Universidade do Minho
Escola de Engenharia

Tiago João Dias Barbosa

Serious game framework for statistics learning in higher education

Dissertação de Mestrado

Mestrado Integrado em Engenharia Eletrónica Industrial e Computadores

Trabalho efetuado sob a orientação do

Professor Doutor Sérgio Adriano Fernandes Lopes

Professora Celina Maria Godinho da Silva Pinto Leão

Novembro 2018

Acknowledgement

Foremost, I would like to thank my dissertation advisor Prof. Sérgio Lopes and co-advisor Prof. Celina Pinto Leão, for the continuous guidance, patience and motivation, as well as the constant readiness for any support when I had any questions about my research or writing.

Besides my advisors, I would like to thank the rest of my dissertation committee: Prof. Filomena Soares and Prof. Vítor Carvalho, for the reviews of papers and insightful comments.

I would like to share my gratitude towards the University of Minho, specially the school of engineering, for the shared knowledge and a special thanks to all my friends for the shared experiences and fun times spent together.

Finally, I must thank my family, especially my parents, for providing me with unshaken support and never-ending encouragement throughout all my years of study and the writing of this dissertation. Nothing would have been possible without them.

Thank you!

RESUMO

Nos últimos anos, os videojogos têm crescido em popularidade e cada vez mais têm vindo a reforçar a sua posição na indústria do entretenimento. Isto é refletido no aumento do rendimento da indústria dos videojogos e a expansão notável do seu público-alvo. Nos dias de hoje, os videojogos já não são apreciados apenas pelas crianças ou adolescentes, sendo que os adultos têm vindo cada vez mais a dedicar parte do seu tempo livre a jogar. Esta popularidade tem levado a um aumento de discussões acerca da incorporação dos videojogos nas salas de aulas, referindo-se aos mesmos como jogos sérios e usá-los como estratégias de aprendizagem.

Através de simulações de situações práticas do dia-a-dia, ou situações críticas num ambiente industrial, os jogos sérios permitem o desenvolvimento de várias capacidades, tais como, capacidade de observação e raciocínio. Ao mesmo tempo, estes jogos são desenvolvidos de modo a transmitir conhecimento para os alunos/utilizadores e funcionam como uma ferramenta de suporte ao processo de aprendizagem, aplicado a várias arear disciplinares e desenvolvendo competências específicas para cada caso.

O propósito principal desta dissertação passa pelo desenvolvimento da *framework* de um jogo sério, que disponibiliza ferramentas e informação suficiente para permite o futuro desenvolvimento de um jogo sério completo para ser usado como uma ferramenta de aprendizagem, que promove e apresenta a usabilidade de conceitos de estatística no dia-a-dia e no exercício de tomada de decisões.

A ideia proposta é a criação de um ambiente virtual desafiante (investigação/procura de provas) onde as pistas estão contextualizadas na área da estatística, através da prática de conhecimentos de probabilidade, intervalos de confiança e testes de hipóteses. Com isto, as atividades para aprendizagem passam por tomadas de decisão baseadas em subáreas da estatística.

Os resultados deste trabalho incluem a implementação das primeiras partes do jogo e o desenho do jogo envolvendo todos os aspetos necessários para futuras implementações. O trabalho desenvolvido é uma base sólida para futuros trabalhos, estando já implementadas a estrutura e mecânicas do jogo, a arquitetura principal do software e o início da narrativa, servindo como exemplos para outros elementos estruturais (cenas, cenários, minijogos, etc.).

Palavras-Chave: Jogo Sérios, EDUCAÇÃO SUPERIOR, ESTATÍSTICA, NARRATIVA, INTERAÇÃO ADAPTATIVA.

ABSTRACT

In recent years, video games have been growing in popularity and made their stand in the entertainment industry. This is reflected by the revenue increase in the video game industry and by the noticeable expansion of their target audience. Indeed, nowadays, video games are not only played by kids and teenagers, but adults are also spending their free time playing. This popularity is constantly leading to talks about incorporating video games in classrooms, referred to as serious games and using them as a learning strategy.

Through simulations of day-to-day practical situations, or critical ones on a business environment, video games allow the development of capabilities, such as observation and reasoning skills. At the same time, Serious Games are strategically developed as a way to transmit knowledge to the users/students, and work as a tool in the learning process, applied to a variety of disciplinary areas, developing specific skills for each case.

The main purpose of this dissertation is to develop a serious game framework, to provide a solid foundation for the creation of a full-fledged serious game to be used as an educational tool, to promote and present the day-to-day applicability of statistic concepts and in the exercise of decision making.

The proposed idea is to create a challenging environment (investigation/pursuit of clues) where clues are contextualized in the statistics area through the practice of probability knowledge, confidence intervals and hypotheses tests. With this, the learning activities include decision making based on statistics' subareas.

The results of this work include an implementation of the first parts of the game and a game design featuring all aspects necessary to start further implementations. The work developed is a solid foundation for further development of the game. Specifically, the game structure and mechanics are defined, the core software architecture is implemented, as well as the start of the storyboard, serving as an example for other structural elements (scenes, dialogues, mini-games, etc.).

KEYWORDS: SERIOUS GAMES, HIGHER EDUCATION, STATISTICS, STORYTELLING, ADAPTIVE INTERACTION.

CONTENTS

List of Figures

List of Tables

ABBREVIATIONS

SG	**S**erious **G**ames
LMGM	**L**earning **M**echanics – **G**ame **M**echanics
DPE	**D**esign, **P**lay, **E**xperience
RETAIN	**R**elevance, **E**ngagement, **T**ranslation, **A**ssimilation, **I**mmersion, **N**aturalization
NPC	**N**on-**P**layer **C**haracter
SBSG	**S**and **B**ox **S**erious **G**ames
UI	**U**ser **I**nterface
GO	**G**ame**O**bjects

1. INTRODUCTION

With the digital boom in modern society, there is an immensely vast new variety of methods for teaching and learning, even more in recent years, with the increased usage of mobile devices, that brings a lot of tools and information directly to the hands of users/students, wherever they are [1].

However, one of the biggest challenges in education remains to find interesting and engaging techniques or instruments, that increase and quantify levels of engagement from students. Studies show that learning involves high levels of attention, as well as many other cognitive processes, like memory, motivation, excitement and communication, and it is often proved that "hands-on experience" is one of the most effective and quickest method to target all those processes [2]. This way, computer simulations and serious games (SGs) can be seen as a good mixer of both area of application and learning processes, that not only allow students to engage and experience real-life scenarios, in a safe and cheap environment, but it also allows them to quickly understand how to apply their knowledge to solve working problems [3].

Kevin Corti [4] describes serious games as a type of computer games, where its virtual environment is developed in order to simulate real-life, practical scenarios, and present problems, that increase the users´ interest and motivation to solve them [5]. However, there is not a consensus for a unique and single definition of the term, besides serious games are not developed for entertainment only, but as a tool for learning, that takes advantage of game elements, like interactivity, a set of agreed rules and a clear goal [6]. Balancing the level of entertainment and gameplay elements with the educational content of the game is a crucial process and not a trivial one, since it requires a deep understanding on how these types of games should be designed.

Within the scientific and academic communities, there is no doubt that serious games represent an efficient and engaging tool, used for improvements in the learning process [7]–[12]. Serious games are developed to be used as a learning tool in wide areas and different levels of knowledge [10], being STEM field (Science, Technology, Engineering, and Mathematics) and health the most common. Higher education has also proved to be another area of concern in the development of serious games, presenting positive results and contributions, especially in areas where students experience more difficulties [13]–[15]. Moreover, serious games benefit skill's development, namely communication, creativity and adaptability competences [13], rewarding users´ involvement and providing feedback on their performance, while at the same time surrounding them with "rich visuals and spatial aesthetics" [16],

seducing them into completing a set of tasks, solving many different problems and spending a lot of time and energy playing the game.

Despite the positive features mentioned above, researchers recognize that a lot of improvements remain to be made, namely in the design phase, in order to not compromise the success of serious games as a learning tool and encouraging the interest of students in specific areas of knowledge [8], [12], [15].

Following these trends and ideas, this dissertation presents a serious game framework on Statistics in higher education, contributing with a solid foundation for future development of a complete serious game that helps the comprehension/learning of statistics by replicating real world situations in a virtual environment, showing the applicability and relevance of statistical analysis and decision making, in.an area where engineering students fail and it is still considered by many as not relevant to their courses [5].

1.1 Motivation

A large number of published papers, works, general opinions articles and/or game sites related to serious games (SG) can be easily found. By simply googling "serious games" one can find about 8 million results. In order to address the main subject of interest, specific terms like "education" and "statistics" were added, reducing the matches to nearly 200 thousand. Narrowing the search to the main area of interest of the present work, adding AND "higher" AND "engineering" to the search, resulted in nearly 131 thousand matches. Nonetheless, these values show the adhesion and use of these games as an alternative and complementary tool in the teaching/learning process.

However, this extensive research proved unsuccessful in the task of finding a previously developed SG for statistics learning in higher education. The only similar work found was a literature review of games, animations and simulations to teach statistics [9], leading to the conclusion that there is a lack of serious games in this area.

With this gap in the market and the need to increase students' motivation towards statistics, as evidenced in the work of Leão (2016) [17], where the author mentions the difficulties students tend to have with statistics, mainly due to non-cognitive factors, such as negative attitudes towards the subject and its complexity, it is important to create new and interesting tools and methods that emphasize the practical component of statistics and motivate students to study it.

By developing a SG, not only is this issue being targeted, but at the same time it also explores the pedagogical component that serious games provide.

1.2 Objectives

The main task of this dissertation is to develop a Serious Game framework that clearly defines the Statistics' educational contents and structure to be addressed, implements the main structure and theme of the story of the game and, on a more technical standpoint, implements the game's basic mechanics that provide a solid foundation for future work.

Before beginning the development of the SG, it is important to define the core ideas of the approach to be followed. Since the main objective is to engage the users until the end of the game, the proposed SG is going to focus on four essential requirements to maximize users' engagement: a) allow relevant and personal associations with real-life elements or situations, by developing a story; b) give easy access to the game with adjusting levels of difficulty and tutorials; c) create vivid graphics in order to immerse users into the virtual world and d) measure and provide feedback of players' performance [2].

1.3 Document Organization

This dissertation is structured in seven sections:

- Section one is the introduction of the theme, presenting the main problem and the motivation to tackle it, as well as all the objectives to reach.
- Section two presents the different Serious Games frameworks, starting with an introduction to three education principles, followed by three Serious Games frameworks.
- Section three details the two types of Serious Games that best describe the Serious Game related to this dissertation.
- Section four is the presentation of the design of the game, detailing the game's main idea, the educational concepts tackled, the main storyboard and an overview over the game's mechanics.
- Section five details the implementation of the software created.
- Section six presents the results obtained, by detailing a scene of the game and how it flows.
- Section seven is the conclusion and possible future work.

2. SERIOUS GAMES FRAMEWORKS

Robert Gagne and James Keller declare that many serious games fail due to the inability of developers to support the serious game design on learning theories [18]. Although players are entertained while playing the game, sometimes the learning outcome is nearly inexistent, which inspired the creation of frameworks and methodologies that aide developers to integrate educational content into their games.

However, due to the number of different frameworks and methodologies, the task of choosing a specific one that fits a particular game often proves to be difficult.

This effort instigated Slimani Abdelale et al. [19] to compare and classify the different methodologies, allowing for a more informed decision when picking one for the analyses and/or design of serious games. To accomplish this, the authors defined a variety of components and described how each methodology targeted each one of them: category(design purpose of the game), level (methodology relates to low level components, high level intentions or the relation between both), layers (set of disassembled requirements), relationship, purpose (methodology purpose), process (design process), applications (target area of the methodology), player (player considerations), user (classify methodologies based on users specifications) and evaluation (attributes related to design validation).

With this, Slimani Abdelale et al. [19] provide a table with the classification and comparison of game design methodologies, which proved crucial when choosing the ones that better fitted and supported the development of the SG related to this dissertation (figure 1). The popularity of each methodology also weighted upon this decision.

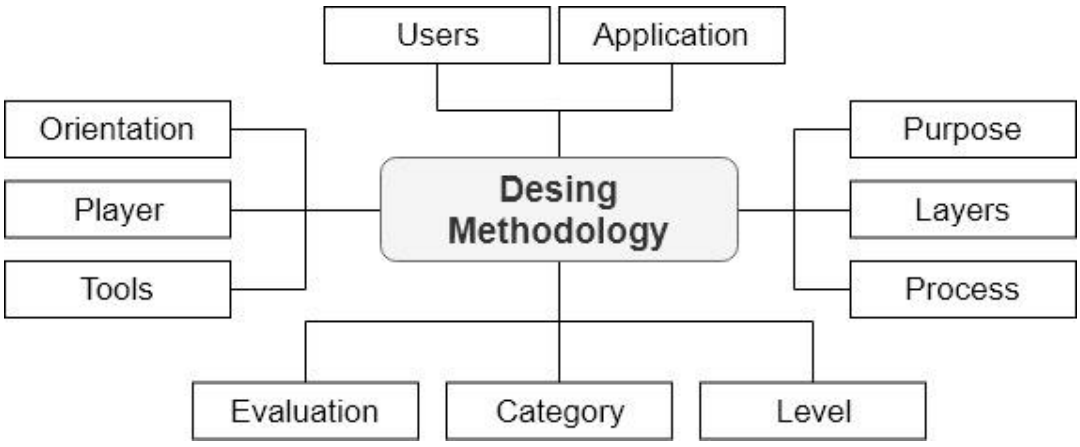

Figure 1: Classification of game design methodologies

For this SG, and for the category subject, the methodology should focus on education, with a variety of layers like pedagogy, story, gameplay, game world, assessment and game structure, with a knowledge and goals evaluation purpose, accomplished by a teacher or domain expert.

After perusing through all the possibilities, it was possible to select three methodologies: Learning Mechanics – Game Mechanics (LM-GM) framework, Design, Play, Experience (DPE) framework and the Relevance, Engagement, Translation, Assimilation, Immersion, Naturalization (RETAIN).

2.1 Educational Principles Overview

As discussed in previous topics, motivation is a central factor when learning, and thus it is crucial to provide players with an adaptive and scalable level of challenge, that matches their profiles, competences and educational evolution so that they feel a sense of accomplishment and acquire as much knowledge as possible while playing the game [1], [20]–[22].

The serious games growth in popularity and the need to adapt these requirements to a virtual environment, led to the research of strategies and the implementation of personalized learning experience algorithms, that allowed technology to take the role of a virtual private tutor [21].

This sub-section focuses on three important educational principles, Gagne's Nine Events of Instruction, Bloom's Taxonomy and Keller's ARCS Model, that sought to stimulate and improve learning and improve learner's motivations, followed by a description of each of the three chosen methodologies.

2.1.1 Gagne's Nine Events of Instruction

Educational psychologist Robert Gagne, considered to be one of the "foremost contributors to the systematic approach to instructional design and training" [23] identified nine important events, called Gagne's Nine Events of Instruction, for an optimal mental condition to effectively stimulate and improve learning.

Gagne's first event mentions that the first step in any learning process should be the **capture of learners' attention**, by means of though provoking questions or facts that appeal to their curiosities. With the same intention of keeping them focused in the subject, the second event describes that learners should be provided with **a list of objectives**, that not only helps them assess their progress, but also provides a feel accomplishment.

The third event dictates that there should be a **bridge between prior knowledge**, like past experiences or the understanding of previous concepts **and the current lesson**. Gagne defends that learners have

increased efficiency for storing information in long-term memory when they are able to link the learning process to past experiences.

This leads to the fourth event, where the actual **content of the lesson** should be presented in an organized and meaningful way, starting with the explanation and followed by a demonstration. This demonstration is essential, as it blends with the fifth event and the need to **provide guidance** to learners when learning a new concept and putting it to **practice**, that according the sixth event, the practice of prior acquired knowledge should be elicited and repeated, in order to improve retention.

Additionally, during the initial stages of learning a new concept, **feedback** is extremely important and should be provided immediately, forming Gagne's seventh event. Gagne clearly differentiates between this feedback, that shouldn't have any impact on the leaner's score, and his eight events, a final assessment, where learners are required to complete a test to measure all the knowledge retained, without providing any hints or feedbacks.

Finally, the ninth event determines the need for additional efforts to ensure and **enhance knowledge retention**, in order to better transfer and continuously apply that knowledge in a **job**.

2.1.2 Bloom's Taxonomy

Benjamin Bloom, educational psychologist, dedicated most of his life's work to "improve student Learning". With his research he sought to understand the requirements to acquire knowledge, more specifically, to "classify the thinking behaviours important in the processes of learning" [24], providing teachers with measurement tools for thinking.

Bloom defined three domains present in all learning experiences: cognitive domain, related to the student´s intellectual level; affective domain, related to the student´s emotions, interest and desires and the psychomotor domain, related to the student´s motor skill and physical abilities.

By leading a group of educators, Bloom and his team created a method to classify according to the different hierarchical levels of the cognitive domain: knowledge, comprehension, application, analysis, synthesis and, as the highest, evaluation. This method is known as Bloom's Taxonomy.

Although the original Boom's taxonomy was a loved tool amongst educations, Lorin Anderson, Bloom's former student, tried to improve and update it, in order to bring it into the light of the new 21st century and expand the target audience to modern teachers. Figure 2, adapted from M. Forehand publication [24], depicts the comparison between both versions of the taxonomy, where it is possible to note the

update in terminology, mainly from nouns to verbs, as well as some structural changes, such as the inversion of the two highest steps (Evaluation, and Creating as the two highest ones).

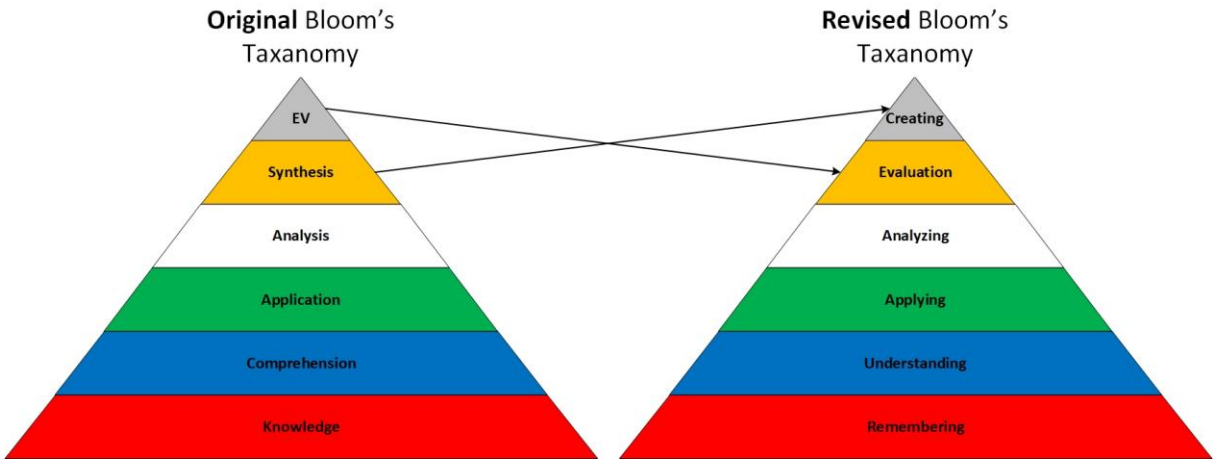

Figure 2: Comparison between the original and revised Bloom's Taxonomy (adapted from 24[])

On structural changes, the revised taxonomy adds a new dimension, identifying it as "The Knowledge Dimension", comprised of a Factual level, a Conceptual level, a Procedural level and a Meta-Cognitive Knowledge level, that intersected with a cognitive process, that includes the Remember, Understand, Apply, Analyse, Evaluate and Create level, creates a grid with twenty-four separate cells.

2.1.3 Keller's ARCS Model

Educational psychologist James Keller believed that, when it comes to learning, even the best educational method or strategy will not be able to reach an unmotivated audience. With that, Keller designed a motivational model, called Keller's ARCS Model, to be incorporated in other instructional theories, like Gagne´s events of instruction, in order to improve and support them.

According to Keller there are four steps essential to guarantee the motivation of learners: A-attention, R-relevance, C-confidence and S-satisfaction [25], [26].

Similar to Gagne's first event, Keller's model dictates a tutor's first step is to capture the **learner's attention**, dividing it into three different subcategories: perceptual Arousal, inquiry Arousal and variability. Perceptual Arousal seeks to make learners notice the topics being presented, Inquiry Arousal seeks to provoke curiosity from learners, preferably relating the questions to problems the user is interested in solving and finally Variability dictates a change in the way topics are presented by using a variety of methods, increasing the coverage of the different learning styles of each learner.

The next step is **Relevance**, where Keller presents a set of strategies to establish the topic's level of relevance, so that learner's motivations increases. One of those strategies dictates that tutors should

provide examples that easily allow learners to connect them to past experiences, providing a better sense of progression and realization of its usefulness and future worth.

The Confidence category describes that confidence is built by clearly relaying the expectation for each learner, allowing them to quickly assess the likelihood of their success. It is obvious that learner's confidence dwindles if they feel unable to succeed, however providing too easy of a content, not only limits drastically the knowledge gained, but increases boredom. Ideally, leaners should feel in control, confident that their capabilities will lead them to success and should be occasionally noted for their efforts.

Finally, the last category, **Satisfaction**, mentions the importance of learners feeling of accomplishment and satisfaction, born form their efforts and by being able to put to practice what they learned and be rewarded for it.

2.2 Learning Mechanics – Game Mechanics (LM-GM)

A key requisite, while developing a serious game, is the ability to balance gameplay with learning principles, mainly because the pedagogical intent of most games is achieved through game mechanics [27].

The LM-GM framework links learning mechanics to game mechanics, allowing for a better balance between gameplay and the educational purpose of a game, avoiding negative impacts to user´s motivations to continue playing or diminishing the amount of knowledge retained [28]. By identifying, describing and mapping some of the most important learning and game mechanics, this framework seeks to provide a tool for users to more easily describe serious games situations by relating both game and learning mechanics.

Exemplifying the use of the LM-GM framework on a real case scenario, figure 3 was adapted from a document published by a British Journal of Educational Technology [28], and identifies the gaming and learning mechanics of a Serious Game named Re-Mission, connecting them to the different stages of the game, forming a structural game map.

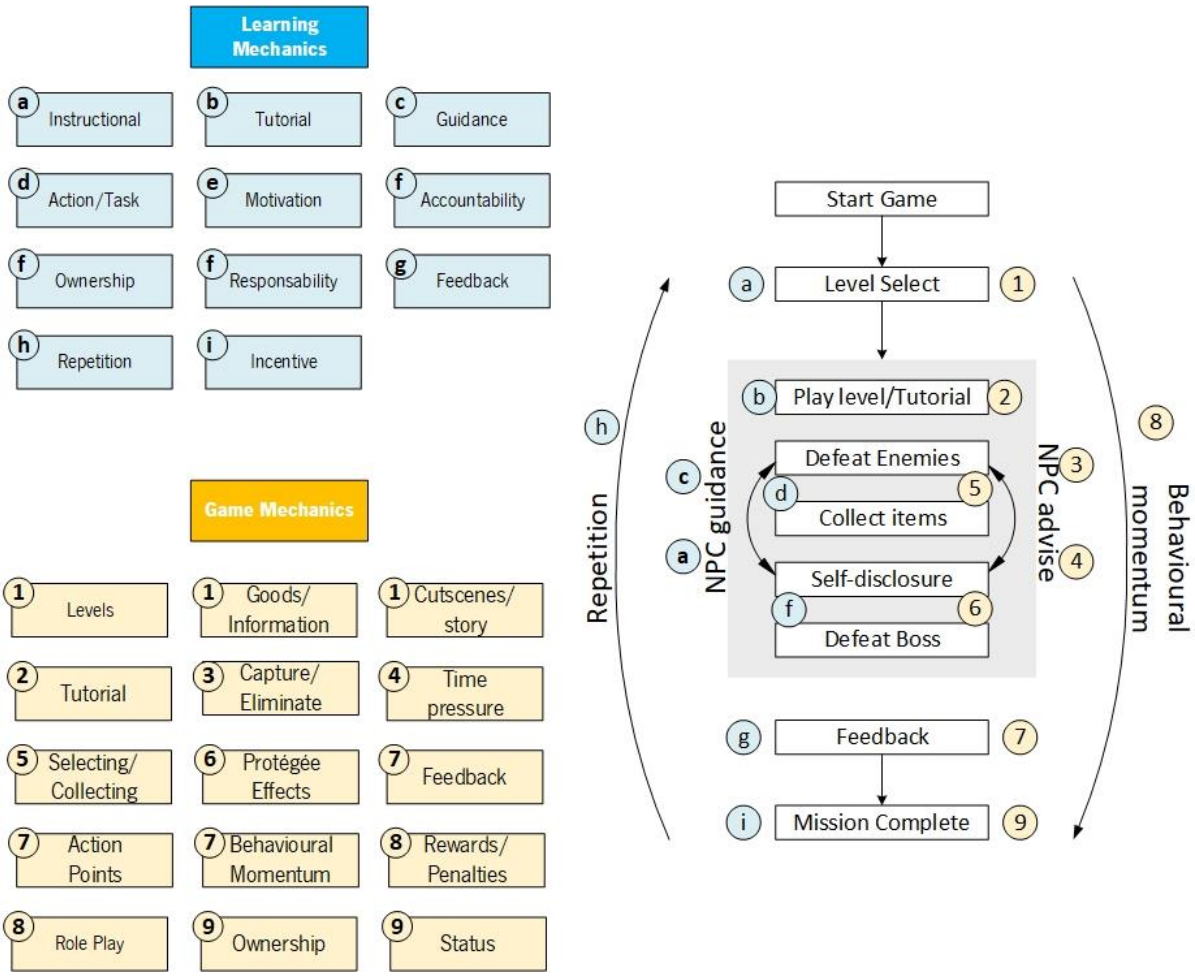

Figure 3: Learning and Game Mechanics identified in the flow of the Serious Game Re-Mission (adapted from [28])

Re-Mission is a third person shooter game, developed by HopeLab, Palo Alto, CA, developed to help young cancer patients be more involved in their own treatment, promoting "adherence to self-care during treatment" and teaching "self-care skills and related cancer knowledge" [29].

2.3 Mechanics, Dynamics and Aesthetics (MDA) / Design, Play and Experience (DPE)

To better understand the design, play and experience (DPE) framework, first it is essential to understand the concept of the Mechanics, Dynamics and Aesthetic (MDA) framework, seeing as the former is as an expansion of the latest [30].

The MDA framework was "developed and taught as part of the Game Design and Tuning Workshop at the Game Developers Conference, San Jose 2001-2004" by Marc LeBlanc and is described as a "formal approach to understanding games" with the purpose to "bridge the gap between game design and development, game criticism, and technical game research." [31].

Developing a video game is a task that requires many different areas of knowledge, and therefore a team of specialized developers to tackle each one of them. This leads to the need of a "common language", in order for everyone to better work together and understand the limitations and potentials that each area offers. The MDA framework offers the necessary tools to achieve a translation, starting by dividing the game design process into three main components, that only when working together, is possible to create a game well received by players. These components are: Mechanics, Dynamics and Aesthetics (figure 4).

Figure 4: MDA Framework

All video games are built upon numerous scripts and algorithms (Mechanics) that upon combining them are able to respond to players' inputs and interactions, changing the game world based on their decisions (Dynamics) and evoking certain emotions from them (Aesthetics).

Expanding upon this framework, Brian Winn [28] adapted it to serious games, trying to balance pedagogical intents with entertainment, providing a structure to decompose game designing and adding a variety of layers [30].

Similar to the MDA framework, the DPE framework focuses on the relationship between developers and players, depicting both parties as dependent of each other to achieve their goals and includes them into a design cycle composed of three main steps: Design, Prototype and Playtest (figure 5).

Figure 5. DPE Framework

Responsible for the **design** and development of the game, developers are tasked with the initial process of creating a set of goals and defining their game's target audience, ensuring not only their readiness to act upon feedback received from players' experiences, but also allowing for a quicker and clearer check of goals achieved.

During different stages of development, a set of players, **playtest** the game, **experiencing** the different mechanics and flow of the game, providing valuable feedback on the current game's effectiveness to

keep them motivated. This rises players to an essential position on the iterative design process, because not only they are the ones that provide developers with the necessary feedback to improve the game, but also because developers need to consider different players' backgrounds and experiences, so that they can design the game with a target audience in mind.

As mentioned before, the DPE framework divides the process of game designing into different layers, each with the respective aspect for the Design, Play and Experience phases. This framework includes the pre-existing MDA framework, compacted and renamed Gameplay, where the significant variation, from the original, is the change in terminology from aesthetics to affect. In the end this amounts to a total of four different layers: Gameplay, Learning, Storytelling and User Experience.

