Define what does the directional pad do most of the time

____

Describe function of other buttons if necessary. # of additional buttons:______

Function Button 1:____

Function Button 2:____

Function Button 3:____

Function Button 4:____

3. Keyboard

4. Other

B. Screen Layout:

1.	**Comments**
2.	**Comments**
3.	**Comments**
4.	**Comments**