Each layer of the framework has an influence over each other. When developers need to change a particular aspect on a layer of the game, it is crucial that they consider the impact it causes on other layers. The order attributed to them in figure 6, adapted from [28], organizes layers from the top to the bottom, considering its level of importance and consequently the least freedom for changes, where the top layer (Learning) is usually the least malleable, seeing as SGs depend heavily on their pedagogical content and teaching effectiveness, and the bottom layer (User Experience) is the layers that developers have the most freedom to change. The entire design process is built upon a chosen technology.

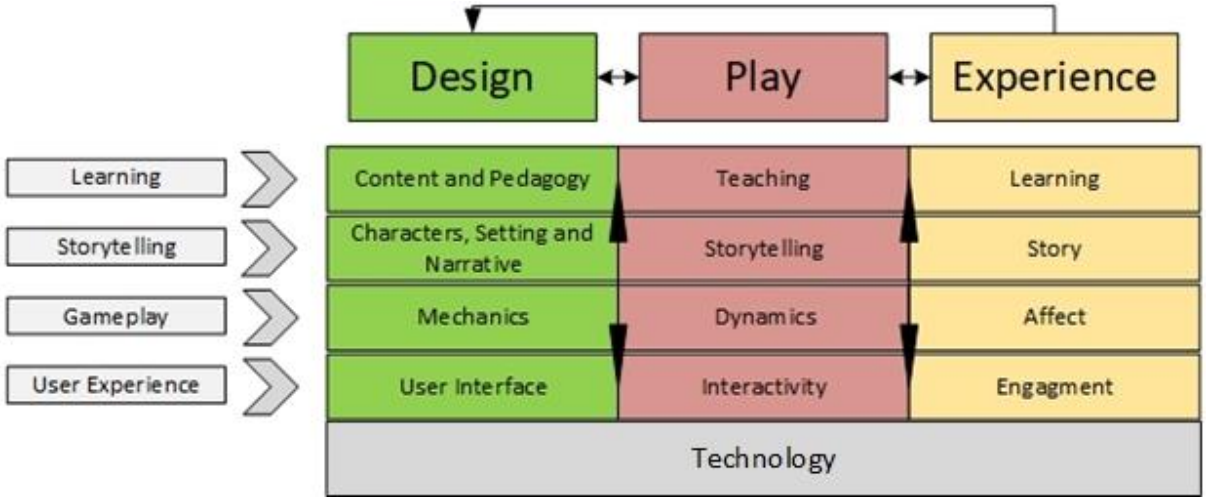

Figure 6. Expanded DPE Framework (adapted from [28])

All the educational content in a game is developed in the learning layer, where Brian Wynn [28] recommends setting clear pedagogical goals early in the developed of the game, using them as a guideline to check if the final product had the expected learning outcome out of the targeted audience.

Storytelling is composed of both the story created by the developers and the one created by players, upon making their choices on the events portraited in the game. Depending on the freedom of choices and

linearity of the story, developers are able to control the influence that players have on the occurring events, where in some cases each player reaches a completely different ending.

Similar to the MDA, the gameplay layer is the game's "rules book", determining what players can do and the impact their actions have on the game. When creating this "rules", developers must never lose focus on the reactions they are expecting from the players, in order to make it entertaining and appealing. However, this process is not always straightforward, requiring many iterations of the aforementioned design process, balancing the game according to players' feedback and the impact the pedagogical aspect is having on them.

Important to any game is the balance that developers are able to achieve between the difficulty of gameplay and challenges and the rewards attributed to player for completing them. On his literature review [28], Brian Wynn mentions three popular strategies implemented to balance gameplay: difficulty balancing, frequency of rewards and progression of play.

Basing his analyses on the theory of flow from Mihaly Csikszentmihalyi (1990), Brian Wynn [28] refers that, in order to balance the difficulty of games, there is a need to understand the skill progression of players, trying to match the difficulty of tasks to players' skill, never allowing the challenge to be too difficult, which might frustration, but also never making it too easy, risking making it boring.

Subsequently, as the game progresses, the increased difficulty and steeping learning curve requires an increase to the quantity and/or quality of rewards, keeping players motivated to play and eager to progress.

The balance of progression of play, dictates a limit to the number of players' freedom of choices as they are starting the game, increasing them as they gain new skills and new goals are introduced. This way seeks to slowly make players comfortable with the game and its mechanics, avoiding to overwhelm them with tutorials.

At the bottom of the table, the User Experience Layer encompasses all interactable elements of the game, including the user interface, where players are able to gather vital information about the current state of the game, like quest progress, rewards earned, etc. Although the user interface is important, developers have always tried to make it as invisible and simple as possible when players do not want to interact with it, creating a more immersive environment and allowing for players to focus on more important elements, like the story and gameplay.

In conclusion, this framework allows the breakdown of serious games into their most prominent layers, setting determined rules that must be followed by developers from the beginning, in order to not only make the game entertaining, but allow them to achieve their educational goals.

2.4 Relevance, Engagement, Translation, Assimilation, Immersion, Naturalization (RETAIN)

Crated by three professors of the University of Central Florida, Gunter, Kenny and Vick, the RETAIN framework [29] seeks to provide a tool to be used during either analysis, development or evaluation processes of Serious Games, so that game designers are able to evaluate the pedagogical effectiveness of an already developed Serious Games or for teachers or educators to select the ideal game for their classes. The framework uses Keller's ARCS Model, Gagne's events and the principles of Bloom's taxonomy to create six distinctive parts on every SG, each represented by each letter of its name and based on principles of game and instructional design: **R**elevance, **E**mbedding, **T**ransfer, **A**daptation, **I**mmersion, **N**aturalization [30].

As the name dictates, **Relevance** seeks to relate learners' experiences to the learning subjects, guaranteeing a link between all topics. Learners should be familiar with the learning contexts and easily connect with the scenarios and characters used, stimulating their emotional responses and motivation.

Targeting the same challenge as the LM-GM framework, **Embedding** is responsible for the integration of learning and game elements in the Serious Game, creating a natural flow and scale of the game's gameplay and story, where learners are not only acquiring knowledge but have fun at the same time.

Transfer evaluates the capacity that each player has to transfer knowledge from one situation to another. With each new challenge, players should be able to apply any gained knowledge to solve them and be able to adapt to any circumstances that might occur, making them proficient users of knowledge. This ties closely to the **Adaptation**, where the game seeks to balance assimilation with accommodation.

Immersion on the game can be measure by the level of enthusiasm, interest and intellectual investment that learns have while completing a task. Since the immersing of learners in the fantasy stetting is directly related to learning efficiency, it is required to find several strategies to maximize it. Having a clear objective with appropriate difficulty and providing at the same time valuable feedback, is proved to be a huge step on increasing learners' investment on the task at hand and increase how much knowledge is retained.

Naturalization is how much knowledge is spontaneously applied by users, without having to dedicate cognitive effort.

This framework led to the development of a table (http://www.games-ed.co.uk/retain-model.html), a product of an adaptation of RETAIN's table on [29] and [30]. Each increase of level is attributed depending on the construct of certain conceptual elements, starting at level 1 where there is a clear lack of said elements, up to level 3. It is up to the evaluator, to analyse the game and classify each of its parts according to the RETAIN table's level and attribute it a score within each of these levels.

3. TYPES OF SERIOUS GAMES

This section presents the two types of Serious Games that better describe the SG related to this dissertation, Adaptive Serious Games and Sand Box Serious Games, and an overview over some examples that most accurately represent and describe this dissertation.

3.1 Adaptive Serious Games

Motivation is a key element of learning, as it improves student´s knowledge retention when studying a new topic. Tutors/teachers play a crucial role in captivating and grabbing student's attention, having to provide them with an adaptive and scalable level of challenge, that matches each student's unique profile, competences and progress [1], [20]–[22].

Serious games growth in popularity led to research on how to include these requirements in a virtual environment, allowing the implementation of personalized learning experience algorithms, where technology takes the role of a "virtual private tutor"[21]. These types of Serious Games are also known as Adaptive Serious Games.

With a constant non-disruptive assessment of players' skills and progression, Adaptive Serious Games provide players with a personalized experience, balancing challenges

according to players' needs and difficulties, trying to fill the gaps in their knowledge, motivating them to continue playing.

Further research on Adaptive Serious Games led to the formalization of the Competence-based Knowledge Space Theory (CbKST) framework, that tries to provide the basic concepts and strategies to structure and relate a finite set of competences to their respective prerequisites [20], [21]. When trying to study a subject, it is important to have a clear idea of the set of competences that should be acquired in each lesson and consider them as prerequisites for future lessons.

Although this may prove ideal for simple case scenarios, real world problems often prove too complex, providing numerous possible actions and paths in the learning process. However, this can be simplified by "assigning each object of learning or assessment situation a set of descriptions or specifications" [20]. This means that, by attributing a numeric value to describe players' knowledge and constantly updating it as they progress, adding or subtracting from that value, it is possible to determine the specific competences of each player and, in the case of Serious Games, trigger events without stopping the flow of the game, helping them with any difficulties.

M.Kickmeier-Rust et al. [20] define five possible types of interventions (competence activation, acquisition and motivational, feedback and assessment clarification), depending on the competence state of the player (table 1).

Table 1: Types of Interventions in Serious Games

Intervention type	Classification
Competence activation intervention	Applied if the learner is stuck in some problem
Competence acquisition intervention	Learner lacks certain competences
Motivational intervention	Learner doesn´t act for a long time
Feedback	Provide learner with information about the game or progress
Assessment clarification intervention	Learner´s actions provide contradicting support for the assumption of a certain competence state

Following, two SGs that use an adaptive strategy, ELEKTRA and The Journey, where the later uses the CdKST framework.

3.1.1 ELEKTRA

ELEKTRA is a prototype 3D adventure game developed to teach optics to kids from ages 12 to 13 years old, that experimented and tested the various approaches and mechanics to improve knowledge acquisition [20]. Some of the focal points of this experimental serious game were the appropriate regularity of personalized game interventions and how to provide them. In order to test this, the game introduced a non-player character (NPC) that accompanies the player throughout the entire playthrough providing motivational and educational tips, using both visual and auditory elements.

Making experimental work is important for deep learning. It allows students to better understand concepts and realize the impact of their mistakes. Heavily penalizing players for each mistake while they are trying to solve a problem, is not be the best method to analyse their performance. To solve this, ELEKTRA's most important pedagogical tasks are only assessed by their successful completion, not allowing the players to progress unless they show the necessary competencies.

In the end, this serious game allowed to conclude that a better learning performance is possible if the game's interventions do not influence the overall flow of the game or halt player's experimentations. This way, players are able to maintain their pace, while provided with occasional hints or feedbacks.

3.1.2 The Journey

"The Journey" [21] is an adaptive SG, developed at University of Genoa, that teaches the basic concepts of probabilities to high school and entry-level university students, by employing the CbKST service to provide basic adaption features.

In this game, players take the role of the leader of a group of hikers, whose objective is to reach the top of a mountain. Along the way the group is presented with various forks on the road, prompting players to choose which path to follow and making them base that decision on the results of probability problems (as illustrated in figure 7). After choosing one and if the results obtained for each path are wrong, the game displays the correct answers and take the group back to the crossroad where a new set of tasks are loaded. If correctly calculated, the game leads the group across the chosen path and determines the success of their crossing, based on the probability distribution of the path. If successful the group moves higher in the mountain, otherwise the game leads them across the other path. In-game hints are provided after three minutes of inactivity when solving a problem.

With a database to store all the tasks, the previously mentioned CbKST service is responsible for picking the appropriate challenge for the player current competences. Once there are no more tasks and all the competences have been acquired, the group reaches the end.

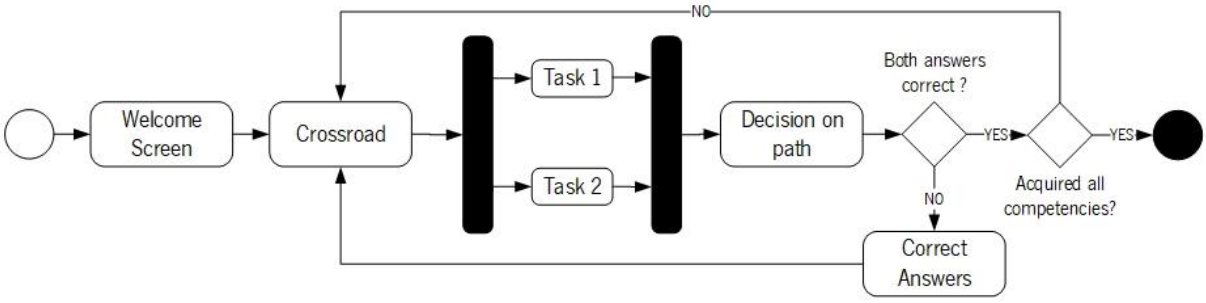

Figure 7: The Journey's game sequence [20]

3.2 Sand Box Serious Games (SBSG)

Sandbox, also known as open-world or free-roaming, is a style of video games, characterized by the freedom that players have to explore the virtual world, as well as freely select the various tasks and challenges spread across it.

This style inherent freedom and ability for players to choose their own path in the game, presented itself as a perfect scenario to build Serious Games and explore specific educational domains [1], [22], creating the Sand Box Serious Games (SBSG).

By encouraging players to explore and interact with the environment, by solving different problems or puzzles in the form of quizzes, mini-games or conversations, this type of Serious Game provides a strong basis for configurable software templates that can be easily shaped for various pedagogical intents (i.e. Adaptive Serious Games).

There are two fundamental aspects that need to be considered when designing these tasks: the content of tasks, englobing the difficulty, relevance and relation of the various topics and defined by a pedagogical expert and the delivery strategy, implemented by the game designer, defining where, when and how the different tasks become available to the players.

"Travel in Europe" is a perfect example of a SBSG, where players engage in a cultural treasure hunt across different faithful recreations of European cities, by freely controlling an avatar around the virtual city, completing different missions and challenges, signalled by 3D icons [1], [22]. Additional to the challenge of searching for the correct monument or item within a city, players need to complete mini-games once their objective is found, learning more about the monument or item in question, by manipulating 3D recreations of items, puzzles or quizzes related to the monument visited. At the end of each level, players need to complete a quiz game with questions related to the missions' city, rewarding them according to their performance.

3.3 Summary

Table 2 provides an overview over the main differences between the three Serious Games, detailed above, across five different aspects: (1) Area, referring to area of knowledge focused by the game, (2) Mechanics, referring to challenges of the game, (3) SG type, (4) Interventions, referring to the way the game provides feedback and helps players progress when stuck and (5) Progression, referring to how the flow of the game is halted or resumed, depending on players level of knowledge.

Table 2: Serious Games Summary

Name	Area	Mechanics	SG Type	Interventions	Progression
ELEKTRA	Physics (Optics)	Experimentations	Adaptive SG	Hints & Feedbacks	Levels locked until player acquires

					required competences
The Journey	Mathematics (Probabilities)	Questions & Answers	Adaptive SG; CbKST service	Hints & Feedbacks	Doesn't allow players to "climb the mountain" until they correctly answer both questions
Travel in Europe	Cultural	Explorations; Questions; Mini-games; Time-limited	Sandbox SG	3D Icons indicating available tasks	Find specific items or monuments; Answer questions

4. GAME DESIGN

This section is divided into four sub-sections, where the first one details the structure and main ideas of the game, from a story designer point of view, followed by a sub-section presenting the educational purpose of the game and describing all the topics and order for them to appear during gameplay. The last two sub-section focus on the story and the demonstration and explanation of the game's main mechanics, respectively.

4.1 Main Idea

Figure 8 depicts the main idea for the structure of the game, where the central branch corresponds to the main module, which includes all the puzzles and dialogue directly related to the main story, branching out at certain points, depending on decisions made by the player. At certain milestones, the player´s level of knowledge is going to determine whether or not the next stage will be unlocked, guaranteeing that players are comfortable using and solving problems related to base topics, before progressing into more advanced ones.

Complementarily to this main story branch, the player will be able to participate in side activities or secondary tasks, that can be accessed any time during gameplay, once unlocked. The end reflects all the main decisions made by the player, as well as the overall performance and score, making it possible for multiple gameplays reach different endings.

As mentioned before, a story will be developed wherein the main scenario will be a university campus.

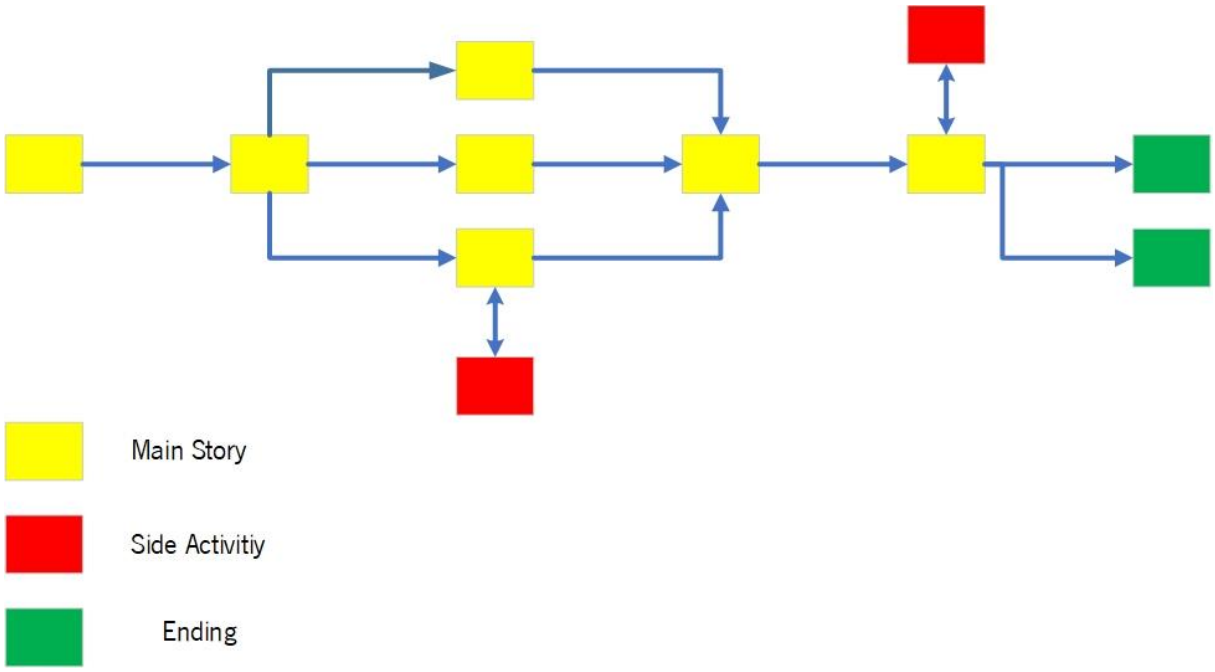

Figure 8. Game Structure

4.2 Statistics Syllabus

Being a Serious Game, it is expected that players acquire some knowledge throughout the entire game, completing it with new competences. Figure 9 represents the map of the pedagogical competencies expected for the players to acquire in the Serious Game presented in this dissertation.

Dependencies between concepts from different stages of the learning process are depicted by dashed arrow while the straight ones, outline the main learning path. Concepts that can be taught at the same time are represented parallel to each other.

In order to guarantee that students have the basic statistic's competences, this learning program starts with a quick review of previously acquired probability concepts, while at the same time, introducing conditional probability and sampling, more specifically the distinction between sample and population concepts. Throughout the entire game, descriptive statists are presented in a variety of ways.

Following the main path, after these first introductory concepts are exposed to the player, Bayes theorem and Probability distribution are taught, followed by the central theorem limit and finally inferential statistics. This final topic englobes both confidence intervals, preceded by not only the central theorem limit but also the introductory sampling concepts and the distinction between errors of type 1 and type 2.

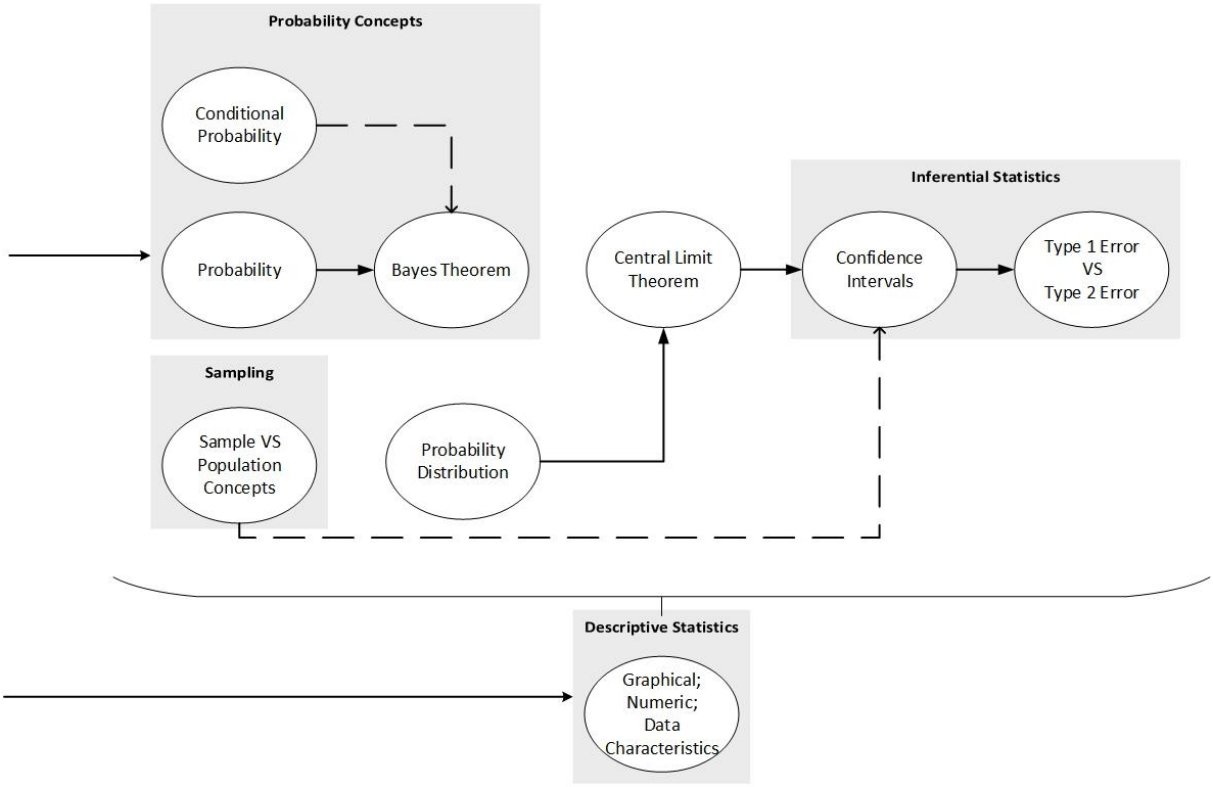

Figure 9. Learning Content Structure

4.3 Storyboard

The proposed SG simulates the life of a university student throughout a semester, allowing players to complete typical activities in the university life, such as attending classes, interacting with classmates, go to the cafeteria or refectory, etc. At the same time, players will play through the main story, where a mysterious set of problems affects the life of the main character.

Controlled by the player, this main character is part of a group of students involved on a renowned competition that requires teamwork between different courses, in order to build a functioning rescue mission robot.

Deeply involved in the project, the character's academic life is shaken when progress is halted due to issues in the university server computers, more specifically, files and test results related to the project disappear and hardware from the robot itself looks as if it was tampered with.

The player is faced with various choices and challenges as she/he has to balance detective work with academic life, and even try to catch up with the competition requirements.

To improve players interest, it is essential that they feel involved and curious to solve the mystery. For that reason, from the first scene, the game needs to catch player's attention and motivate them to keep playing, by quickly introducing the main mystery and presenting a variety of clues.

Throughout the game, players will have the chance to gather additional clues, for example interrogating other students, and make the connections necessary to determining the clues importance to solve the mystery.

Although the game has a single final objective (that of solving the mystery), there will be various ways for players to investigate the case, resulting on different outcomes and unique learning experiences.

Where some games rely heavily upon their mechanics (e.g. Simulations), while others focus solely on the story (e.g. Visual Noves), most try to balance both. This Serious Game seeks to achieve just that, providing players with an engaging and personalized storytelling as well as interesting mechanics.

Figure 10 depicts an overview of the structure and flow of the game, which is divided into four main stages: exposition, rising action, climax and falling action. It is usually possible to discern each of these stages in any structured story, either being a novel or in video game storytelling, which makes them guidelines for the writing of a new one. The sub-sections bellow detail the flow and objective of each of these different stages of the game, providing sample case scenarios, framed according to the Statistics SG addressed in this dissertation.

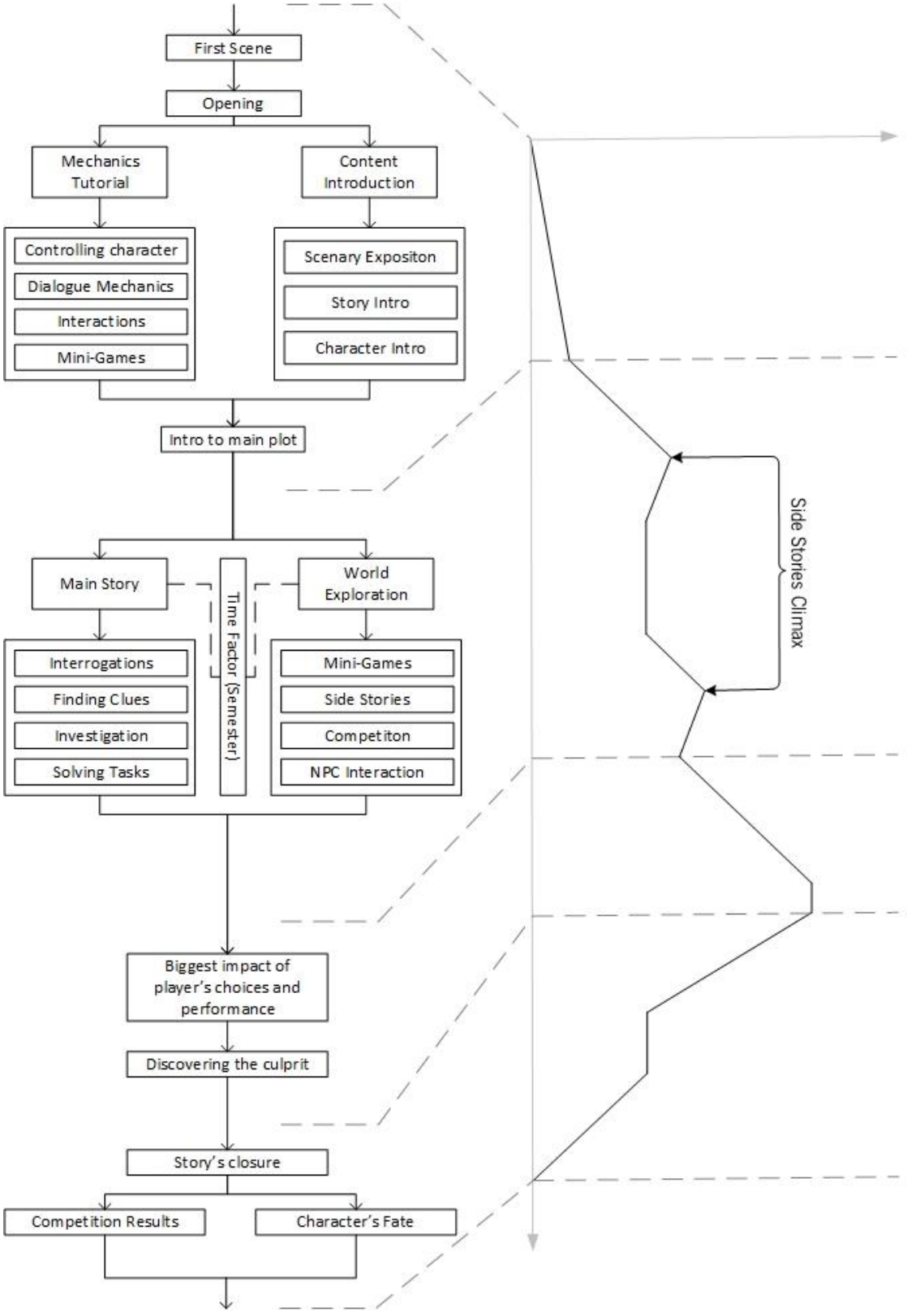

Figure 10. Serious Game Structure

4.3.1 Exposition Phase

Most video games start with an Exposition phase, where the player is introduced to the game's story, world and mechanics. It is important, that by the end of this phase, players understand the tone and main scenario of the game, as well as acquired a basic understanding of the game's style and mechanics, like movement, interactions, dialogue and task system. The first scene of the Exposition phase is usually an introductory one that poses questions that catch players' attention and motivates them to keep playing and answering them. As an example, the first scene could be a cutscene that starts by showing the context of a university (e.g., a known building facade) and a Mechanics laboratory, during the night, proceeding to focus on a single room in that building and assuming the point of view (POV) of a hard-working student finishing some project´s work. Satisfied with his/her progress, and eager to check, on the next day, the results of the tests running on the computer, the character leaves the room, stepping onto an ampler space (the Mechanics' lab), filled with machinery of all kinds, where, unexpectedly, he/she walks onto a professor. Seemingly distraught for being caught off guard in the lab that late at night, the professor immediately rebukes the student for wandering around and demands an explanation for his/her presence, coming as being a sullen and strict person. The student, shaken by the professor's reaction, explains his/her motives to be there and presents the proper authorization to use the lab. After this small exchange of words, the scene changes from the student's POV to the player's, showing a character, in such a way that he/she is not recognizable, entering the previous room and walking towards the computer.

Being the first scene in a mystery video game, its purpose is to spark players' curiosity and interest to what occurred and prompt them to find the answers: "Who were the characters in the scene? Why were they in the university so late at night? What were they working on? What happened at the end of the scene?".

As the cutscene ends, the game resorts to visual and/or audio User Interface (UI) elements to portray the end of the current day and the start of a new one, shifting the POV over to a third character: the one players are going to control for the majority of the game, i.e., the player's avatar. This time, players find themselves in control of the avatar's dialogue, while he/she casually chats with friends, over at a university's bar, about the end of the holiday season and the start of a new academic year. This creates a great opportunity for some story development, mentioning events crucial to the main story, as well as introducing new characters.

With some time to spare, the characters decide to enrol on a quick card game, changing the POV and UI of the game to a 2D overview of a table with various characters holding cards. This way players face a new challenge (i.e. mini-game) that requires them to learn new mechanics in order to successfully complete it. In this case, that means learning the card game's rules and how to interact with the different elements on the screen. Being a SG, it is extremely important to make sure that players understand that all mini-games are based on statistics' concepts, and their performance is being monitored and directly influences their final stats.

After completing the mini-game, the player's performance is evaluated and displayed. Then the game returns to the previous scenario bringing a new character into the scene, the one previously introduced working on the Mechanics lab. This event leads to the discussion about a competition involving some of the present characters, providing some details about it and referencing the influence of the main character on the team's success, even though he/she is not an official member. This transmits the competition's importance, hinting to its big role on the story ahead.

From a technical point of view, the entire scene described above, from the chat in the bar to the characters reaching their classes, is seen as a tutorial to make players comfortable with the game, slowly introducing them to the story and the different mechanics. Although these basic mechanics will be the same throughout the entire game, like the movement of characters and the means of interaction, it is possible to introduce players to different ones later on, mainly through mini-games, providing them with a sense of progression and avoiding the boredom of repetition.

Depending on the complexity of the game, the length of the Exposition Phase may vary. However, in the example given, this phase should be short, in order to quickly introduce players to the game and make them comfortable playing it, improving knowledge retention.

4.3.2 Rising Action

The transition between the Exposition Phase and the Rising Action occurs with the introduction to the story's main event, which, in this case happens with the discovery of deleted and modified important documents related to the competition, when both the main character and team members try to check on the progress of the tests mentioned in the first cutscene. This sets the main character on the path to try to discovery the cause of such event and find if someone is trying to sabotage the project's work.

Being the longest phase of the game, the Rising Action encompasses most of the main and side story's events, introducing players to all the mechanics available and different scenarios. Since the example

provided is an adaptive serious game built inside a sandbox environment, this phase gives maximum freedom to player's actions, allowing them to interact with the virtual world at their own pace choosing the course of action they desire. More specifically, players are given total control of their avatar and are able to move around the university, interacting with the NPCs to learn more about the current events, engaging in either side stories or mini-games to improve their statistical knowledge and progress in the main story. At this top level, the player (main character), together with some friends, tries to gather as many clues as possible to find the culprit behind the corrupted data. Players' knowledge is mostly gathered during this phase, where the game uses it to understand players' needs and knowledge gaps, in order to provide them with the necessary tools and activities to help surpass them. In some cases, this means blocking progressing on the main story, requiring the players to explore more

of the world and complete more side stories or mini-games, ensuring their readiness for the challenges ahead.

4.3.3 Climax

All of this effort and progress eventually lead to the climax of the game, where all questions regarding the main story's events are answered and the final most difficult challenge is presented, requesting players to apply all knowledge and skills gathered throughout the entire playthrough, resulting on the most satisfying scene of the game. It is also during this phase that Adaptive Serious Games should handsomely reward players that made good decisions in the story and managed a good performance, giving them a final feel of accomplishment and involvement in the storytelling process.

4.3.4 Falling Action

Finally, players reach the final act of the game, the Falling Action, where the game gives some closure to the story and characters.

4.4 Game Mechanics

Video games can be described as a variety of mechanics working in parallel and applied to objects and characters inserted in a constructed scenario. This section, focuses on the mechanics used in the SG and how they affect each another. Figure 11 depicts all five main mechanics: day simulation system, dialogue system, quest system, inventory system and mini-games, as well as the relation between them, where mini-games stand as the most prominent and main mechanic to convey the educational content

and knowledge to players, and is influenced by both the Daily, Dialogue and Quest System, that in turn are related between themselves and the remaining Inventory system.

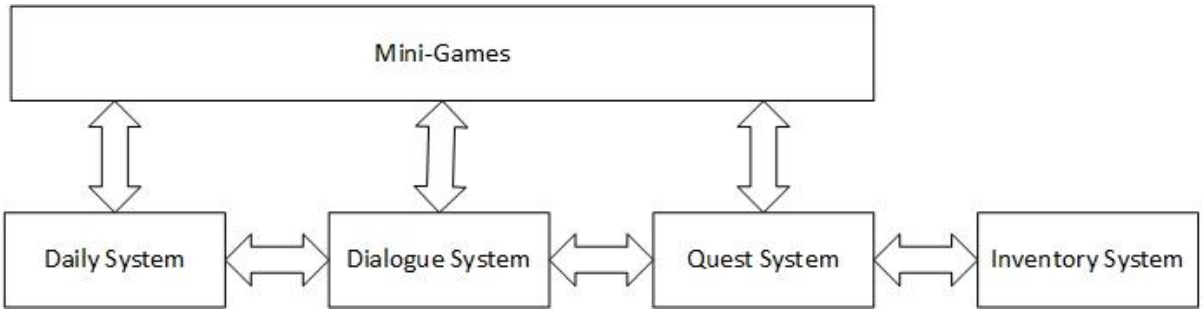

Figure 11: Game Structure

4.4.1 Daily System

To support a story whose action goes through an entire semester, it is necessary to implement a day-to-day simulator. This mechanism dynamically changes and affects the environment and non-playable character (NPCs) surrounding players, like the weather, activities being influenced by the day of the week (i.e., the week schedule), special events on certain days, like weekends or holidays, etc.

Beyond these aspects, this system also divides each day into three main stages – morning, afternoon and evening, in which the player is free to choose the main character's actions. For instance, in one morning the player decides to send the character to attend a seminar, and in one evening she/he opts to focus on trying to solve a specific quest.

The player choices have benefits and disadvantages and affect the game progress. For example, when attending an optional seminar (i.e., a game activity), the player learns certain topics that can prove useful to solve future challenges or increase players' knowledge. On the other hand, by spending time interacting with NPCs, players may acquire other insights on subjects that could help them solve challenges or unlock certain benefits. It is important to mention that in all of these cases, every challenge or activity presented will be based on statistics concepts.

Information about the current day will be accessible to players through UI elements, depicting the current day's date, weather conditions and special events (Fig.1).

Figure 12 depicts the main menu of the game, accessible to players at any point of the game. In it, players are able to check the current day's information and the closest special event (blue rectangles), the weather conditions for that day (red rectangle), as well as the current time of the day (yellow rectangle, currently displaying the afternoon icon).

With a quick glance at this menu, players are able to retrieve the general information about the current day, in order to better determine what to do, as in everyday real life.

Players can get currently available information for all the days of the semester and check on approaching deadlines or future events, by accessing the calendar app, located on the section bellow the general information and marked with a typical calendar icon.

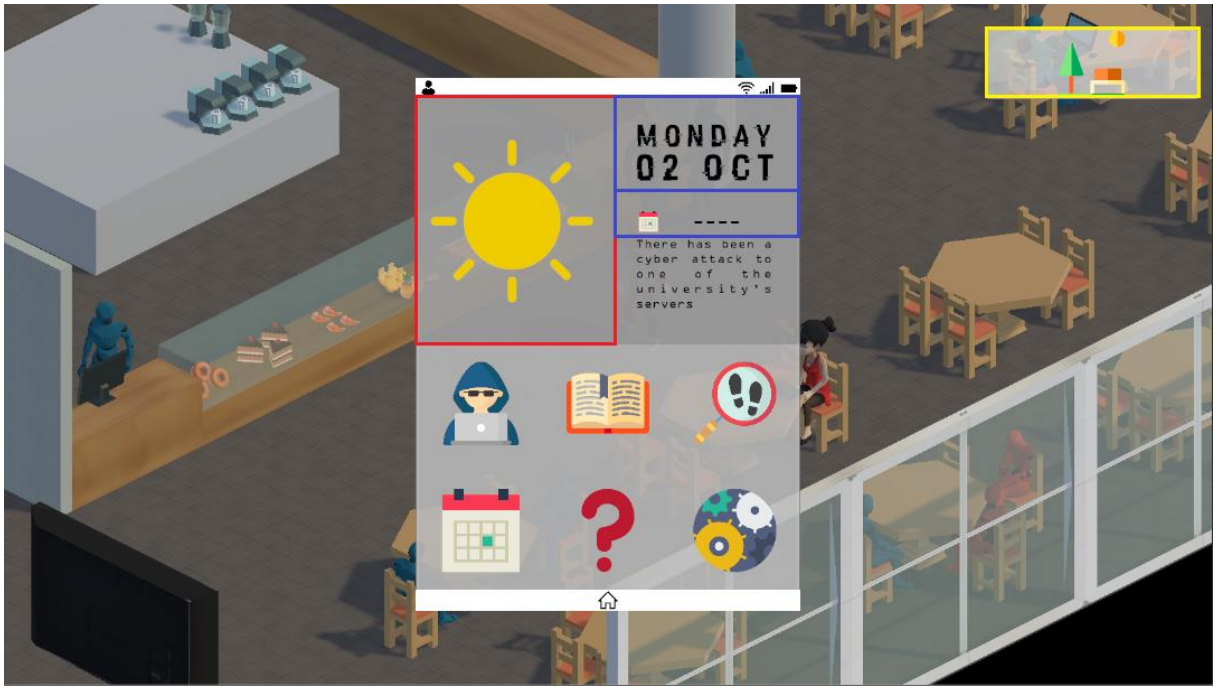

Figure 12. Main Menu

4.4.2 Dialogue System

The dialogue is a fundamental aspect of this SG, as most of the information and story is relayed through interactions with the different NPCs that populate the virtual world (example illustrated in figure 13).

With this in mind, it was necessary to explore the best approach to the development of a dialogue system, mainly on the best way to access the data, since there will be a great number of files for each character in the different scenes, on how to manipulate data and create variables, so that it becomes possible to provide players with options during dialogues, and on give NPCs “awareness” and “memory”, meaning that they will be able to react according to the changes in the environment surrounding them and to players choices and actions (e.g. grumpy when raining or mention that it was not supposed to rain so much that month).

In the early stages of the SG development, the dialogue system was supported by XML files that stored all the different nodes and respective dialogues, identifying the different conversation paths that triggered

dynamically depending on players' choices. Although this solution proved to be an efficient storage method it was only possible to achieve basic dialogue interactions and it quickly proved to not be able to meet all the required levels of customization without giving up on other aspects.

After some research on methods used by professional video game development teams, it was possible to find three open-source tools for interactive, non-linear stories: Twine (http://twinery.org), Ink (https://www.inklestudios.com/ink/) and Yarn (https://www.secretlab.com.au/yarnspinner/).

Although, any of those tools would prove to be a good choice for the case at hand, Yarn ended up being the language chosen, mainly because it was developed by Secret Lab, an Australian game dev studio, famous for "Night in the Woods", a video game developed in unity that uses Yarn as its dialogue system.

Yarn is a language designed to create interactive and dynamic dialogues for games, that uses Yarn Spinner as a parser and interpreter. Written in C#, it extremely easy to add this parser to a Unity project and start exploring its functionalities, as well as write all the dialogue using Yarn language. Although it provides a variety of pre-developed default commands, it was required to explore the libraries that make up this tool, in order to add extra functionalities.

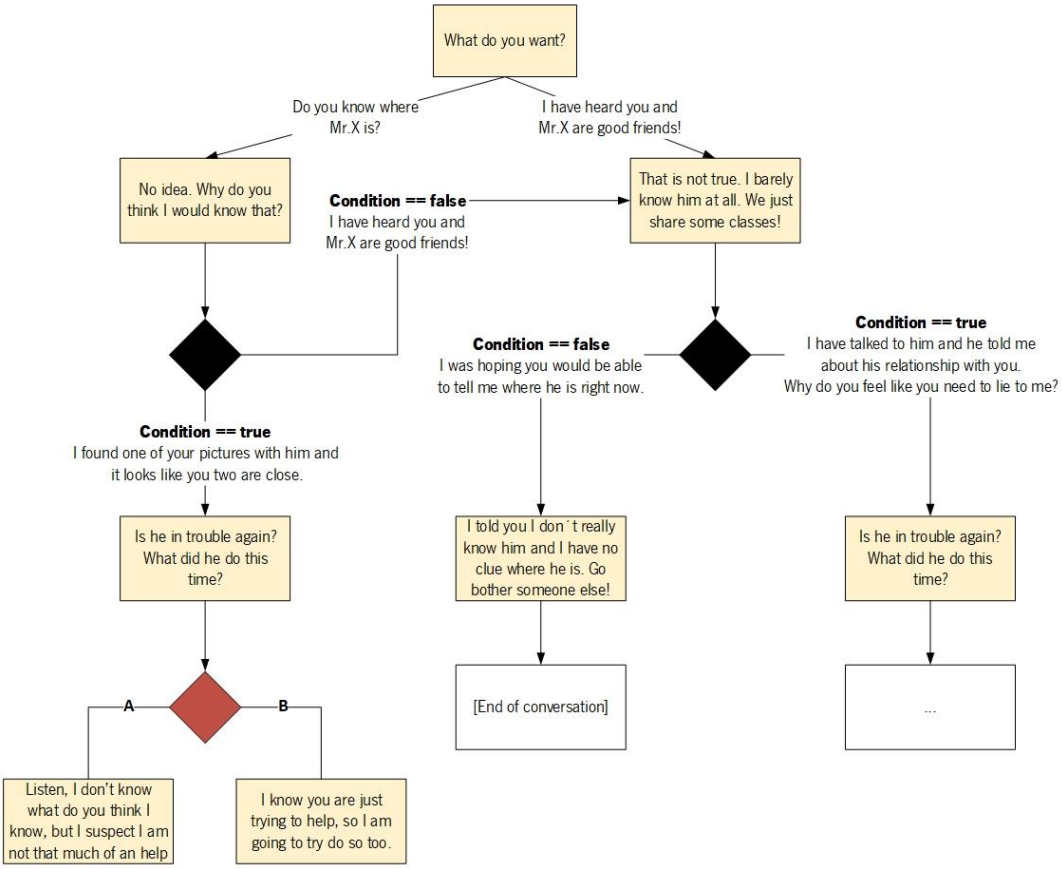

Figure 13. Dialogue System

Figure 14 shows a screenshot of an in-game dialogue between two characters. From this picture it is possible to understand the way in which the dialogue is presented to players throughout the entire game: a dialogue window to display the text, a smaller window to display the character name, which placement changes depending on who is currently speaking, and the respective icons of the characters engaged in conversation.

Figure 14. Screenshot of in-game dialogue

However, it is possible for external interference on conversations, other than NPCs, like a television in figure 15, to meddle in conversation and provide vital information to the characters and as such, the middle section of the screen, between two characters, is reserved for such occurrences.

Figure 15: Dialogue external intervention

4.4.3 Quest System

With the dialogue system in place, the next most important element of the game is the quest system. Responsible for storing and managing all the quests assigned at any stage of the game, this system allows players to check all the quests they have already completed, accepted and in progress of being completed and, if not yet accepted, the location to start the next quest of the main story. Moreover, and since each quest is com-posed of several different goals, players are able to keep track of them, as well have access to a description of the quest.

After completing all the goals in a quest, the system automatically marks that quests as completed, by displaying a small green check icon, rewarding the player and, in the case of main story's quests, assigning the next one. As the name indicates, side quests are not obligatory and if they so desire, players can only focus on the main story's path.

Figure 16 depicts the quest menu, accessible through the main menu. Once there, players are presented with all the quests completed and in progress, divided by main quests and side quests. If they want, players are able to get more information about a specific question, by clicking on the desired one.

Figure 16. Screenshot of the quest menu

4.4.4 Inventory System

In video games' terminology, an inventory is a virtual representation of a bag or pack that a character carries with him at all times, used for storing all the items gathered throughout the game.

Throughout the game, players will need to gather as many physical clues as possible, which means that they will need a storage to keep them and, if needed in the future, interact with these clues. As such, an inventory is needed not only to allow players to check which objects they have gathered, but also for the game to be able determine if players possess certain objects that unlock specific dialogue options when talking to NPCs.

Figure 17 exemplifies an in-game inventory, currently filled with two items. Besides automatically unlocking certain options, inventory items can be examined by the player to find further clues left in them. to do that the player only has to click on a clue they think hold further information, and a 3D representation of it is presented, allowing its inspection. The player can rotate the object in all directions and look for intriguing aspects. If one is found, it can be pressed and it will trigger another dialogue, providing more insights on it.

Figure 17. Screenshot of the in-game inventory

4.4.5 Mini-Games

Mini-games will put players knowledge to the test, providing challenges to players while at the same time, tutor and evaluate them, by providing hints and feedback throughout the entire experience.

Players' performance in these mini-games will have a variety of effects on the game's flow. Some of them will be what determines if players are ready to progress in the story, which represents to them that they have acquired the required knowledge up to that point of the game. Some mini-games will greatly influence players' decision when picking a path for their characters to follow, as some answers and clues are only possible to gather by completing them. In cases where players lack some knowledge and make mistakes, the game will notify their failure, guiding them to a similar challenge that focus on similar topics.

Players will also be able to find some mini-games spread throughout the entire map, at all stages of the game, fitting the environment where they are included (e.g. playing cards in the bar where the player is required to point out the probability of winning). This extra mini-games provide players with means to further improve their knowledge in a specific topic by earning rewards and preparing them to meet the main-stories' challenges.

Figure 18 is an example of a mini-game implemented in the SG. At a quick glance it is possible to determine that it is included in a completely different environment and perspective from that of the rest of the game. In this scenario, players are presented with a visual representation of the table where the

game is happening, populated with the rest of the playing NPCs, represented by their portraits. At the bottom centre of the table, depicted with a different colour from the others, is the game space of players. In this particular example, the mini-game requires additional visual elements like the flags, dices and score counters. At the top of screen, players will constantly receive feedback and hints in the form of a dialogue from one of the characters.

The mini-game's rules and objectives were based on an idea provided by a website [31], in which it is declared that the game consists on a variety of turns, where in each one, two dices are rolled, adding the sum of their values to the current points of the players. At the end of each turns, players are able to keep their score and sit the rest of the rolls, meaning that future rolls do not influence their score. As for the sum of the dices, if the total is 2,3 or 12, all characters still playing get a game score of 0 and are obliged to sit. However, if the total is 4 to 11, that value is added to their current score. The game progresses until all players are seated.

This game raises some questions related to probability concepts, for example what the expected score after a number of rolls is or when should players sit, in order to get the best chances of maximizing their score.

The objective is to populate the world of the game with as many as these mini-games as possible.

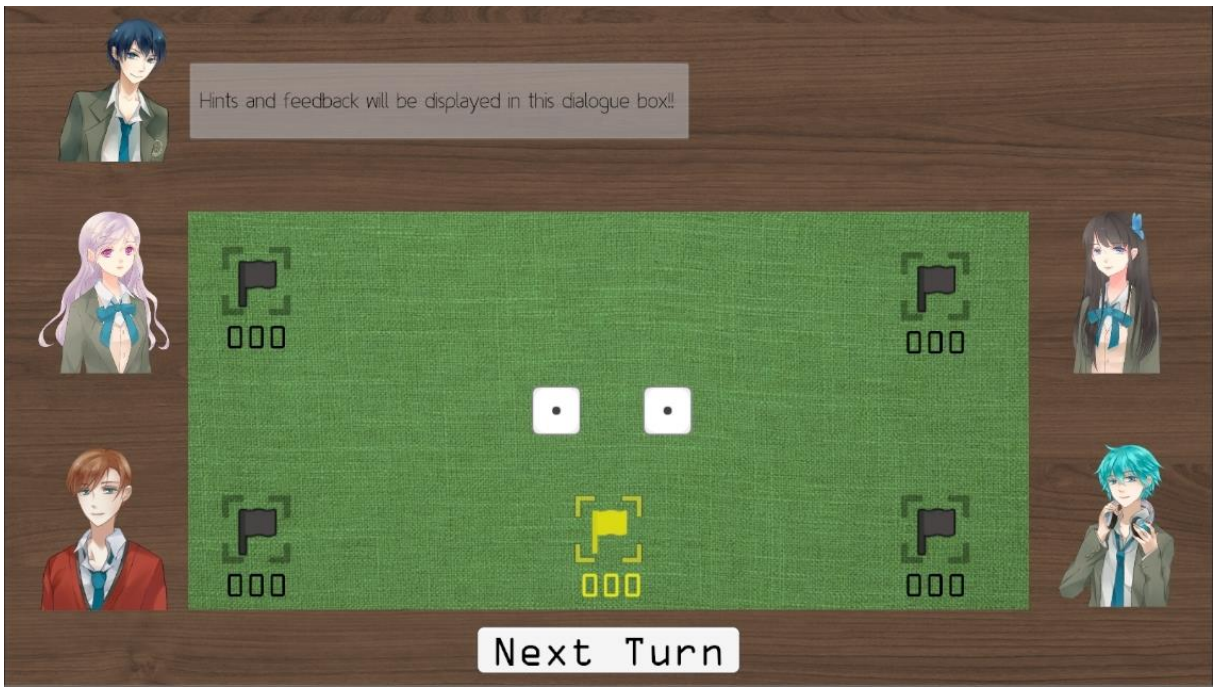

Figure 18. Screenshot of the Mini-Game

4.5 "Orchestration" Scenarios

This section explores the process of building and structuring the first scenario in the game. As this scenario is where players will first interact with the game, it is essential to introduce them to the basic mechanics of the game, through simple methods, fitting in the story and flow of the scene.

As such, this scene opens with the main character engaged in a conversation with a friend at the university's bar. Without the need of hints or feedback, players will be able to realize that they are required to choose dialogue options at certain stages of the game and understand that some of them have repercussions in the future.

At some point, additional characters will join the scene and the game unveils some elements of the story, quickly leading players to start their first mini-game. As mentioned before, mini-games change the scenario to a simplistic one, allowing players to easily understand the rules and objectives of the game.

After completing the mini-game, players are given their first main and side quest, requiring them to explore and interact with the environment.

5. IMPLEMENTATION

Although the development of this SG involved the combination of different processes and technologies, this section focuses solely on the detailing of the software created.

The SG's architecture is defined in 5.1, followed by the description of the game's structure, from 5.2 to 5.4. Finally, 5.5 describes the SG from a behavioral standpoint.

5.1 Approach

Developing a video game is a task that requires many different areas of knowledge, and therefore a team of specialized developers to tackle each one of them. Team members include: (1) programmers, to implement the main mechanics and gameplay of the game, (2) 3D modellers, that create and animate all the characters and assets, (4) technical artists, that ensure all the art is easily integrated in the game, without exceeding any technical limitations, and (5) Level and Game designers, to compose and integrate all these different elements, tinkering with the flow and feel of the game. One way to reduce the workload of a serious game development project is to reuse existing artefacts. A critical example is art design, where it is possible to search and use existing assets from huge collections on the internet. However, a large part of them are not free, and therefore project budget increases and/or additional time is necessary to find artefacts that fit right into the game scenes.

On top of this, assembling a scene is a time-consuming process on itself, as it requires the developer to use many different tools to achieve the required level of detail and graphical realism, while keeping in mind the general flow and theme of the game. Game development is not a simple task, it takes people with diverse competencies, time and money. Therefore, the focus has been in developing a game foundation, which includes (1) plotting the complete main story, (2) defining the base mechanics, and (3) implementing the initial phases, starting the story and covering the first syllabus topics.

Freely available artefacts and tools have been used for the implementation. and the graphics realism has been kept on a level capable of fulfilling the game objectives. More specifically the game graphics must offer an immersive environment, which is an essential ingredient for a (serious) game. This constitutes a solid game foundation that helps further developments and improvements, both of scenes and graphics, in the future.

5.1.1 Adopted Technologies

This SG was developed using Unity, a cross-platform game engine, created by Unity Technologies, currently supporting twenty-seven different platforms, that allows for the development of both 2D and 3D video games and simulations. Bundled with a powerful editor that allows quick prototyping and immediate deployment of levels, and combined with an easy to use interface and cross-platform support, Unity stands as a favourite in the video game Industry, mainly used by independent video game developers, who are nowadays a big influence in the video game market.

It is important to mention and discuss some of Unity's specific features, that were crucial to the development of the SG.

Unity's importance and impact in the gaming community is noticeable since, in the mobile market, games like Angry Birds 2, the sequel to the six-year-old world-wide phenomenon that was the first Angry Birds which reached over three billion downloads, Monument Valley, Pokemon GO and Super Mario Run, stand as prime examples of popular video games, were created with Unity.

As for the computer/console market, Endless Legends, Cities: Skyline, Kerbal Space Program, Ori and the Blind Forest and the world-wide famous card-trading video game, created by Blizzard Entertainment, Hearthstone: Heroes of Warcraft, stand as some of the prime examples of popular video games, created with Unity for computers/consoles.

In Unity, Scenes are the containers of environments and menus of the game, and usually, games include various of these Scene files. Upon creating a new Unity project, a Scene view is immediately displayed for the user and, apart from a default Camera and Light, they are otherwise empty. Each Scene file is treated as a unique level, and its objects are not shared, by default, with other scenes.

Another specific concept in Unity are GameObjects (GO), which are containers for components, the functional pieces of any GO. Although Unity provides multiple built-in components, it is possible to create custom ones, by writing scripts that inherit from Unity's base class MonoBeahaviour. By default, GO have only a Transform Component attached, which dictates its rotation, scale and their location in the game world. However, by attaching a variety of other components to a GO, it is possible to shape it into a character or scenery object, like an interactable item, camera, special effect, light, etc.

To create cutscenes, that is cinematic content where characters interact without input from players, Unity provides a rich tool called Timeline. This allows for the creation of gameplay and audio sequences, and even complex particle effects.

Players are able to navigate the game world by selecting a visible location and mouse click it, commanding the main character to walk there. In order to accomplish this, it is necessary to define the path for the character, avoiding obstacles along the way. For that Unity provides the NavMesh class, used for spatial queries, pathfinding and walkability tests.

Another Unity feature is Colliders, a component from Unity's physics module responsible for providing objects with a "physical body", that interacts with other colliders the same way two objects would in the real world. Prefabs is an asset type that allows the storage of GO objects, using them as templates from which it is to create new objects instances in the scene. This is useful when a GameObject is reused multiples times throughout a scene and supposed to keep with the same properties. Any change to one of these Prefabs, is mirrored to all other ones.

Although it was not an objective for this SG, Unity also allows for a quick and easy conversion of the game to other platforms, besides the targeted Desktop, requiring only adjustments to the input and screen manager.

Besides using Unity's core assets, included in the free package of Unity and available right after installing it, a variety of other tools were used for the development of this SG. These include the offline tools ProBuilder [32], ProGrids [33] Cinemachine [34] and TextMesh Pro [35], acquired by Unity Technologies and expected to make it to the base Unity installation package in the future, and the previously mentioned third-party C# library YarnSpinner, used to build the dialogue system.

The combination of both ProBuilder and ProGrids allowed for an even quicker prototyping of scenarios, where the former provided tools to easily build and mould simple and complex geometric shape and the latest added simple and functional grids to the world, allowing a better aligning of objects and modular level design.

Cinemachine provided a "unified procedural camera system for in-game cameras, cinematics and cutscenes", allowing for an easier deployment and control of cameras and TextMesh Pro is a replacement of Unity's native UI text, providing a substantial visual quality improvement, as well as other text styling and texturing tools.

5.1.2 Software Architecture

Figure 19 depicts the software architecture of the SG related to this dissertation, dividing it into three main groups, where at the base of the structure is the game engine Unity, that provides the core functionalities and features to develop a video game, like the rendering engine for 2D and 3D graphics,

the physics engine, etc allowing for the development and use of different tools. In this case, these tools belong to the upper group, and can be sub-divided into the Unity's native ones, identified by their white background, and the ones downloaded through Unity's Asset Store and added in order to facilitate certain tasks. At the top of the stack is the game itself, composed of both the custom created C# scripts and all the GOs that compose the different game scenes.

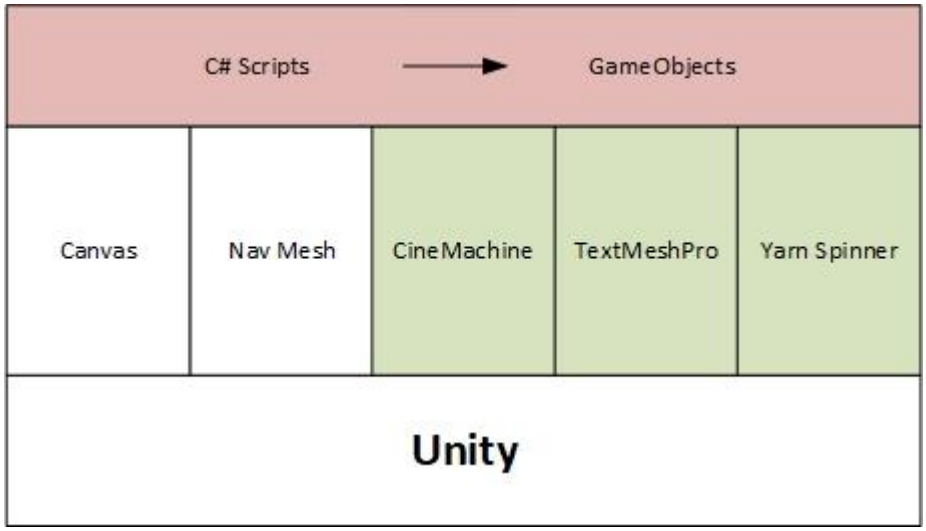

Figure 19: Software Stack

Table 3, provides an overview over all the custom scripts created that were implemented directly with Unity's Native Tools, belonging to the middle group from figure 19, and figure 20 the ones built upon these same tools, the upper group from figure 19.

It is important to notice that both the Canvas and TextMeshPro were bundle together, seeing as the TextMeshPro tool works as an extension of the Canvas.

Table 3: Custom Scripts for Unity's Native Tools

Unity Native Tools

	Canvas/ TextMeshPro	CineMachine	NavMesh	YarnSpinner
Custom Scripts	QuestUI	TimelineController	PlayerController	DialogueRunner
	QuestGoals	DialogueBehaviour	PlayerMotor	VariableStorage
	QuestButton	DialogueClip		
	QuestSlot	StopBehaviour		
	InventoryUI	StopClip		
	InventorySlot			
	Icon Switcher			

Table 4: Custom Scripts

	C# scripts -> GameObjects			
Custom Scripts	QuestManager	Interactable	Semester	Feedback
	Quest	NPCInteract	Day	LevelChanger
	Goal	NPC	Month	PlayerExitPoint
	CollectGoal	ItemPickUp	SpecialEvent	
	GoToGoal	DoorSceneChanger	GlobalControl	
	Inventory	SpecialDoors	LevelMaster	
	Item	QuestGiver	SceneFlowManager	

The sub-topics on the following section focus on each class that composes the different scripts, from the various custom created systems.

5.2 Scripts System

Scripting is an essential aspect of video game development, since scripts are necessary to respond to user inputs, arrange for events to happen, create graphical effects, control the physics of the game, implement the AI system for all characters. etc.

Following is a more detailed approach to the software created, presenting a UML diagram of all the classes that integrate each system and script of the game.

5.2.1 Quest System

Figure 20 represents the three main classes responsible for the quest system of the game. As the brain of the system, the Quest class is responsible for tracking the state of all quests, signalling when a quest

or goal is completed or assigning a new quest or goal to players. The QuestUI class aids players tracking their progress in any quest, displaying an interface with the updated data of any quest or goal. The Quest Manager is the bridge between the Quest and QuestUI class, signalling when a new Quest or Goal was completed or assigned.

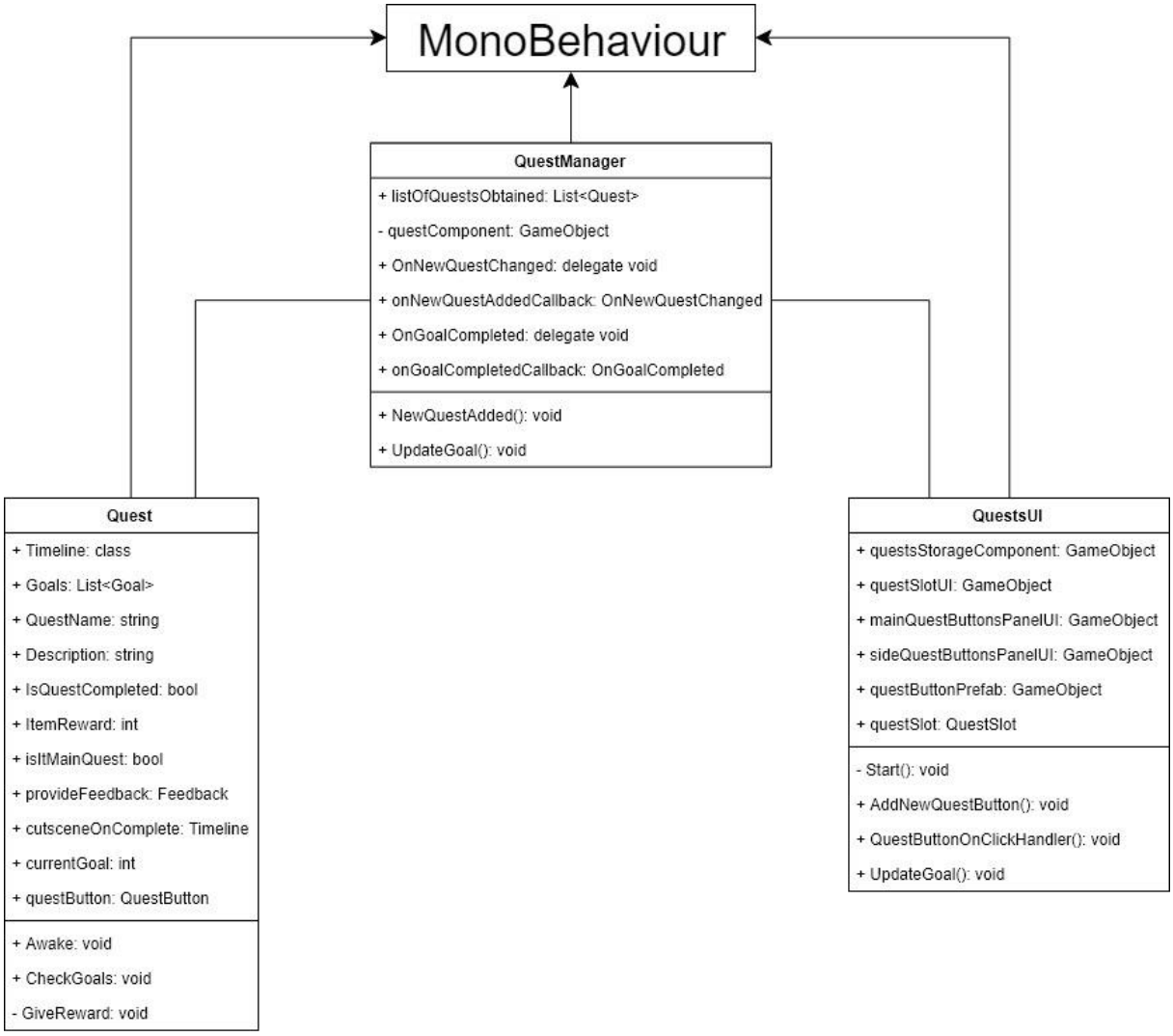

Figure 20: Classes of the quest system

However, the Quest and QuestUI classes are abstract, from which concrete subclasses are derived to fulfill the game needs.

Starting with figure 21, the Quest class is inherited by a Goal class, responsible for updating and attributing the correct goals of each quest. There are two types of quests: a pick-up type of goal processed in the CollectGoal class, completed when players retrieve a specific object placed in a scene, and the GoToGoal class, that specifies a location on a scene for the players to reach.

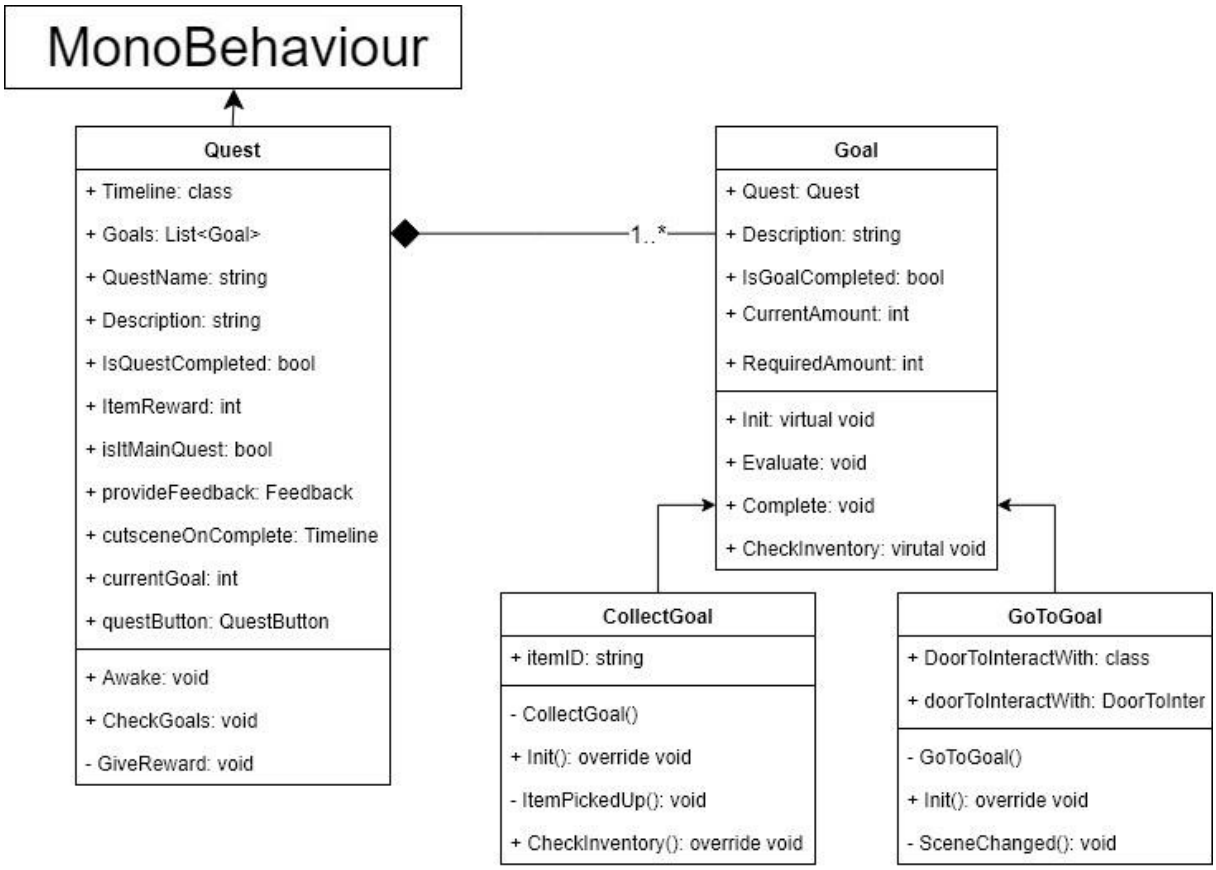

Figure 21: Classes of Quest class

QuestsUI is associated with three other classes, QuestGoals, QuestButton and QuestSlot (illustrated in figure 22).

The QuestButton class is responsible for displaying all the ongoing and completed quests, in the form of a button uniquely identified with the respective quest's name.

Upon pressing one of these buttons, all information about the quest, linked to that button, is provided and managed by the QuestSlot component, that displays the quest tittle, general information and stores the different goals of that quest to be managed by the QuestGoals class, responsible for displaying the different goals' name and progress.

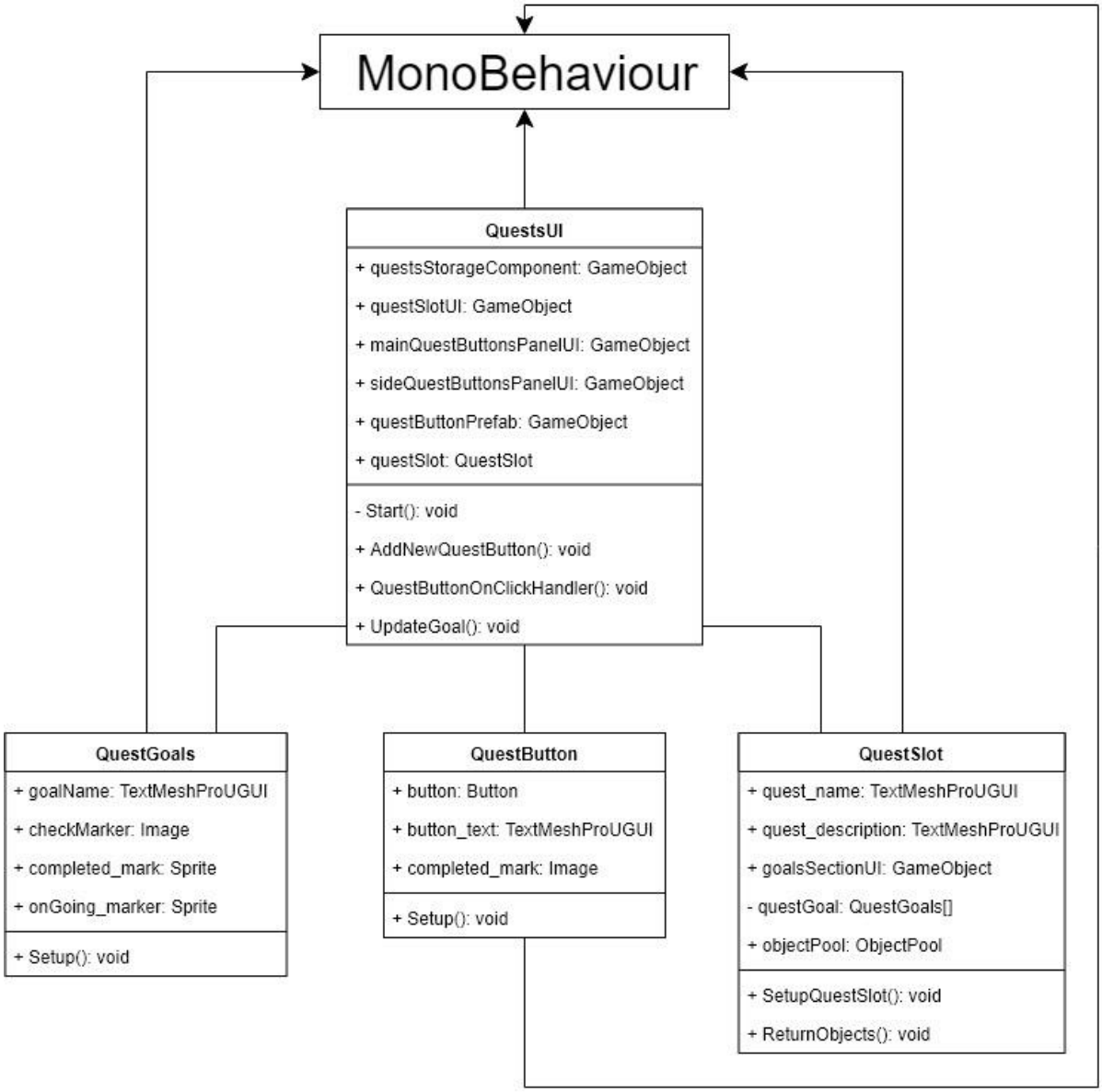

Figure 22: Classes of the Quest System interface

5.2.2 Inventory System

Similar to the Quest System, the Inventory System has two main components, the brain class Inventory, and the UI class, the InventoryUI. However, unique to this system are the collectible items, saved to an inventory slot when picked up and available for future perusal and use through the inventory screen. Each interactable item is a Scriptable Object, that is, a data container that doesn't need to be attached to any GameObject in a scene and can be saved as an asset in the project and instantiated when needed.

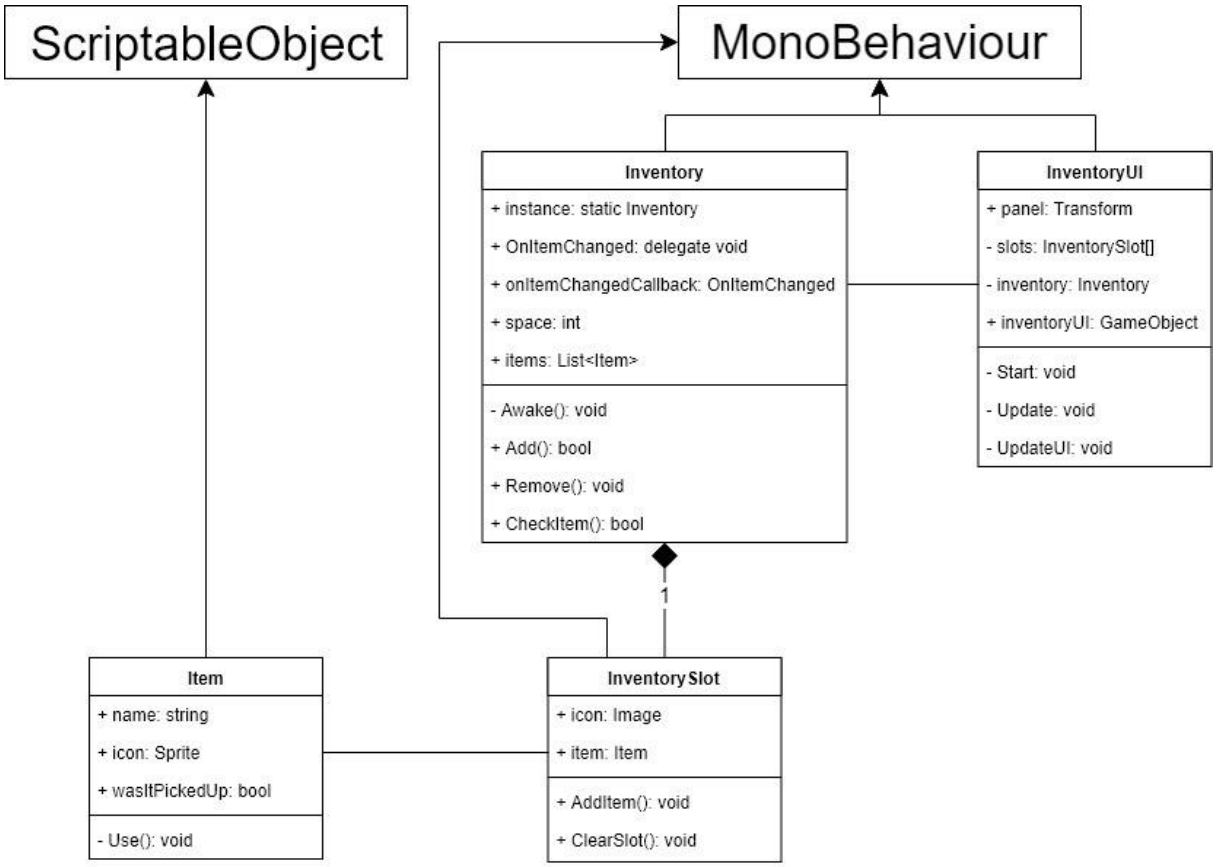

Figure 23: Classes of the Inventory System

5.2.3 Interactable

Talking to NPCs, changing scenes and investigating a particular item, are examples of possible interactions with the game's world that require individual classes to handle each case. Although each one of these types of interactions has specificities, they all share the same base Interactable class that defines all the common properties between them, namely the input required to start interacting with any object. All the details and actions for each specific interaction is then handled by a subclass, that inherits from the Interactable class. Figure 24 depicts all of these subclasses, more specifically the NPCInteract, QuestGiver, DoorSceneChanger and SpecialDoors.

Starting with the NPCInteract, this class is responsible for all interactions related with NPCs. Each NPC is unique and as such, needs a unique and distinct personality and response to interactions, provided by the NPC class. This is easily perceived by the different dialogues presented by interacting with different NPCs. For a more specific type of NPC interaction, there is the QuestGiver class, responsible for the assignment of a quest to the player.

The DoorSceneChanger is the class responsible for transporting players throughout the different scenes of the game. It is inherited by the SpecialDoors class, attributed to a particular type of doors, accessible

only at special occasions. As an example, a door that allows players to access a laboratories room, can be accessible only on Mondays and Fridays, barring players access if they try to enter on any other day.

Finally, the ItemPickUp class is responsible for managing the acquisition of all collective items on a scene, like players retrieving a book, placed on the table of a scene.

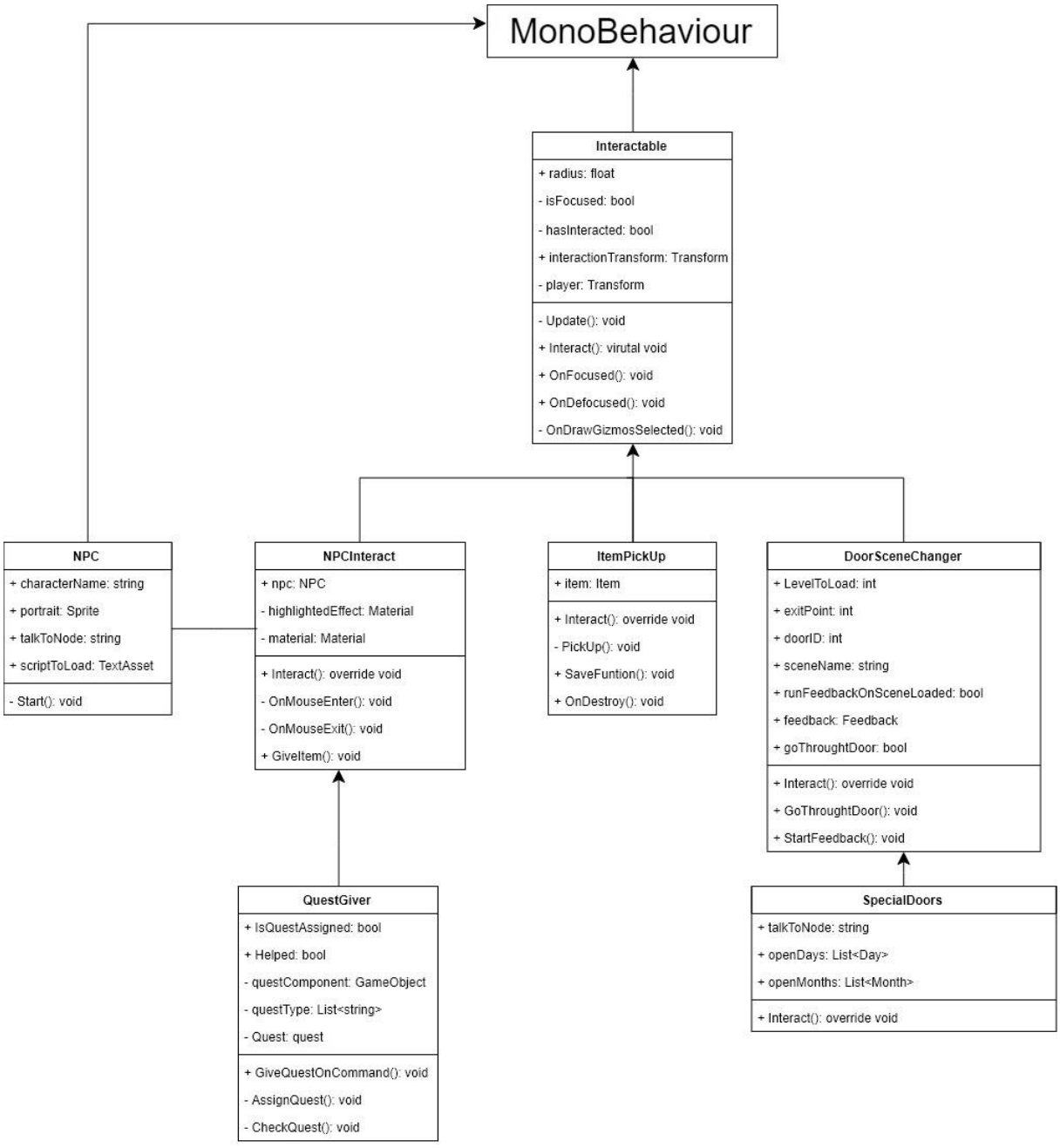

Figure 24: Interactable classes

5.2.4 Dialogue System

Since the YarnSpinner library was used for the Dialogue System, it was required to follow some of the developers' specifications, in order to simplify the integration of the library with rest of the code. As seen

in figure 25, that simplification was achieved by the inclusion of two scripts, the DialogueRunner and the DialogueUI, that handled respectively the processing of the dialogue files and the display of the dialogue content on the interface. Both of these scripts were modified to fit the SG and the IconSwitcher script was added in order to allow the character icons to change, depending on the character currently speaking and the emotions intended for it to portray.

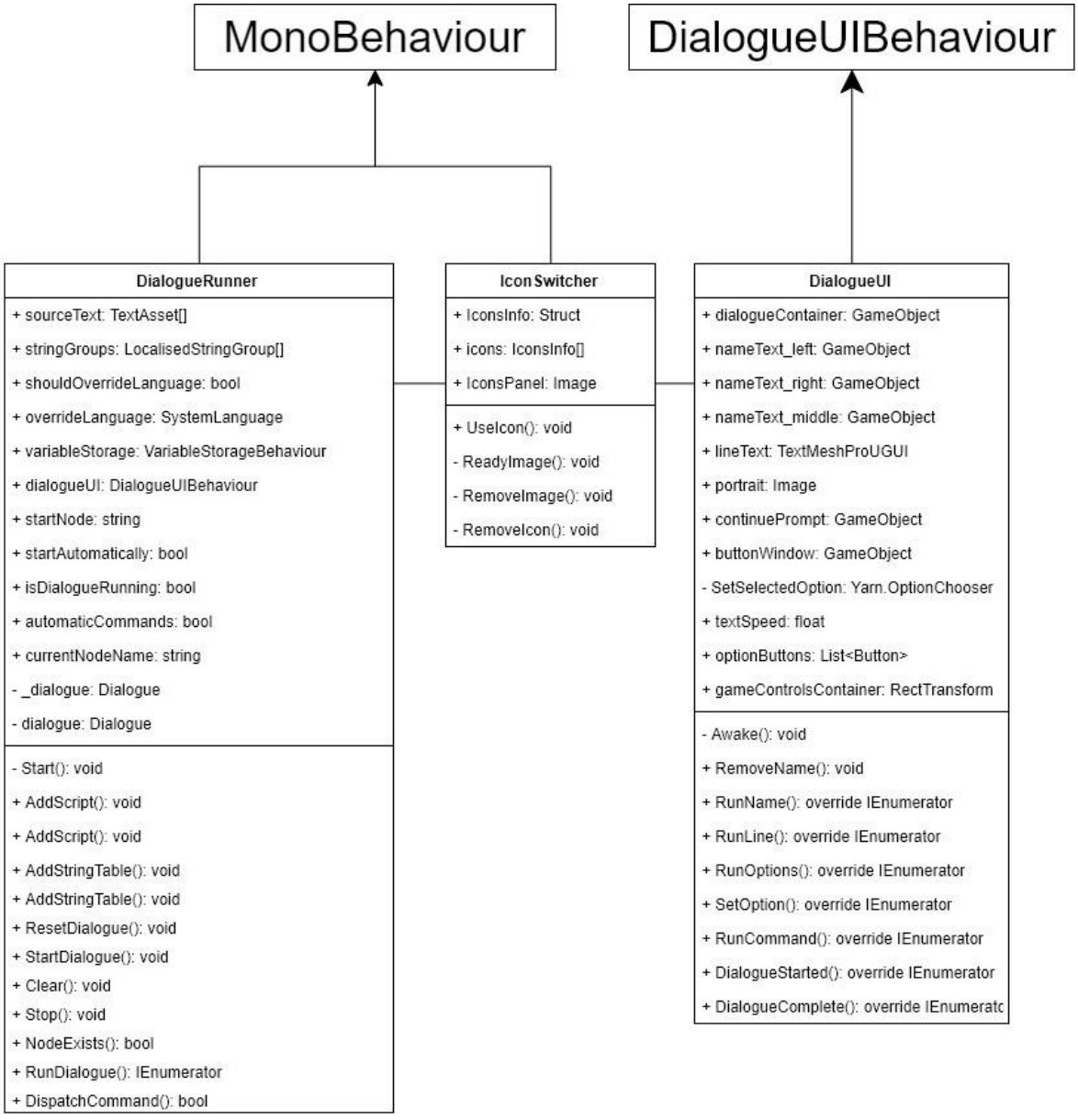

Figure 25: Classes of the Dialogue System

The non-linear nature of the dialogues of the SG required the implementation of a VariableStorage, figure 26, to save and update all choices and reactions that occur during the game. That way, it is possible for the game to adjust the outcomes of each decision.

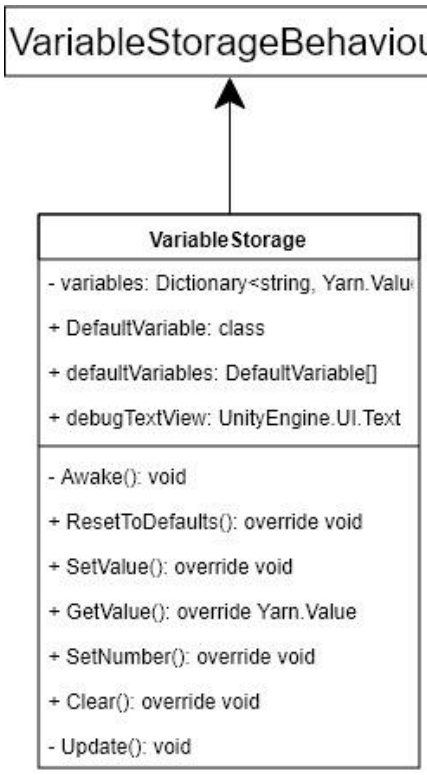

Figure 26: Variable Storage class

5.2.5 Daily System

One of this SG's goal is to simulate a semester of the life of a university student. To accomplish it, a Semester class was developed, responsible for the simulation of every day of each month of a semester, and as such, also involving the Day and Month classes. However, some events are scheduled for specific days, and therefore it is necessary to deal with the particularities of each one of those days. This is achieved by the SpecialEvent class.

As usual, players need a visual representation of these features, which is the responsibility of the DayUI class. It keeps track of upcoming events and displays the current day's information on the screen, such as the day and month name. All of these classes are represented in figure 27.

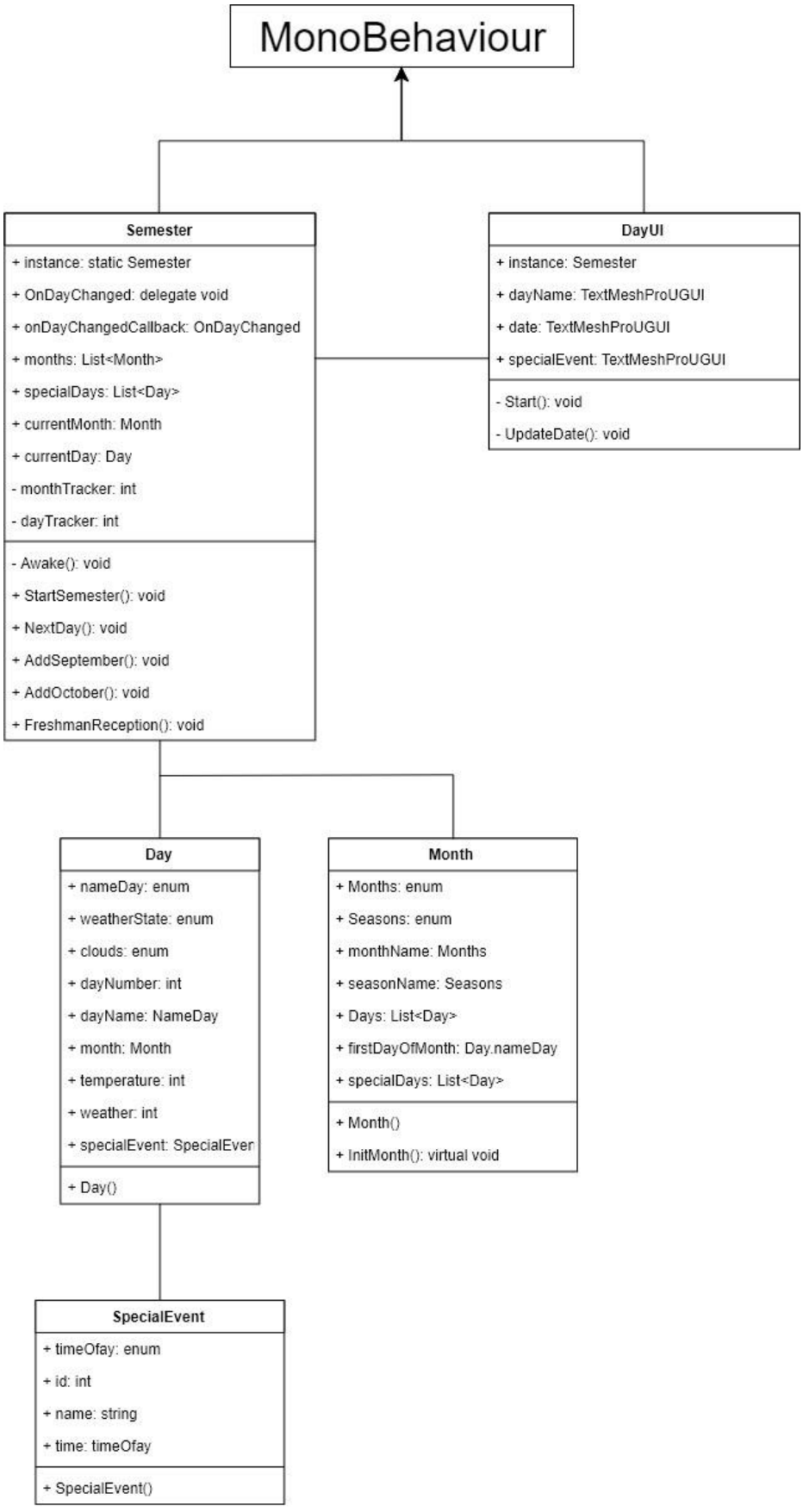

Figure 27: Classes of the Daily System

5.2.6 Timelines

Almost all video games utilize cutscenes to portrait certain vital pieces of information to the player, and this SG is no exception. By using Unity's Timelines feature, together with the previously mentioned Cinemachine tool, it is possible to create animated gameplay sequences with different camera angles and zooms that don't require any user input.

In addition to using the basic timeline tools for triggering events or animating a character, it was required to create custom behaviours to handle particular events, like starting a dialogue or signalling when a cutscene ends. This was achieved by creating timeline controller behaviours and clip scripts, such as the DialogueBehaviour and StopBehaviour, represented in figure 28 and 29 respectively. Both of these classes create clips that, when attached to the timeline in Unity, dictate the course of action of the currently playing cutscene, like signalling the start of a dialogue (DialogueBehaviour), by displaying the dialogue windows and icons and marking the end of a cutscene (StopBehaviour).

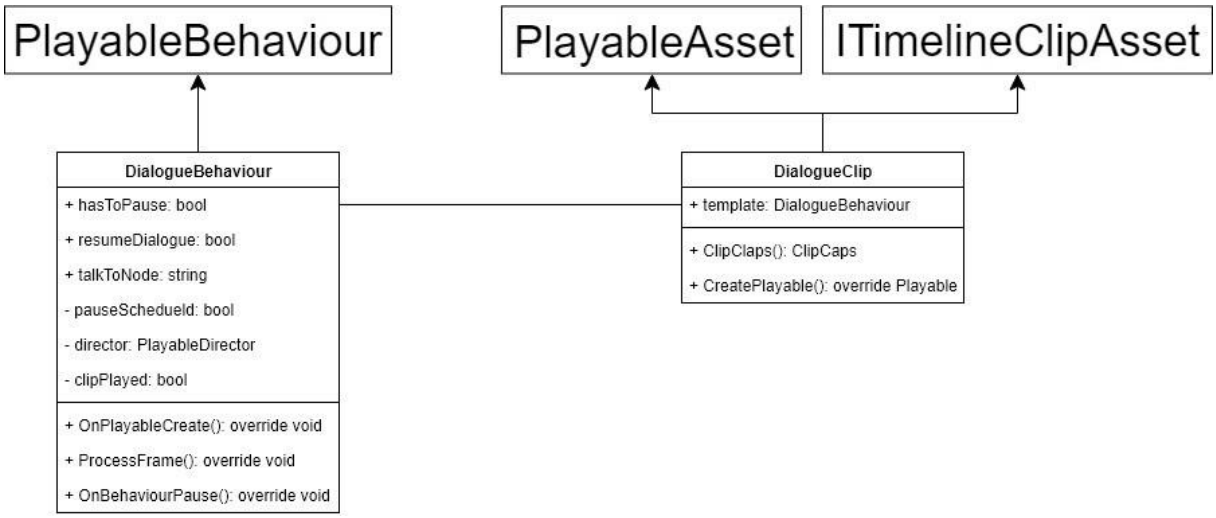

Figure 28: Classes of the Timeline Dialogue Behaviour and Clip

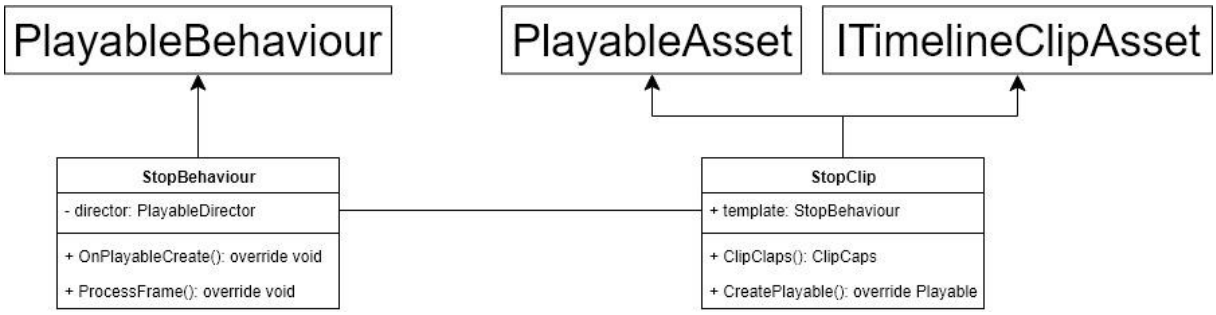

Figure 29: Classes of the Timeline Stop Behaviour and Clip

Controlling when and which cutscene starts to play is the responsibility of the TimelineController class, figure 30, that has access to all cutscenes stored for each scene.

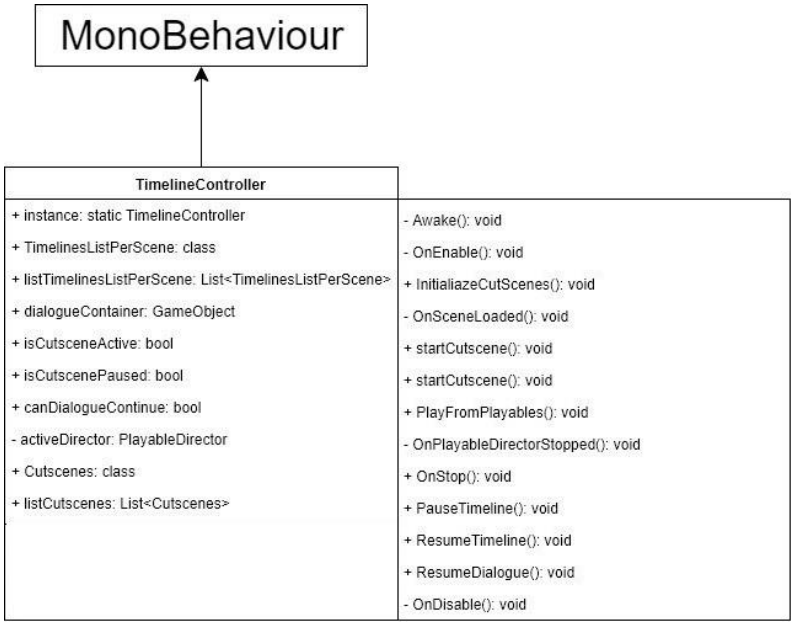

Figure 30: Timeline Controllers classes

5.2.7 Scene Manager

The level of freedom and choices offered by the SG requires a solid game flow manager, which needs to be constantly updated with each action and choice made by the player, so that it can create and unlock new challenges and paths.

For that, a group of scripts and classes were created, responsible for individual tasks to achieve the desired end goal. Starting with the brain of the operations, the GlobalControl class is updated with each action made by the player, providing feedback for the currently on-going quest, guiding players through them, saving and loading the game and handling data persistence between scenes. Saving the game is not only important to allow players to exit and resume the game without losing their progress, but it is also used as means to keep data from scene to scene. (Unity treats each scene independently and reloads them anew each time the player leaves and renters them, resetting every object to its original state.) This way, by saving each modified object and by ensuring that certain GameObjects persist to the next scene, it is possible to avoid losing data. This is done by using the DontDestroyOnLoad function on the desired scripts attached to the target GameObject.

LevelChanger and PlayerExitPoint classes are both responsible for the transition of the main character between the scenes, deciding where the player spawns on the new scene.

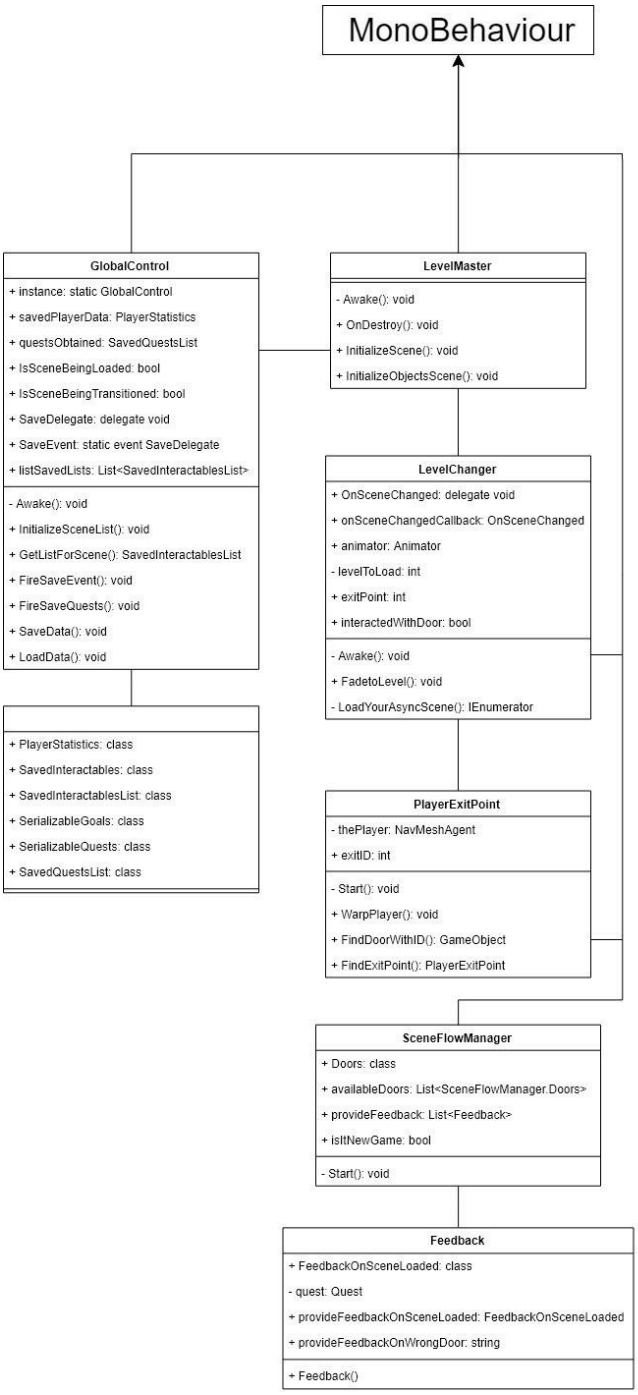

Figure 31: Level Manager classes

5.2.8 Main Menu Controller

Finally, the MainMenuController class is in charge of controlling players' access to the main menu of the game.

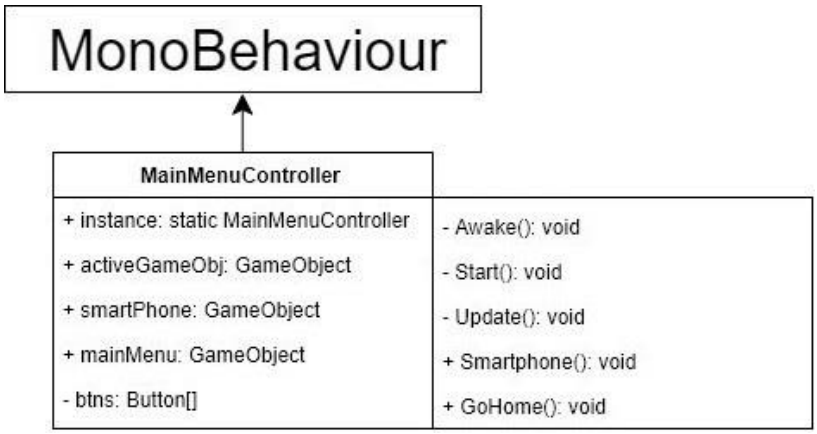

Figure 32: Main Menu Controller class

5.3 GameObjects

This sub-topic presents all the GOs used in the SG. To signal particularities among them, different line borders were used where (1) Bold and rectangular ones represents the GOs that persist between scenes, (2) Small square ones identify the GOs whose number of duplicates vary depending on the scene and a black background with white text for prefab GOs, added depending on the state and scene of the game.

Striving for an organized Unity's editor, each parent GO was divided between a variety of different groups, as depicted in figure 33, including a Camera, Gameplay, UI, Environment, Characters and Timelines group.

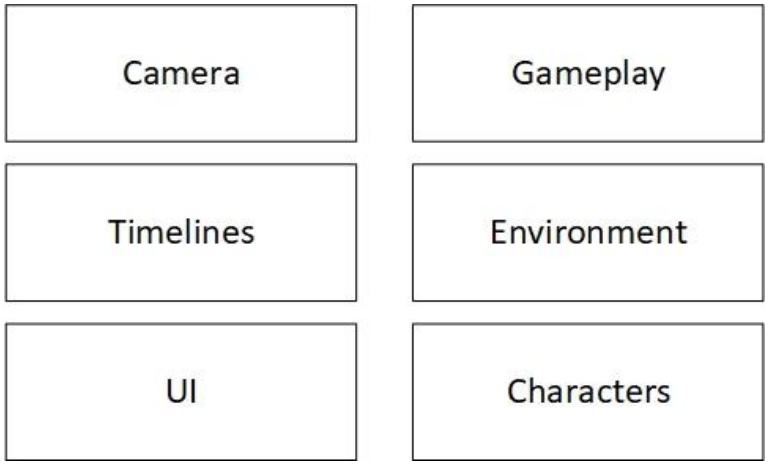

Figure 33: Main Parent GameObjects

5.3.1 Camera

Starting with the Camera related GOs, figure 34 depicts CameraTarget, whose transform component dictates the default location and rotation of game's main camera, and whose child Camera GO defines its remaining attributes and variables.

The Virtual Cameras GO serves as a container for the virtual cameras created using Cinemachine. Each of these virtual cameras, is linked to the main camera, through the Cinemachine brain component, which monitors the priority stack in order to choose the current virtual camera to activate. This virtual camera is then automatically attached to Unity's main camera.

The Main Camera child GO, is also a virtual camera, but one that is used during gameplay, following the main character around, and is only switched off during cutscene sequences.

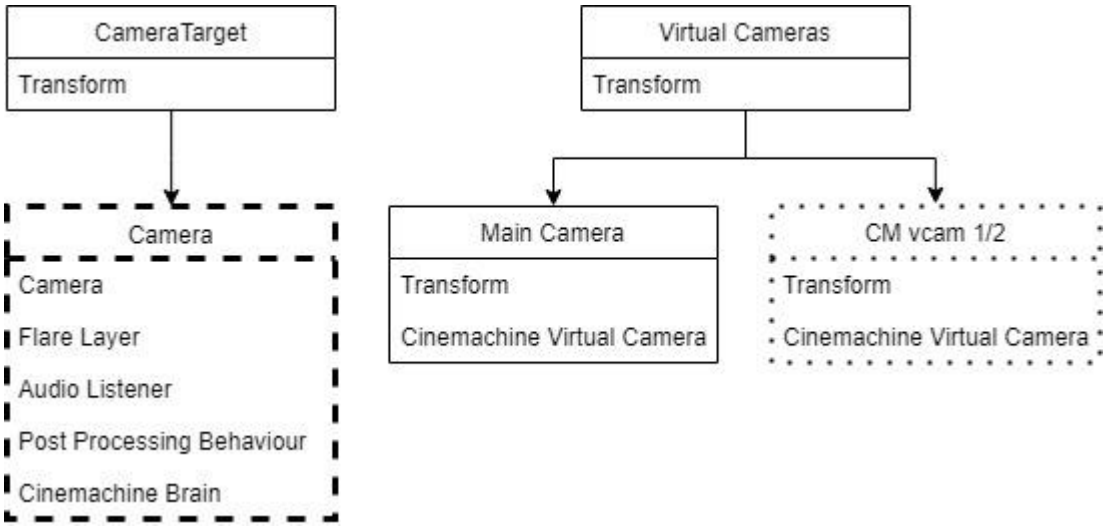

Figure 34: Cameras GameObjects and respective Components

5.3.2 Gameplay

Controlling all the gameplay mechanics are the GOs belonging to the Gameplay group, which includes: GameManager, whose components are the scripts that control all gameplay mechanics of the game; the initially empty Quest_Storage, eventually attached with quests' components assigned to players throughout the game; and NavMesh, which contains the component responsible for the pathfinding of the player's walkable areas and obstacle avoidance (figure 35).

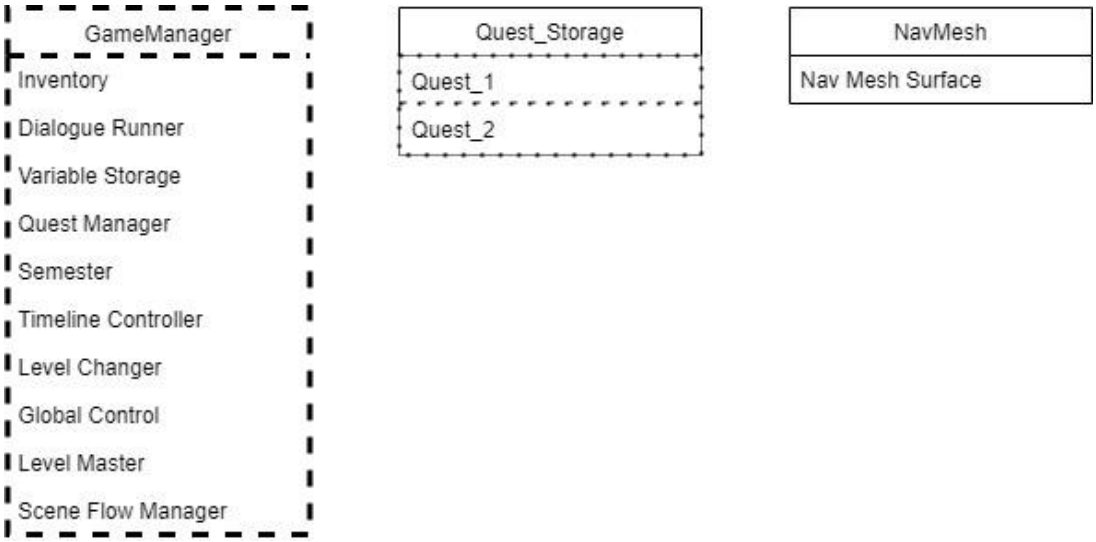

Figure 35: Gameplay GameObjects and respective Components

5.3.3 Characters

All characters in a scene, including the main character controlled by the player, are controlled by GOs that belong to the Characters group. This group includes: the Player GO, whose components define the walking mechanics and the graphical aspect of the player, as well as all the animations available for it to play; and the NPCs GO, which contains a Nav Mesh Modifier, that defines every NPC in a scene as unwalkable, to prevents the main character to walk through them.

NPC has three more specific GOs: the quest givers, NPCs that provide main or side quests when interacted with; NPCs that simply provide information about the game world's events; and the background NPCs, who are merely graphical elements, whose purpose is to better immerse players in the world by simulating varied student interactions throughout the university campus.

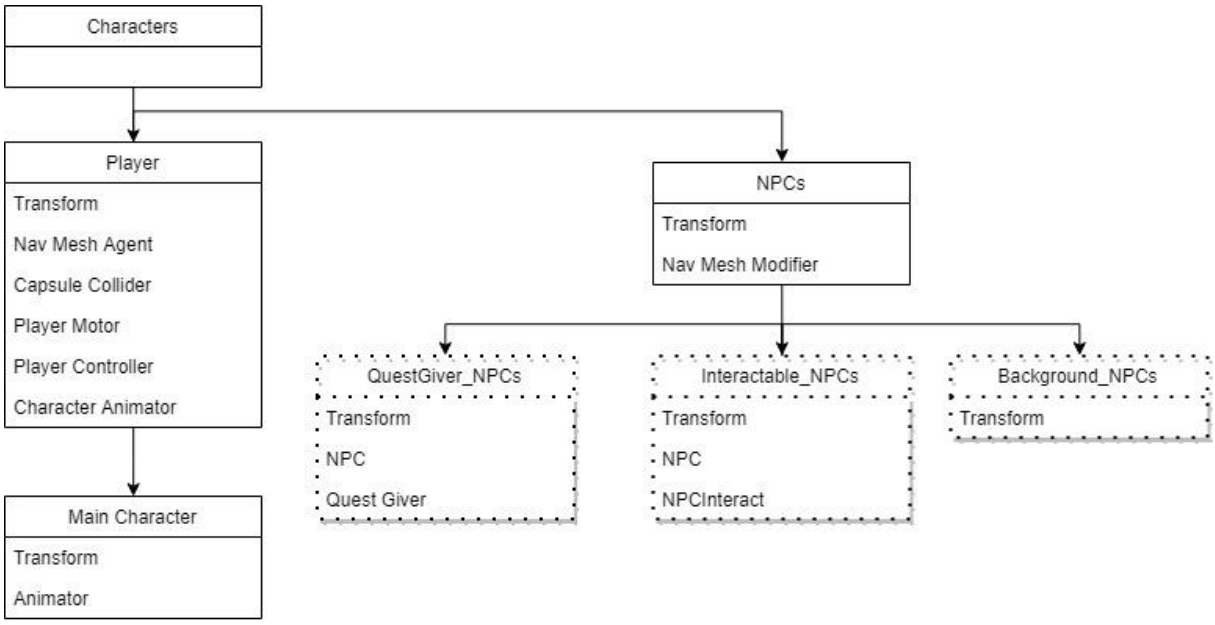

Figure 36: Characters GameObjects and respective Components

5.3.4 Timelines

The Timelines group, in figure 37, includes the GOs responsible for playing cutscenes, as well as the GOs that represent each cutscene's participant.

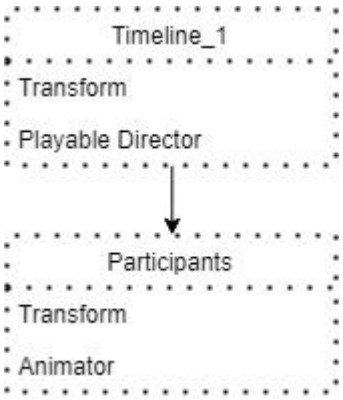

Figure 37: Timelines GameObjects and respective Components

5.3.5 Environment

The Environment group, shown in figure 38, consists of the Environment GO, which includes all the physical graphical objects and effects present in a scene. It has four main children GOs: the static Objects, that serve only as graphical elements in a scene; the Doors GO englobing all the interactable doors in the game, that allow for players to transit between scenes; the Interactable Objects that, as the name suggests, include all the objects available for players to interact with; and the Directional Light, that simulates light across the entire scene.

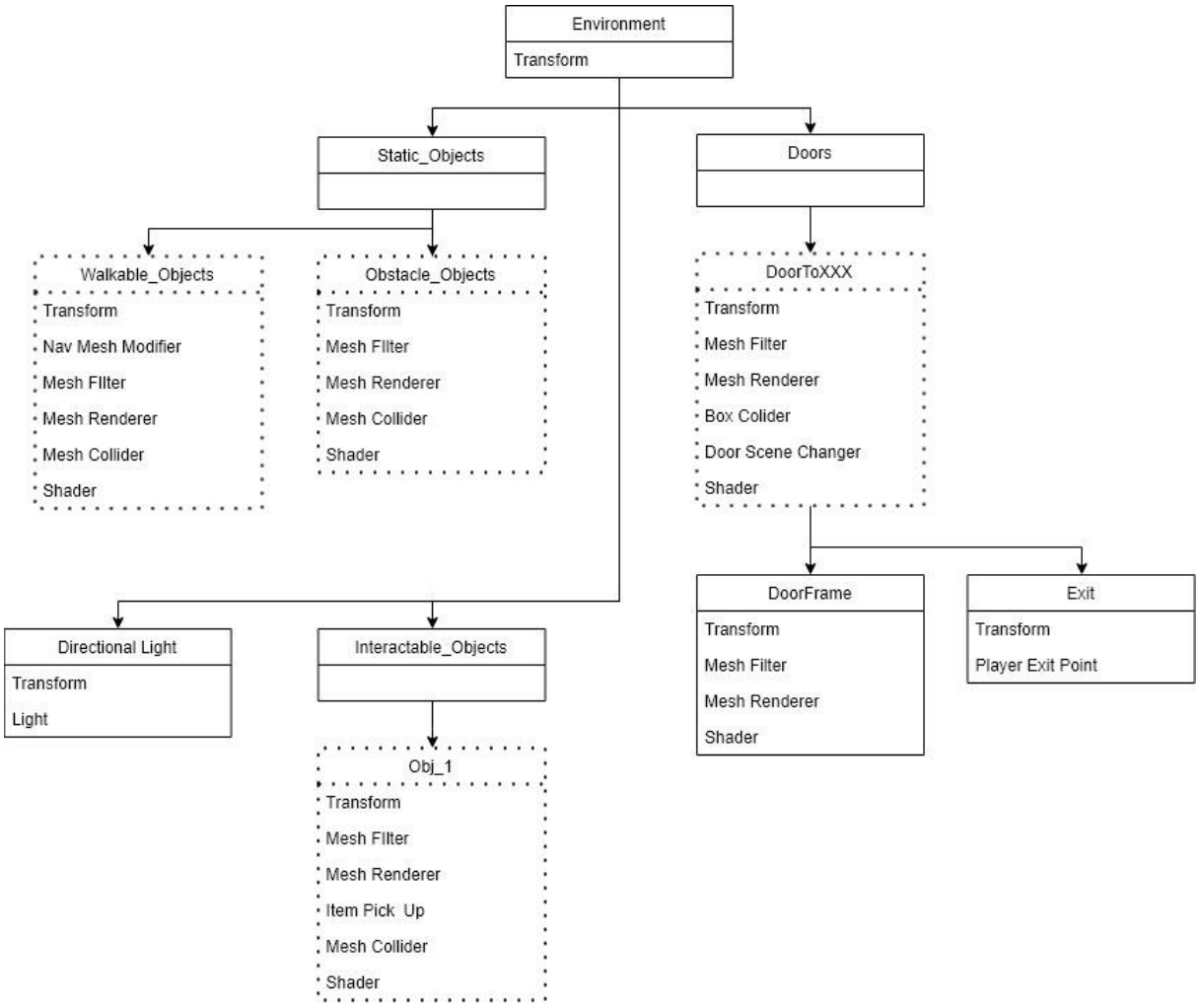

Figure 38: Environment GameObjects and respective Components

5.3.6 UI

Finally, the last group is the UI, populated by the Canvas GO. As presented in figure 39, the children of this parent GO include: Smartphone, responsible for displaying and controlling the main menu of the game; Inventory, which displays and processes all user input on the inventory interface; and Dialogue, where all dialogue content, including icons, names and conversations text is displayed.

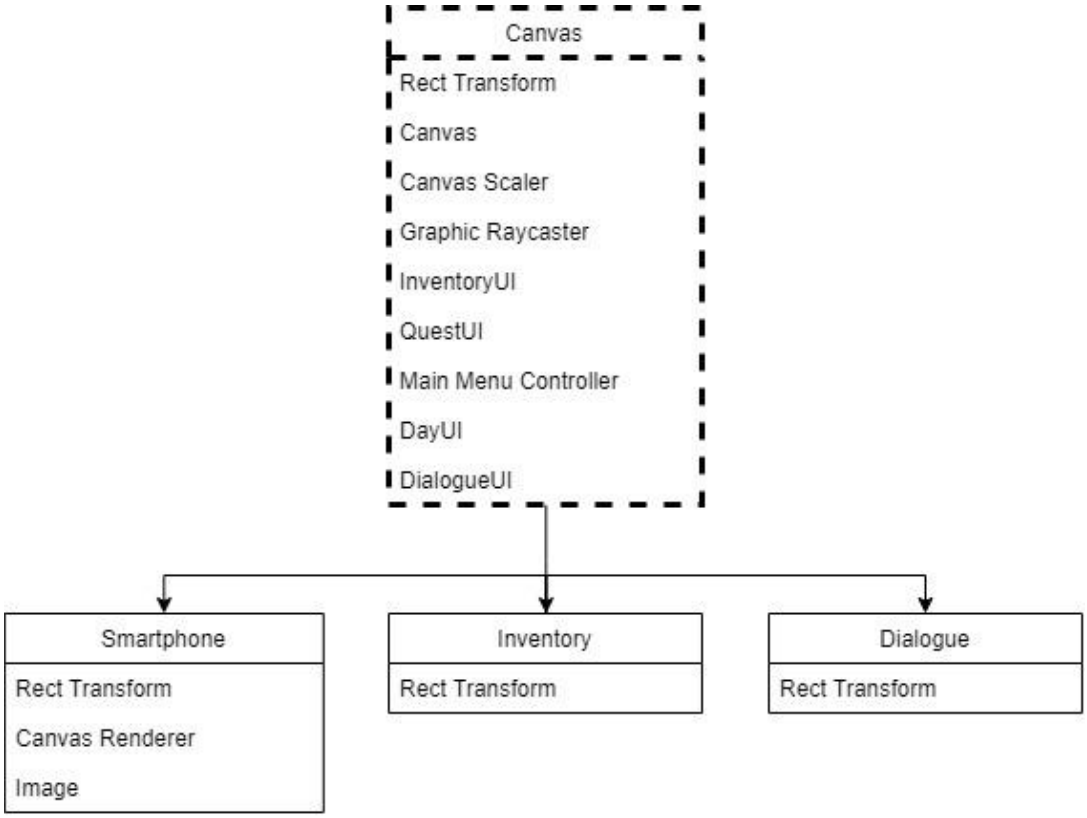

Figure 39: Canvas GameObjects and respective Components

The Smartphone GO, figure 40, controls the display and interaction with the main menu of the game. This menu was developed so that its graphical appearance is similar to that of a smartphone device, and therefore explains the top and bottom border GOs, whose purpose is merely aesthetic. The Blocker GO prevents players from interacting with the game world when navigating the main menu.

The Initial_Screen GO, figure 41, contains all UI elements that are displayed immediately after the player requests access to the main menu. This includes not only the basic data on the current day's events, figure 42, but also an array of buttons that allow the player to navigate the main menu and access the desired information/functionality. At the moment the only accessible information is the quest menu,

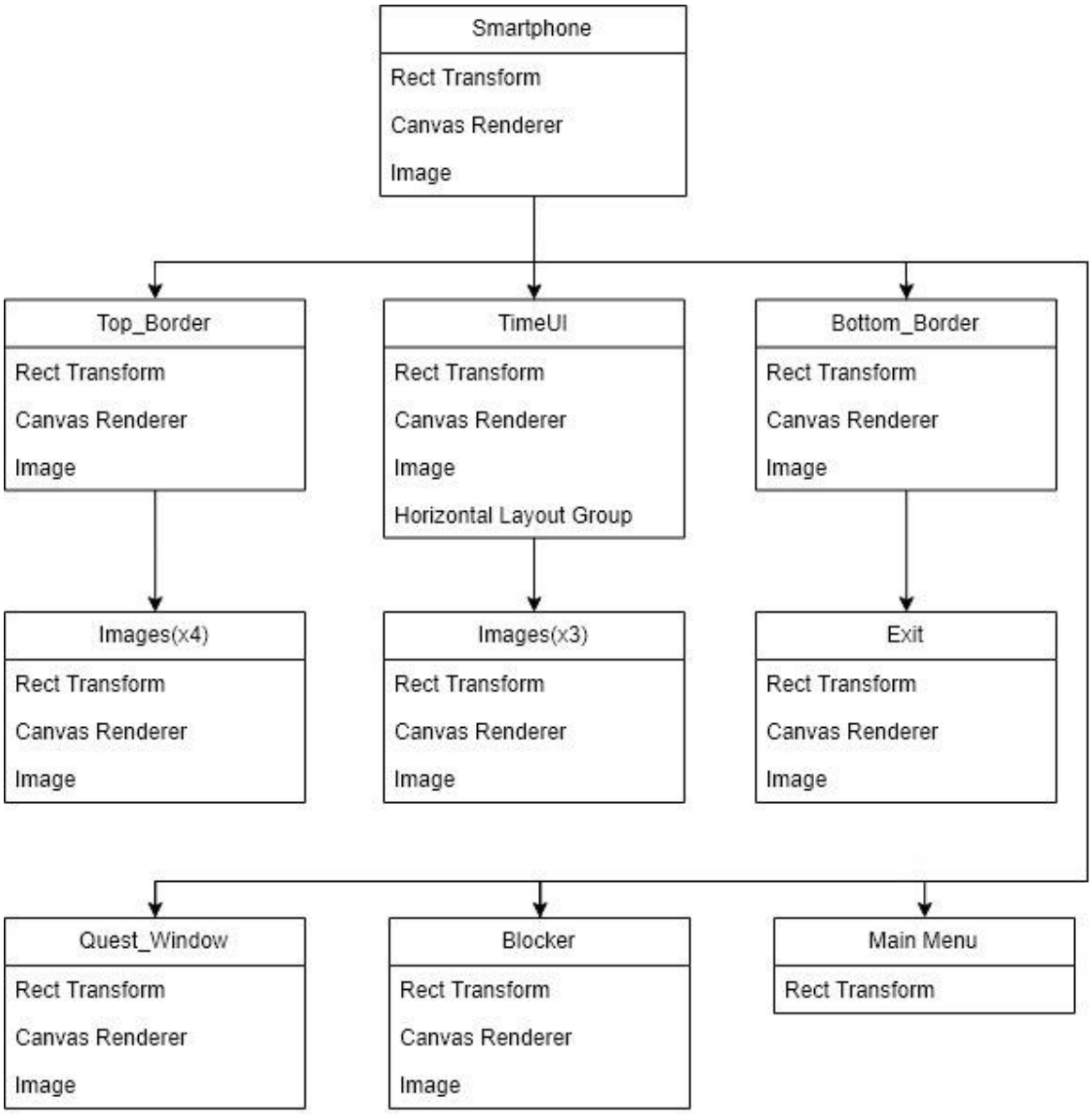

Figure 40: MainMenu(Smartphone) GameObjects and respective Components

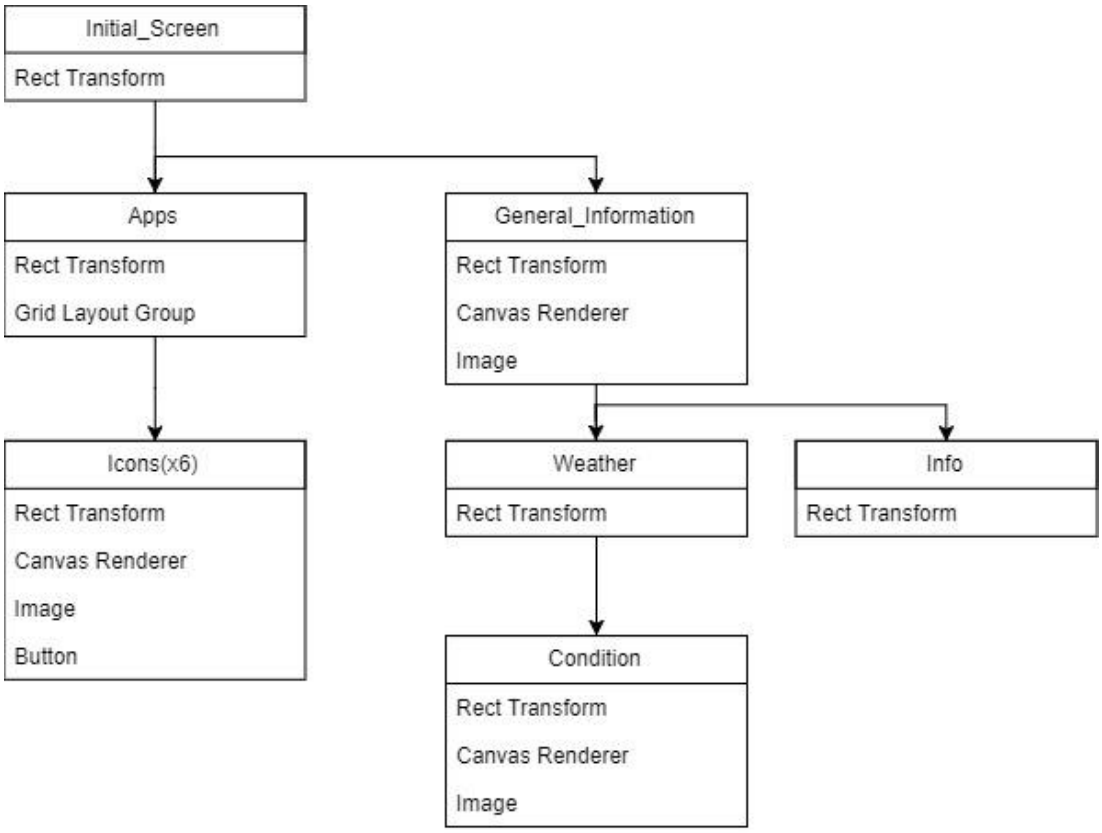

Figure 41: Initial Screen GameObjects and respective Components

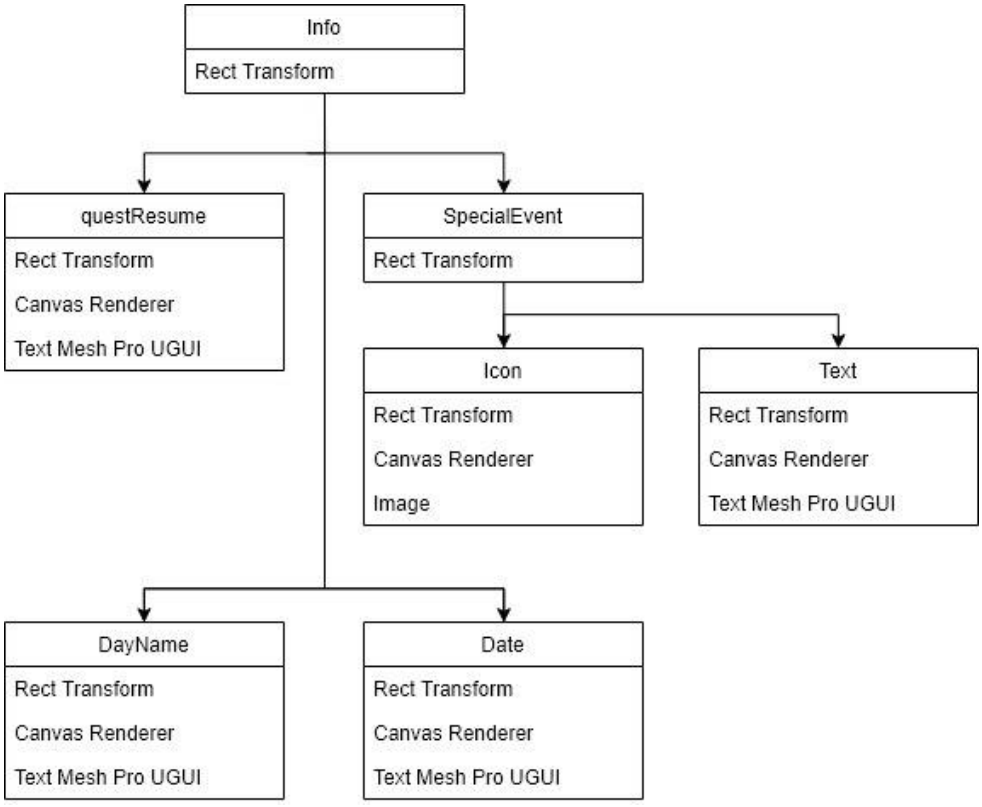

Figure 42: Info GameObjects and respective Components

responsibility of the Quest_Window GO (figure 43).

By accessing the quest menu, players are presented with the quest window, which comprises the Quest_Portfolio and Quest_Information children GOs. The Quest Portfolio is where the name of all completed and in progress quests are dynamically added and displayed. Button prefabs are used to identify each one of those quests, and each button text corresponds to the quest's title. These quest buttons are organized by their type, separating them between the main and side quests. The Quest_Information GO, initially hidden, is displayed when players press one of the quests' buttons. After that, a new section is displayed, separated between a brief explanation of the selected quest, and all the completed and in progress goals, dynamically populated in the Goals_Section GO.

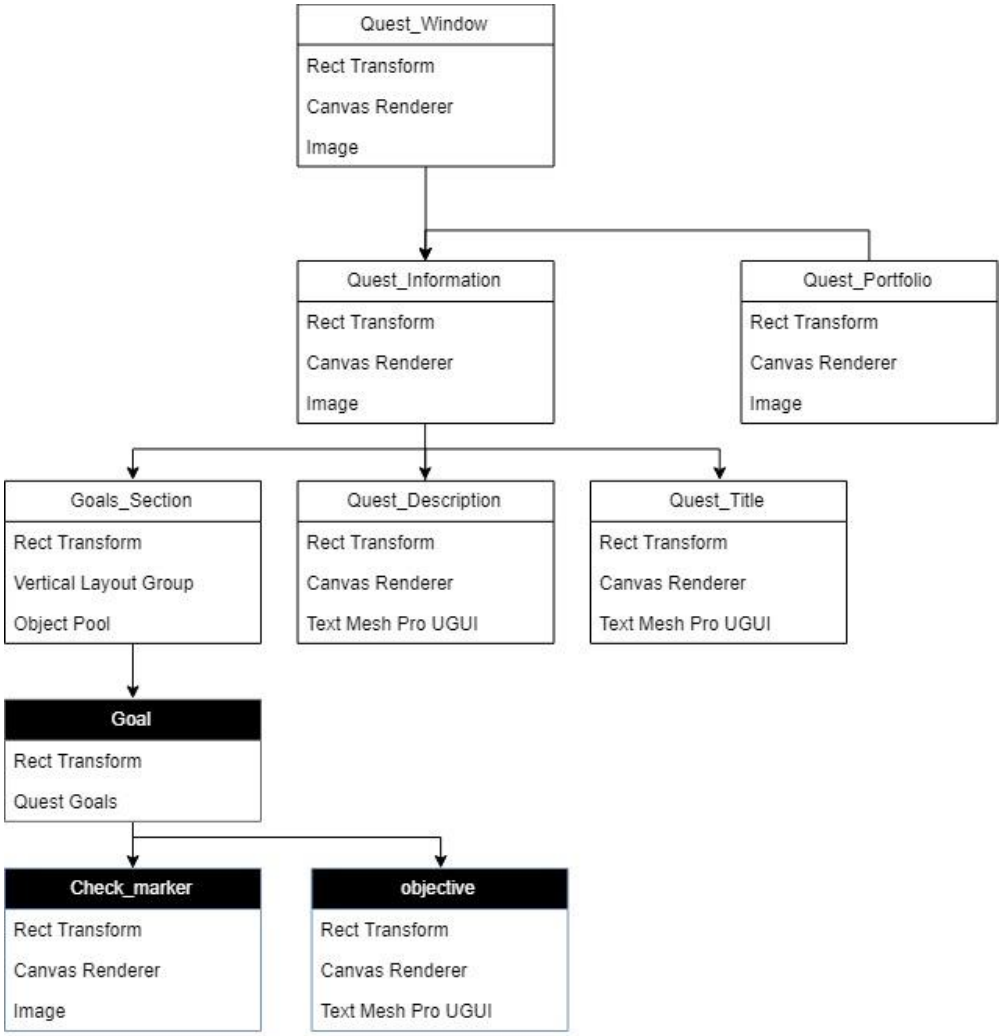

Figure 43: Quest Window GameObjects and respective Components

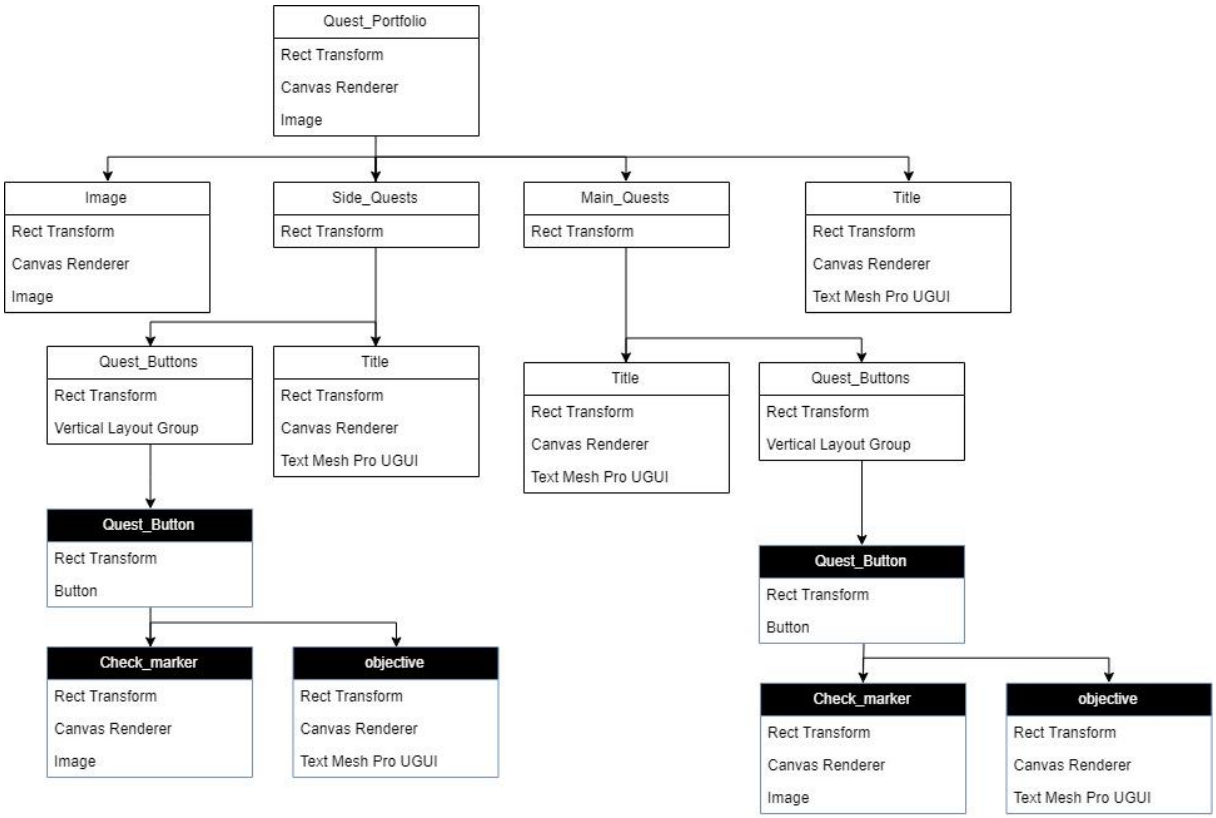

Figure 44: Quest Portfolio GameObjects and respective Components

Outside the main menu, players are able to access their inventory and check all items collected. The Inventory GO is composed of a panel with a grid layout group component that guarantees that all items are organized in an Inventory Slot grid. Each grid element is portraited as a button with a unique icon for each type of object stored, that allows players to distinguish them.

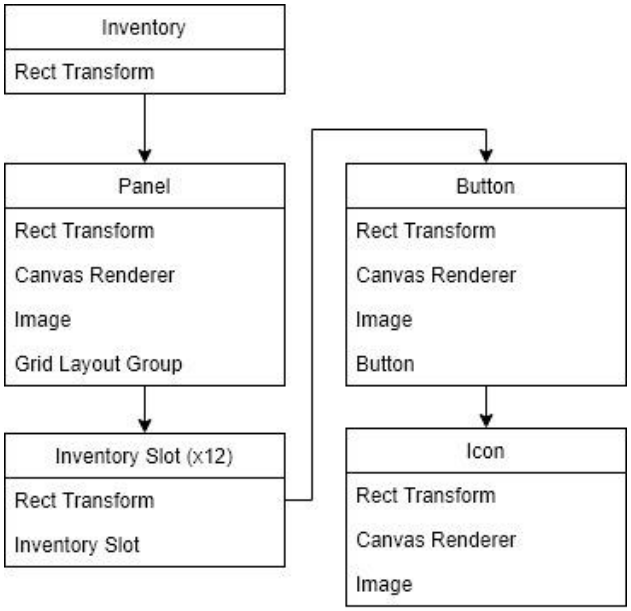

Figure 45: Inventory GameObjects and respective Components

The last parent GO of the Canvas is the Dialogue GO. Responsible for displaying all dialogue information, this GO includes all components present in a dialogue conversation, such as: the Icons GO that displays character portraits and swaps them accordingly; the Dialogue_Window that displays the names of the currently speaking character, the conversation content and a continue button prompt, informing players that it is possible to continue the dialogue; and the options menu, prompted whenever the dialogue leads to a situation that requires players to make a decision. The maximum number of options is limited at three at a time.

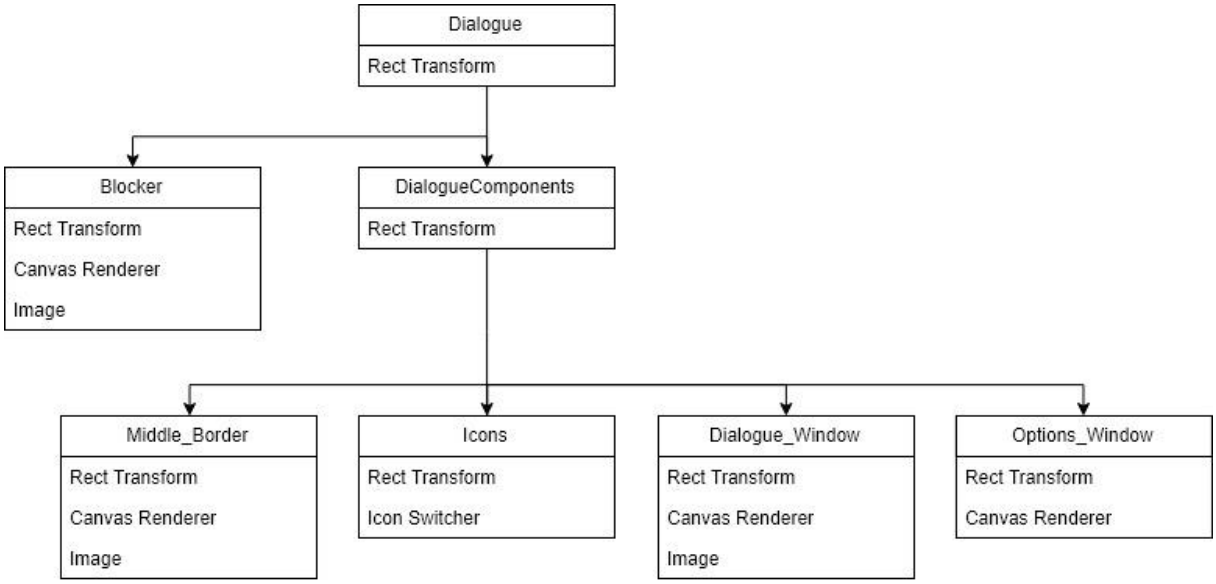

Figure 46: Dialogue GameObjects and respective Components

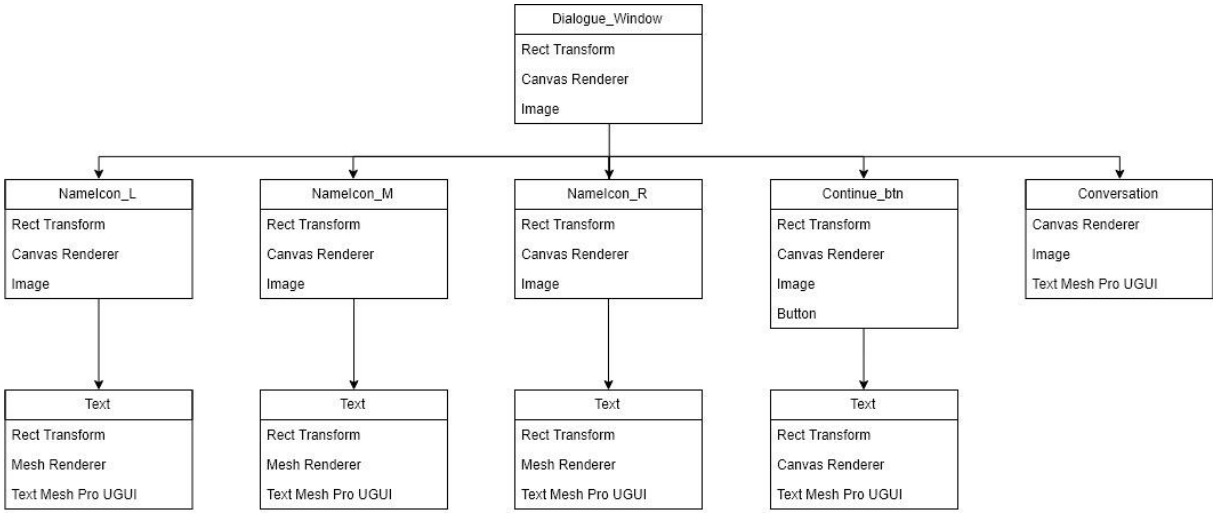

Figure 47: Dialogue Window GameObjects and respective Components

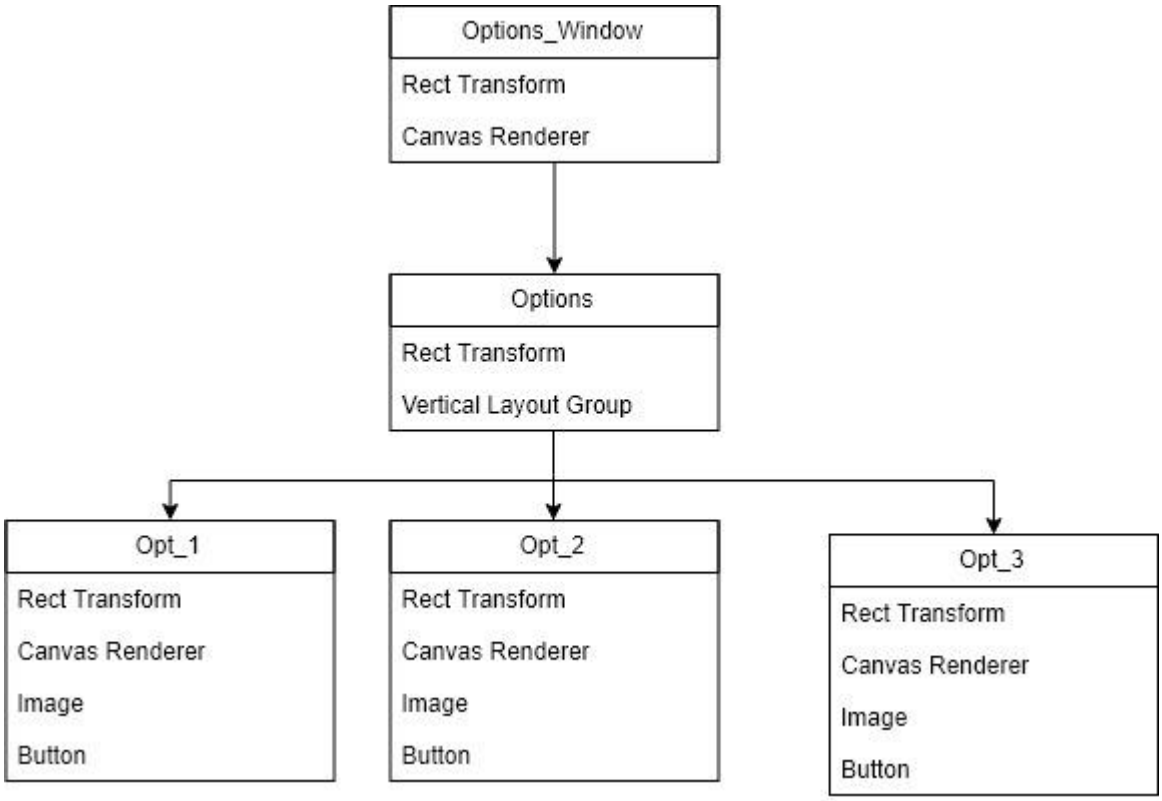

Figure 48: Options Window GameObjects and respective Components

There are three possible locations for the characters' icons, represented in the three different GO, the IconLeft, IconMiddle and IconRIght, where the middle one is usually reserved for external elements to a conversation, like a television, radio, etc.

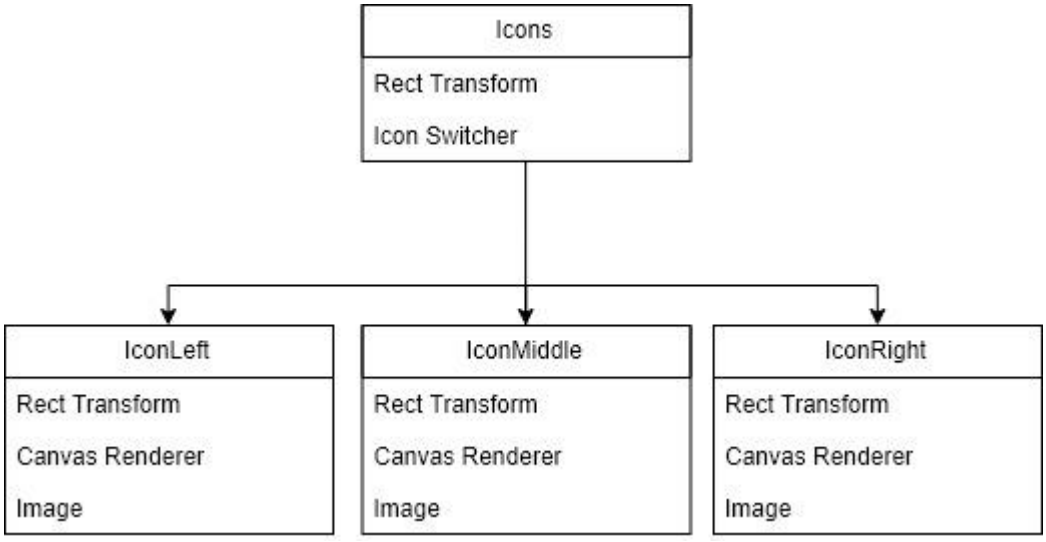

Figure 49: Icons GameObjects and respective Components

5.4 Game Behaviour

Being based on Unity, the game behaviour is event-driven. In Unity, there is a predefined order for event functions to be executed. Figure 50 represents a diagram of the order of the event functions used, followed by a detailed description of the procedures executed in each of those.

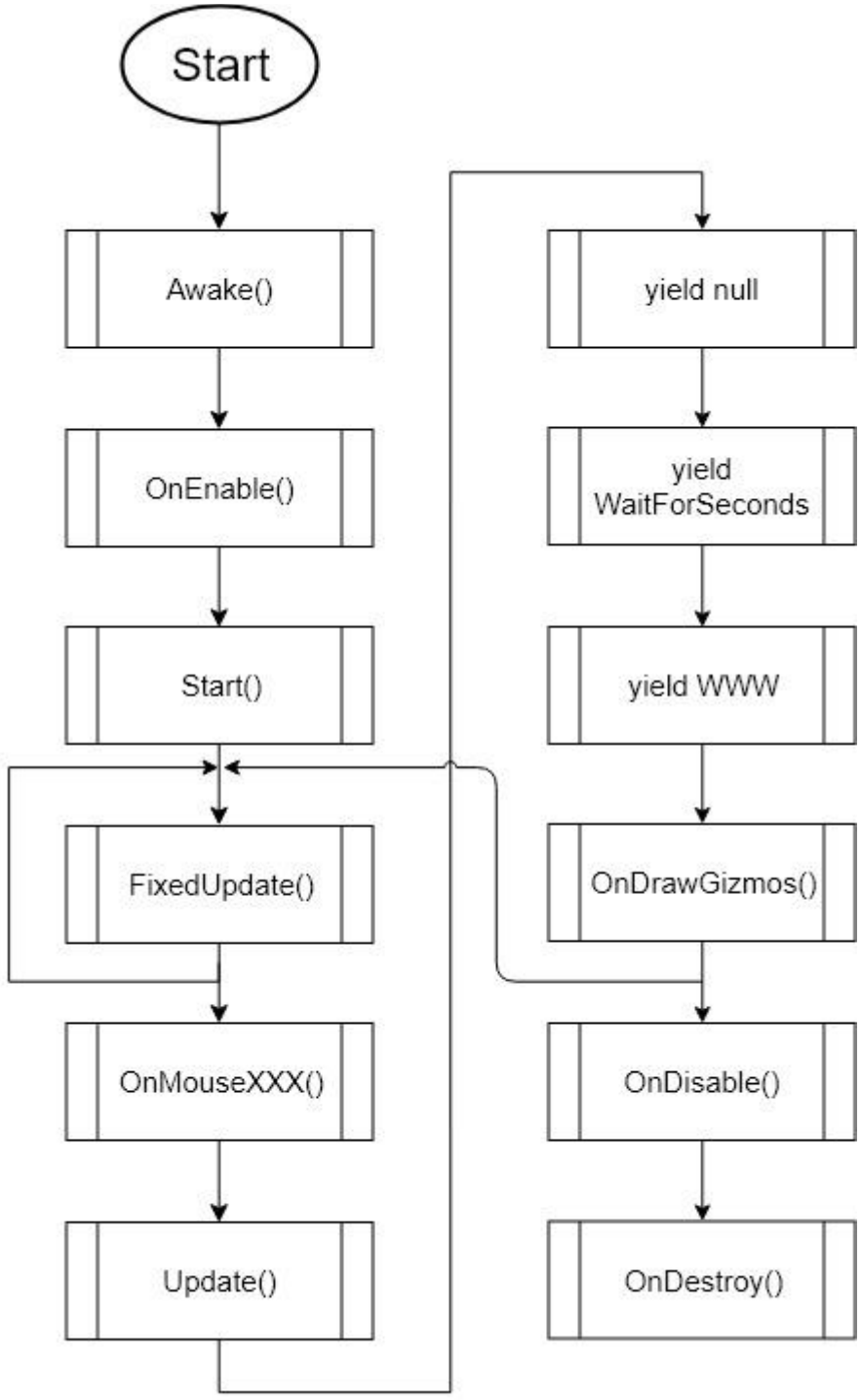

Figure 50: Order of the event functions

When a scene starts, the first functions to be called for each object on the scene are the Awake and OnEnable functions. The Awake functions will always run during start-up, unless the object is inactive. In

that case, the function only runs after the object is made active. The OnEnable function is called only if the object is active and after the level was loaded or the object instantiated.

Before the first frame, if a script instance is enabled, the Start function will run, immediately followed by the Update functions. The main difference between the FixedUpdate and the Update is that the latest is called once per frame, which varies depending on the machine the function is being run, while the former is called depending on a timer and independently from the frame rate, which allows for it to be called more than once per frame.

As for user input, OnMouse??? events are called every frame and act according to the position of the mouse and if it is over a GUI element or a Collider. There are three main OnMouse events, them being OnMouseEnter, OnMouseOver and OnMouseExit.

Yield functions are included inside coroutines that run after the Update function returns, and are able to suspend coroutines execution until a certain parameter is met: Yield null will continue its coroutine execution after the Update Functions have been called on the next frame; yield WaitForSeconds continues after a delay; and yieldWWW continues after the WWW function is completed.

When an Object is destroyed, the OnDestroy functions is called, but only after all frames updates for the last frame of the object's existence and OnDisable is called when an object becomes disabled or inactive.

OnDrawGizmos is used for drawing visual assistants when editing a scene.

5.4.1 Awake

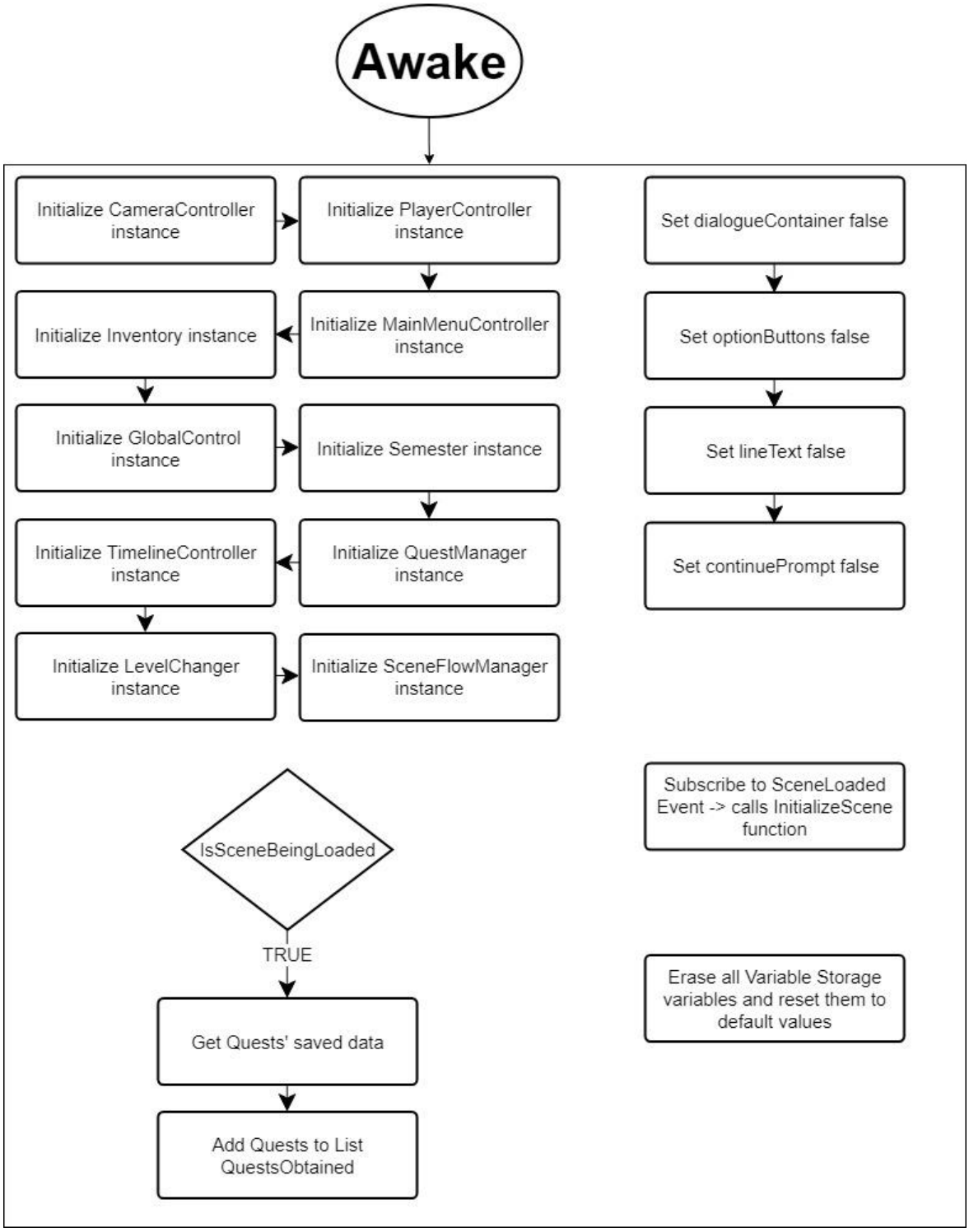

Figure 51: Tasks completed with Awake Event Function

5.4.2 Start

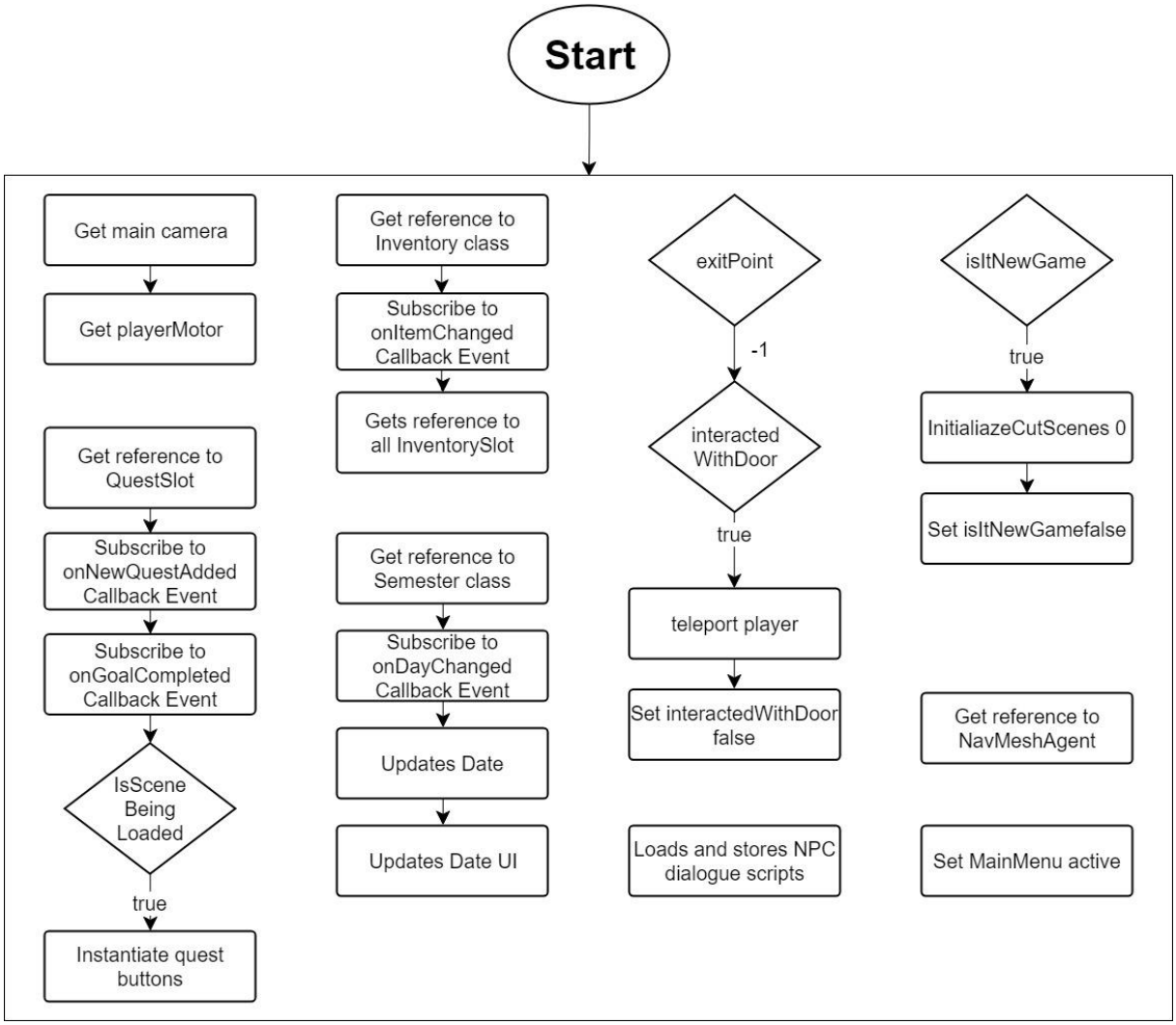

Figure 52: Tasks completed with Start Event Function

5.4.3 Update

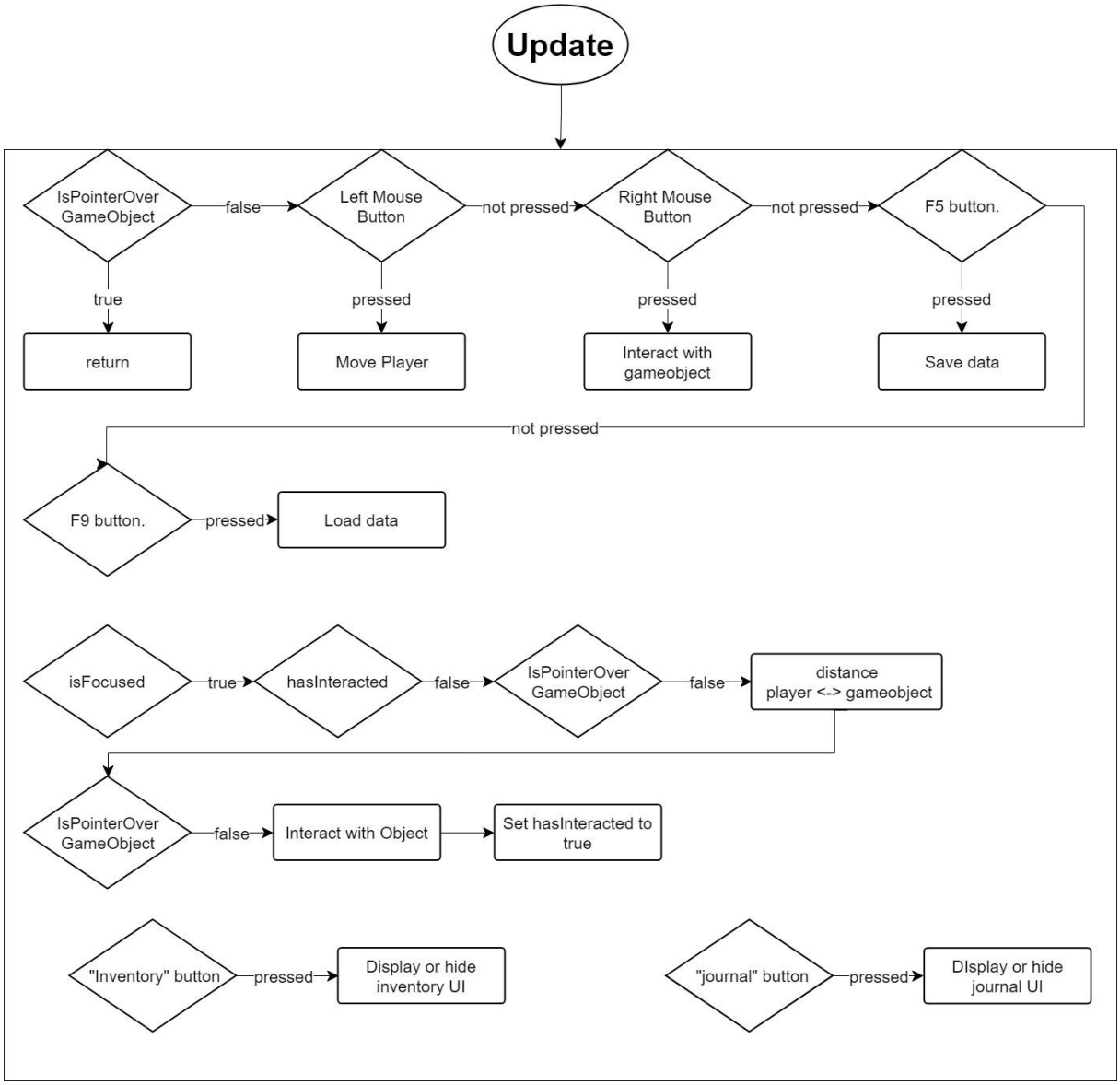

Figure 53: Tasks completed with Update Event Function

5.4.4 OnEnable, FixedUpdate, OnMouseEnter, OnMouseExit, OnDrawGizmosSelected, OnDisable, OnDestroy

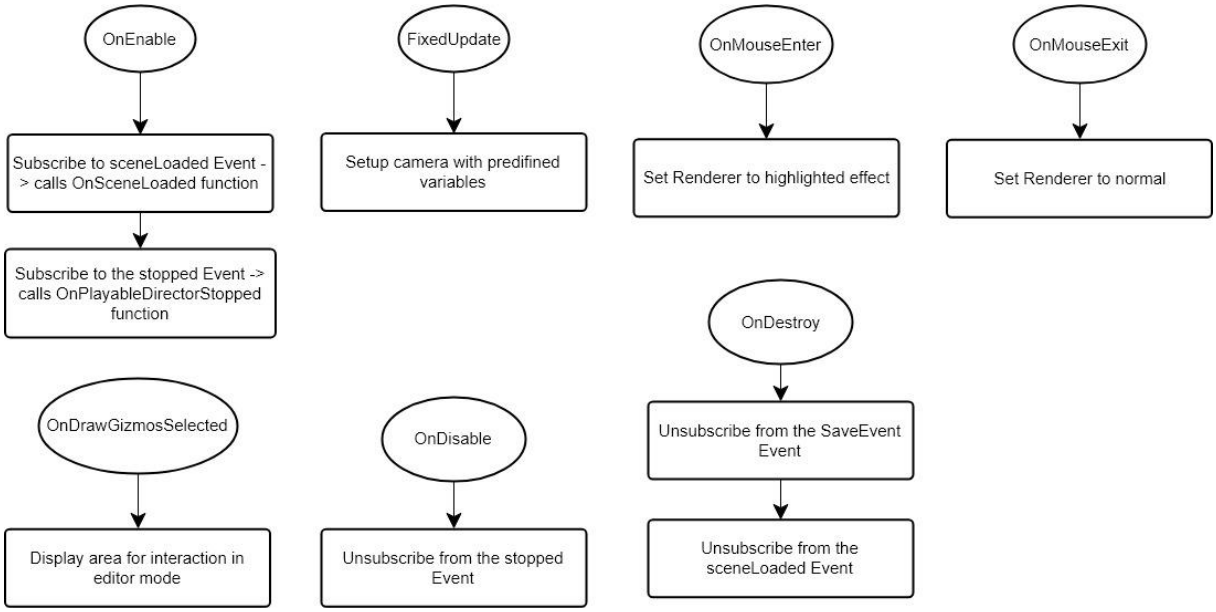

Figure 54: Tasks completed with OnEnable, FixedUpdate, OnMouseEnter, OnMouseExit, OnDrawGizmosSelected, OnDisable, OnDestroy Event Functions

5.5 Used Assets

Icons retrieved from www.flaticon.com made by different authors:

• Calendar and Cogwheel in Fig.12 and Battery in Fig. 12 from author, www.flaticon.com/authors/smashicons

• Hacker, Book and Magnifying Glass in Fig.12, Park in Fig. 12 and Wi-Fi and Home in Fig.12 from author, www.freepik.com

• Question Mark in Fig.12 from author, www.flaticon.com/authors/twitter

• Dices and Flags in Fig.18 from author, www.kenney.nl/assets

• Signal Strength and User icon in Fig.12 from author, www.flaticon.com/au-thors/dave-gandy

• Cloud in Fig. 16 and Fig. 63 originally from author, www.flaticon.com/authors/google with check mark colour modified from white to green

The fonts used are:

• Hacked, from http://watchdogsfont.com

• Fineness light, from https://fonts2u.com/fineness-light.font

• One CTT, from http://freakfonts.com/fonts/ocr-one-ctt.html

Character portraits were created by the artist “Elzee” and retrieved from https://lemmasoft.renai.us/forums/viewtopic.php?t=28377.

Furniture, UI panels, mini-game elements retrieved from www.kenney.nl/assets.

Character and character animations retrieved from https://www.mixamo.com.

6. RESULTS

6.1 Game Description

Although most of the gameplay was built with a single camera angle that follows the main character around the different scenarios, cutscenes will often use various different camera angles that aim to increase immersion and force players to focus on certain aspects of the scenario.

Figures 55, 56, 57 and 58 are examples of these different views. First, in figure 55, the camera is centered on the main character, but zoomed out enough for players to be able to easily discern the environment surrounding them. Figure 56 shows a zoomed-in camera, used during a dialogue to highlight and focus attention on the conversation participants. Figure 57 is the zoomed-out version of the same camera angle, intending to direct player's attention to the arrival of a new character, immediately zooming-in to the previous state, this time with a new conversation participant, as depicted in figure 58.

Figure 55: Gameplay Camera

Figure 56: Cutscene Camera during dialogue

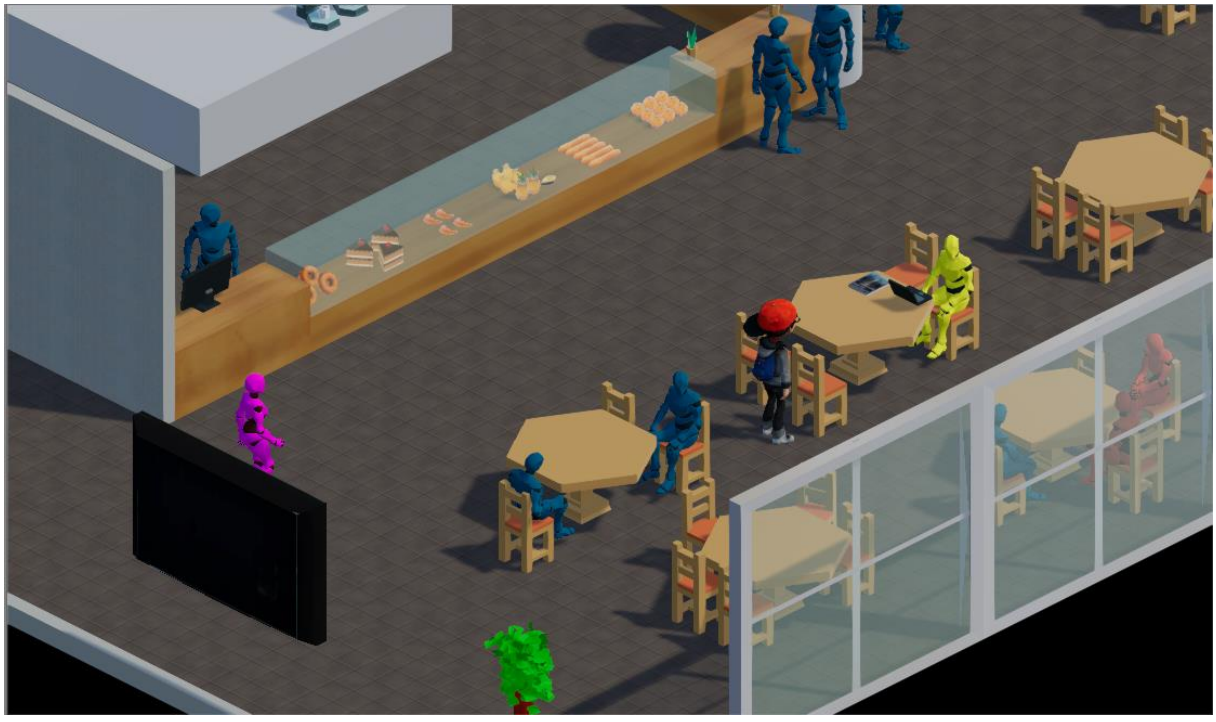

Figure 57: Cutscene Camera zoom out

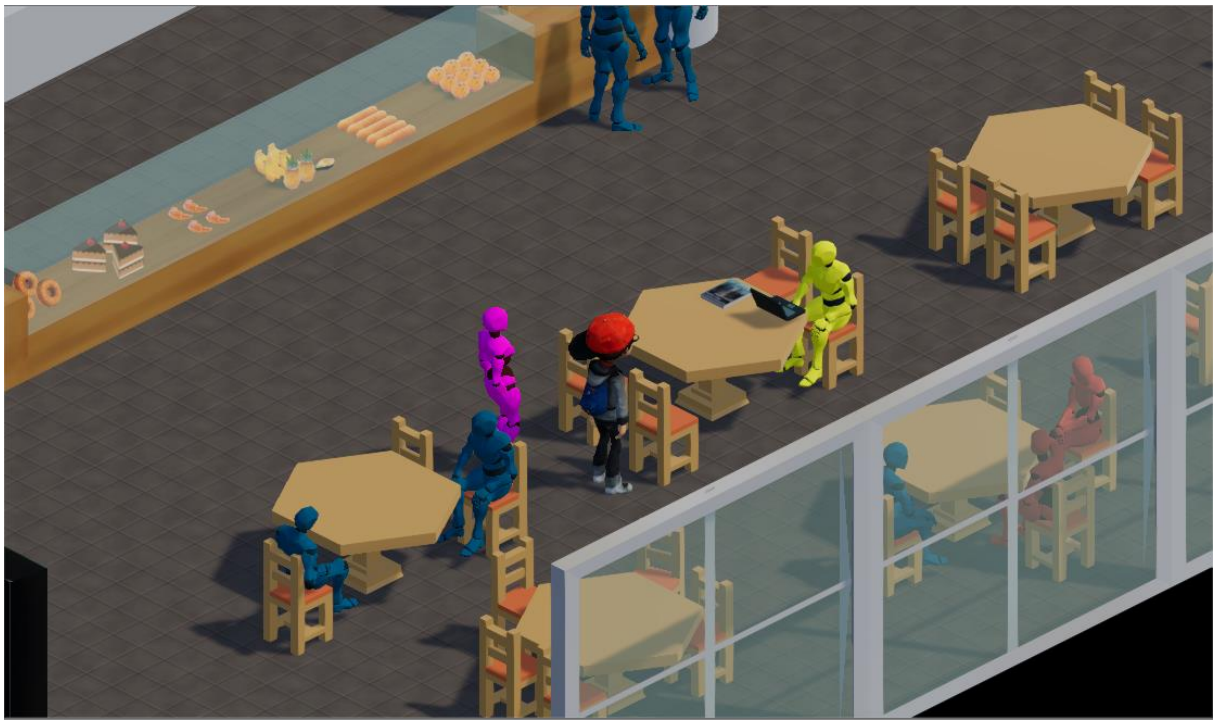

Figure 58: Cutscene Camera zoom in

The start of the game immediately engages players in a dialogue with another NPC, combining the relaxed conversation, with the visual representation of other students on various tables and chairs, to lead players to easily recognize the environment as a Bar and the time as being the beginning of a new semester.

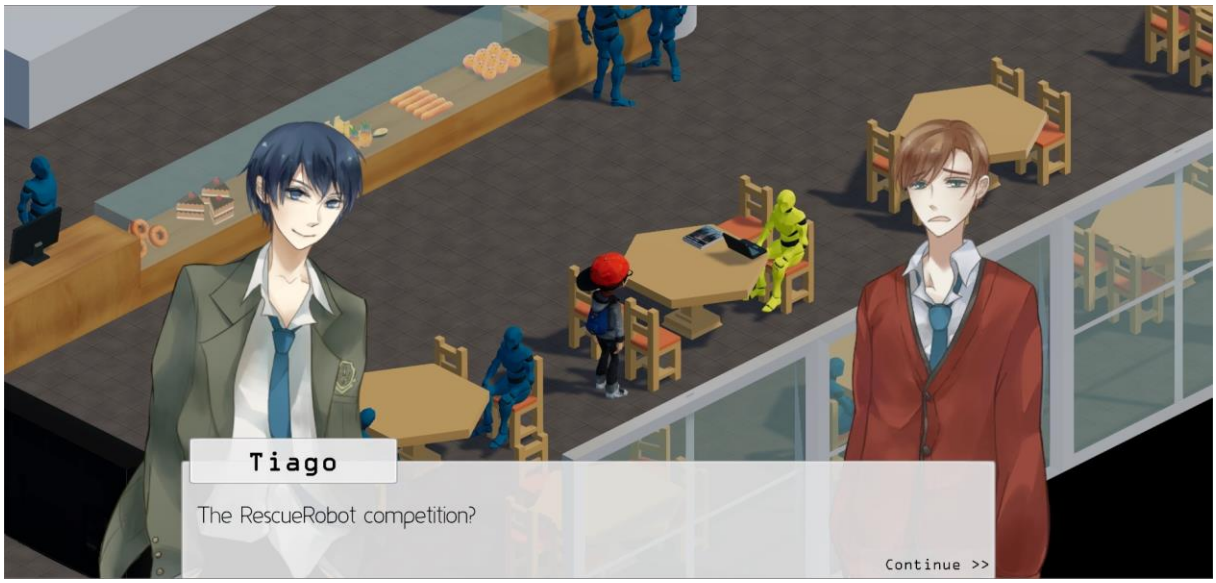

Figure 59: Conversation between the main character (left) and a friend (right)

After chatting for a while both characters are interrupted by a breaking news television broadcast, whose contents involve multiple statistical concepts. Both characters engage in a discussion about the news, explaining and question some of the concepts mentioned, providing an example of the educational mechanics in the game.

Figure 60: Dialogue external intervention

After the discussion, the characters are again interrupted, but this time by the arrival of a new character, shown using different camera zooms in a cutscene to emphasize its arrival.

Figure 61: Example of a Dialogue

Nearing the end of the conversation, players are able to choose to either help the request of their friend to buy some lunch tickets or simply ignore it (figure 62). By accepting it, players are provided with a side quest, updated and displayed in the quest window as represented in figure 63, directing them to buy the lunch tickets. This choice is followed by the newcomer character leaving the scene to attend a class.

Figure 62: Dialogue options

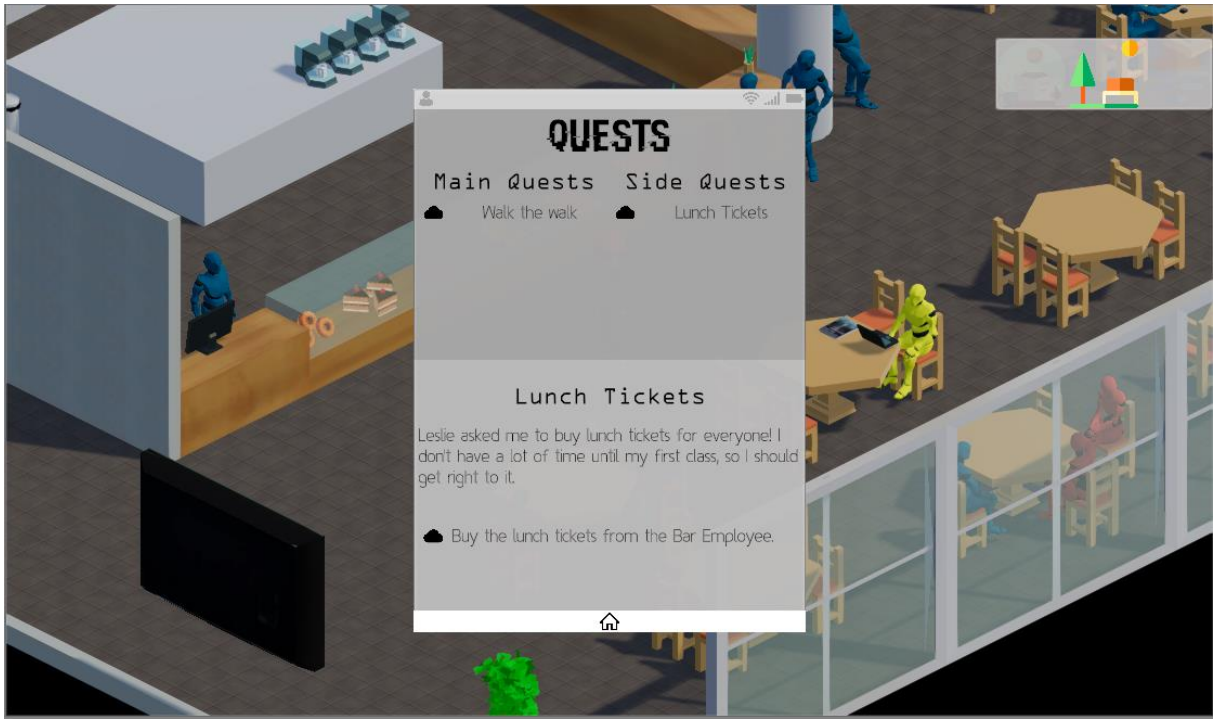

Figure 63: Quest Window

With the initial dialogue finished, the player is in control of her/his character and is assigned with a main quest for reaching a designated classroom, in order to progress with the main story.

Between their starting point and the quest's target location, the player is able to explore and interact with the environment around her/him and, as figure 64 shows, this may lead to finding new side quests along the way, provided by NPCs signalled with a white border each time player's mouse hover over them.

The main quest currently implemented requires players to walk towards a specific classroom on the "nave" scenario and the side quests include the search for a student ID and buying lunch tickets.

Figure 64: Quest Giver NPC highlight

Some of this side quests, often require the player to search the surrounding environment, looking for specific items. As an example, one of the NPCs provides players with the side quest to find her/his missing student ID card.

As seen in figures 65 and 66, by looking around, players are able to quickly find a misplaced item in the environment. By interacting with it, the player adds the item to the inventory. At the moment players are able to gather a student's ID card and lunch ticket, both of which allow players to complete the side quests mentioned above.

Figure 65: Interactable in a scene

Figure 66: Players' Inventory

During main quests, the game focuses players attention on them, by either simulating apprehension or motivating players to quickly finish that quest. This is accomplished by providing feedback and reminders at certain stages and locations. Example of this are both figure 67 and 68, where the former is displayed the first-time players walk into the correct area to reach the destination of their quest, providing further insight on where they need to go, for example when players walk into a hall and have the quest to reach a specific classroom, located along that same hall.

Figure 68 corresponds to feedback provided to keep players on track and pressure them to finish the currently active quest, displayed every time they leave and re-enter the destination area.

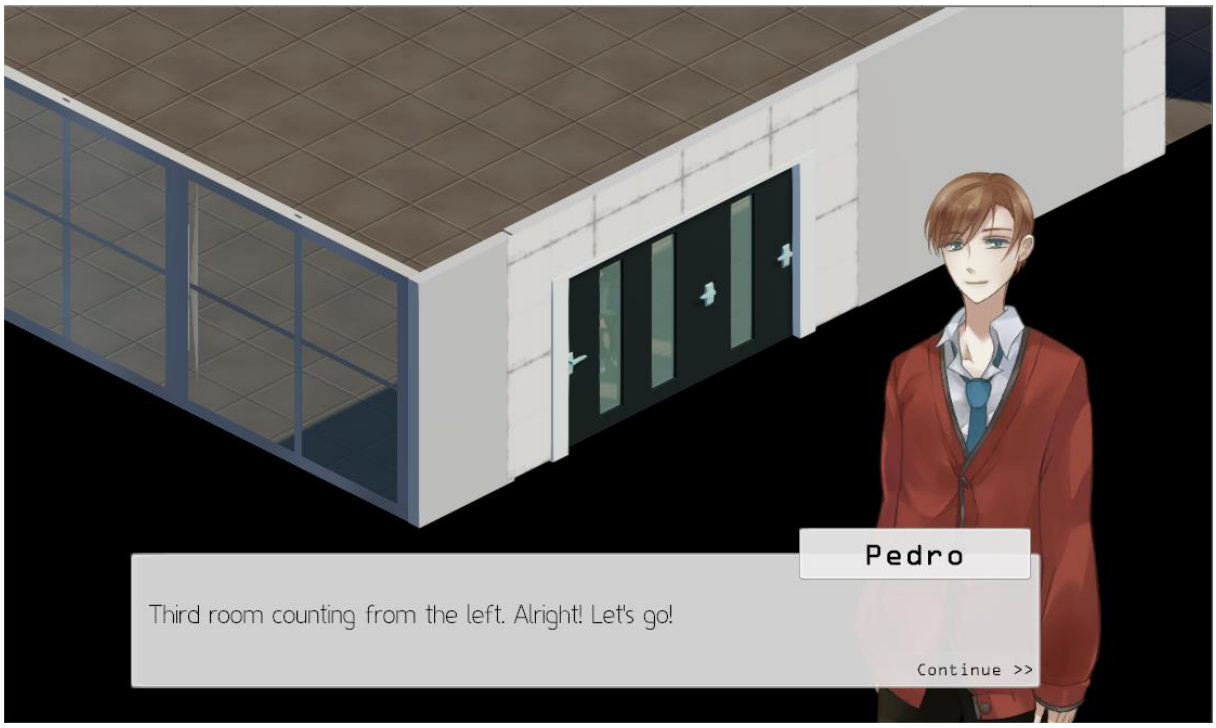

Figure 67: Quest Feedback

Figure 68: Quest Feedback on exiting and re-entering the destination area

A third feedback type, exemplified in figure 69, is provided when players are with a main quest active and try to access other areas that would diverge from the main quest's path. This prevents players from losing track of their objective.

It is important to mention that these restrictions are only applied in certain quests and in most cases, players are able to roam the entire map freely.

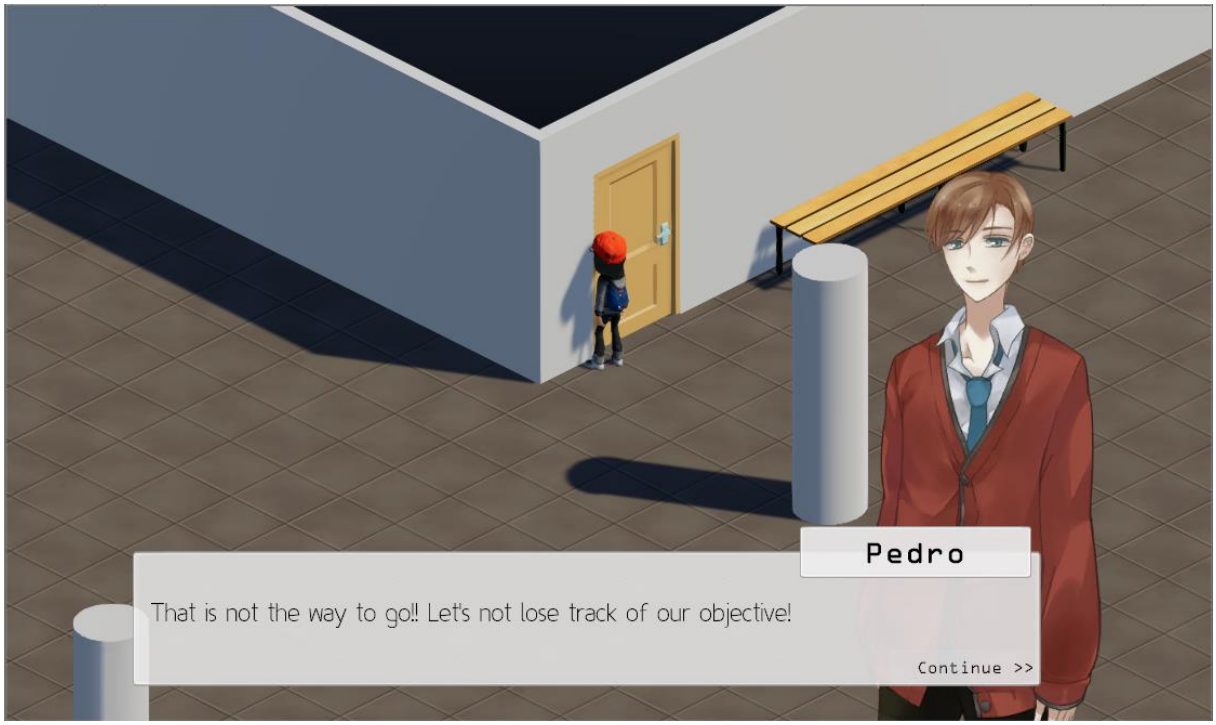

Figure 69: Quest Feedback to keep players on track

After reaching their main quest's location, players are presented with a black screen informing them of a time leap to after the class, as shown in figure 70.

Figure 70: Time leap screen

The main character is then again reunited with all of his friends and is questioned about the lunch tickets. Figure 71 and 72, show different reactions and answers, depending on the previous decision of players on buying or not the tickets.

Figure 71: Failing a side quest

Figure 72: Finishing a side quest

7. CONCLUSION

As far as it was possible to find during the search for Serious Games on Statistics, all of them target the middle and high school levels of education. The idea for this project was born from the lack of Serious Games that focus on the teaching of statistical concepts in higher education.

This dissertation addresses educational principles that are the foundation for most successful techniques and processes used to captivate and guarantee students' attention. More in depth contributions helping with the development of Serious Games are frameworks that identify different (game) aspects and elements, and how they should be combined to build effective games.

The most prominent frameworks are presented, as well as proof of their valuable insight for the development process of the game and the balancing of the Serious Games' mechanics and educational purposes. Besides this, there are different types of SGs in what concerns (1) the game flow and activities involved and (2) how the player interacts with the game. For each one of these aspects, a type of serious game is reviewed, namely Adaptive and Sand Box Serious Games. Examples of these two types of SGs, found in the literature and on topics closer to Statistics, are provided.

The proposed serious game was designed based on this theoretical background, starting with the main idea for the game flow and the context of action. The Statistics syllabus to be addressed was also defined and it concerns the context of a course on statistics for the Master on Industrial Electronics and Computers Engineering of the host university. The context of action was then refined into a general plot of the storyboard, including the activities involved throughout the four phases of a typical narrative. Finally, the game mechanics was defined, identifying the core components of the game software architecture.

A game software application was developed based on Unity. It includes the implementation of the game mechanics and game objects. The implemented scenes and supported activities cover parts of the exposition phase, including the opening cut-scene, introduction of environment and mechanics, and a mini-game on probability.

The next couple of sections tackle the core of the project, that is the Game Design and Implementation, where the former is the arrangement and explanation of the game's theoretical features, concepts and purpose and the latest is a more practical explanation of the software implemented.

Finally, the results are presented as a scene, where each step and decision of the player is explained based on the implemented features.

7.1 Conclusions

The development of a serious game is a challenging project. It involves and requires many different skills, usage of many different tools and technologies and a large volume of work. Therefore, it can hardly be a one man's endeavour and at least a small group of people is necessary to tackle the just mentioned aspects. For this reason, it was not possible to come up with a playable game.

The objectives were fulfilled, with the complete development of a prototype scene. The entire narrative and scenario design creative processes are in their initial stages and a focal point for future works. On a technical standpoint, it was possible to develop most core mechanics of the game, where mini-games stand as the only mechanic not completely built. This kind of mechanics require merging the Statistics topics with mini-games targeting the intended learning outcomes.

The results of this work are twofold: an implementation of the first parts of the game and a game design featuring all aspects necessary to start further implementations. The first implemented parts are the inventory, quest, daily, interactions, scene and UI systems, and a prototype scene with all mechanics working together. The work developed is a solid foundation for further developments of the game. Specifically, the game structure and mechanics is defined, the core software architecture is implemented, and the start of the storyboard is implemented and serves as an example for other structural elements (scenes, dialogues, mini-games, etc.).

7.2 Future Work

Future work may include some improvements on the currently implemented scripts are possible, mostly related to optimizations and increasing performance. The most important is to build on the game design base. Additional implementations include the creation of scenarios for the defined context, detail the narrative by creating dialogues for the main plot, development of mini-games on the identified topics and their integration on the game flow.

After finishing the development of the SG, experimental tests and reviews should be made, in order to update the game on the feedback received and check its effectiveness as a learning tool.

REFERENCES

[1] M. Callaghan, M. Savin-Baden, N. McShane, and A. Gomez Eguiluz, "Mapping Learning and Game Mechanics for Serious Games Analysis in Engineering Education," *IEEE Trans. Emerg. Top. Comput.*, vol. 5, no. 1, pp. 77–83, 2017.

[2] A. Ypsilanti, A. B. Vivas, T. Räisänen, M. Viitala, T. Ijäs, and D. Ropes, "Are serious video games something more than a game? A review on the effectiveness of serious games to facilitate intergenerational learning," *Educ. Inf. Technol.*, vol. 19, no. 3, pp. 515–529, Sep. 2014.

[3] L. Pannese and M. Carlesi, "Games and learning come together to maximise effectiveness: The challenge of bridging the gap," *Br. J. Educ. Technol.*, vol. 38, no. 3, pp. 438–454, 2007.

[4] K. Corti, "Games-based Learning: a serious business application," *Inf. PixelLearning*, vol. 34(6), pp. 1–20, 2006.

[5] C. P. Leao, F. Soares, V. Carvalho, S. Lopes, and I. Gonçalves, "A serious game concept to enhance students' learning of statistics," *Proc. 2017 4th Exp. Int. Conf. Online Exp. exp.at 2017*, pp. 187–190, 2017.

[6] M. Ulicsak, "Games in Education: Serious Games," *A Futur. Lit. Rev.*, p. 139, 2010.

[7] M. Popescu *et al.*, "Serious Games in Formal Education: Discussing Some Critical Aspects," *Proc. 5Th Eur. Conf. Games Based Learn.*, pp. 486–493, 2011.

[8] C. Girard, J. Ecalle, and A. Magnan, "Serious games as new educational tools: How effective are they? A meta-analysis of recent studies," *J. Comput. Assist. Learn.*, vol. 29, no. 3, pp. 207–219, 2013.

[9] E. A. Boyle *et al.*, "A narrative literature review of games, animations and simulations to teach research methods and statistics," *Comput. Educ.*, vol. 74, pp. 1–14, 2014.

[10] E. A. Boyle *et al.*, "An update to the systematic literature review of empirical evidence of the impacts and outcomes of computer games and serious games," *Comput. Educ.*, vol. 94, pp. 178–192, 2016.

[11] H. A. Rosyid, M. Palmerlee, and K. Chen, "Deploying learning materials to game content for serious education game development: A case study," *Entertain. Comput.*, vol. 26, no. March 2017, pp. 1–9, 2018.

[12] D. Buchinger and M. da Silva Hounsell, "Guidelines for designing and using collaborative-competitive serious games," *Comput. Educ.*, vol. 118, no. February 2017, pp. 133–149, 2018.

[13] R. Nadolski *et al.*, "<Emergo - methodology and toolkit for efficient development of serious games in higher education .pdf>," pp. 1–11.

[14] M. Barr, "Student attitudes to games-based skills development: Learning from video games in higher education," *Comput. Human Behav.*, vol. 80, pp. 283–294, 2018.

[15] J. M. Harackiewicz and S. J. Priniski, "Improving Student Outcomes in Higher Education: The Science of Targeted Intervention," *Annu. Rev. Psychol.*, vol. 69, no. 1, p. annurev-psych-122216-011725, 2018.

[16] A. Mitchell and C. Savill-Smith, *The use of computer and video games for learning: A review of the literature*. 2004.

[17] C. P. Leao, "What engineering students tell us about to know mathematics and statistics in their courses?," *IEEE Glob. Eng. Educ. Conf. EDUCON*, vol. 10–13–Apri, no. April, pp. 1221–1224, 2016.

[18] G. a Gunter, R. F. Kenny, and E. H. Vick, "A Case for a Formal Design Paradigm for Serious Games," *J. Int. Digit. Media Arts Assoc.*, vol. 3, no. 1, pp. 1–19, 93–105, 2006.

[19] S. Abdelali, S. Mateu, B. Imma, E. Fatiha, and B. Mohammed, "Improving serious game design through a descriptive classification: A comparation of methodologies," *J. Theor. Appl. Inf.*

Technol., vol. 92, no. 1, pp. 130–143, 2016.
[20] M. D. Kickmeier-Rust and D. Albert, "Micro-adaptivity: Protecting immersion in didactically adaptive digital educational games," *J. Comput. Assist. Learn.*, vol. 26, no. 2, pp. 95–105, 2010.
[21] D. Gloria, "Games and Learning Alliance," vol. 10653, 2017.
[22] F. Bellotti, R. Berta, A. De Gloria, and L. Primavera, "A task annotation model for SandBox Serious Games," *CIG2009 - 2009 IEEE Symp. Comput. Intell. Games*, pp. 233–240, 2009.
[23] K. Kruse, "Gagne's Nine Events of Instruction: An Introduction Instructional Event Internal Mental Process," pp. 1–4, 2010.
[24] M. Forehand, "BloomsTaxonomy.pdf," *Bloom's Taxanomy-Emerging Perspective on Learning, Teaching and Technology*. p. 10, 2011.
[25] Cammy Bean, "A Shot of Theory - Keller's ARCS Model," 2009. [Online]. Available: http://www.kineo.com/resources/top-tips/learning-strategy-and-design/a-shot-of-theory-kellers-arcs-model. [Accessed: 16-Jul-2018].
[26] D. L, "ARCS Model of Motivational Design Theories (Keller)," 2014. [Online]. Available: https://www.learning-theories.com/kellers-arcs-model-of-motivational-design.html. [Accessed: 16-Jul-2018].
[27] M. J. Callaghan, N. Mcshane, A. G. Eguíluz, T. Teillès, and P. Raspail, "Practical Application of the Learning Mechanics – Game Mechanics (LM-GM) framework for Serious Games Analysis in Engineering Education," no. February, pp. 382–386, 2016.
[28] B. M. Winn, "The Design, Play, and Experience Framework," *Handb. Res. Eff. Electron. Gaming Educ.*, vol. 5497, pp. 1010–1024, 2008.
[29] G. A. Gunter, R. F. Kenny, and E. H. Vick, "Taking educational games seriously: Using the RETAIN model to design endogenous fantasy into standalone educational games," *Educ. Technol. Res. Dev.*, vol. 56, no. 5–6, pp. 511–537, 2008.
[30] H. Zhang, X. Fan, and H. Xing, "Research on the design and evaluation of educational games based on the RETAIN model," *Knowl. Acquis. Model. (KAM), 2010 3rd Int. Symp.*, no. 2009, pp. 375–378, 2010.
[31] Sanderson Smith, "SOPHISTICATED PROBABILITY WITH SIMPLE DICE GAME." [Online]. Available: http://www.herkimershideaway.org/writings/cmlink.htm. [Accessed: 15-Jun-2018].
[32] U. Technologies, "ProBuilder." [Online]. Available: https://assetstore.unity.com/packages/tools/modeling/probuilder-111418.
[33] U. Technologies, "ProGrids." [Online]. Available: https://assetstore.unity.com/packages/3d/progrids-111425.
[34] U. Technologies, "Cinemachine." [Online]. Available: https://assetstore.unity.com/packages/essentials/cinemachine-79898.
[35] U. Technologies, "TextMesh Pro." [Online]. Available: https://assetstore.unity.com/packages/essentials/beta-projects/textmesh-pro-84126.