Faculdade de Engenharia da Universidade do Porto

Exploring Player Adaptivity through Level Design: A Platformer Case Study

Pedro Miguel Rodrigues Ferraz Esteves

Mestrado em Engenharia Informática e Computação

Supervisor: João Jacob

Second Supervisor: Rui Rodrigues

July 31, 2022

Exploring Player Adaptivity through Level Design: A Platformer Case Study

Pedro Miguel Rodrigues Ferraz Esteves

Mestrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Daniel Castro Silva
External Examiner: Nuno Rodrigues
Supervisor: João Jacob

July 31, 2022

Abstract

Different people experience distinct feelings while playing the same game. For instance, some players like action-heavy games, while others prefer puzzles. Creating a game that engages everyone is not trivial and, ultimately, impossible. However, generating content adapted to the players could improve their experience, thus maximizing the potential player base.

Therefore, a game designer's role can be challenging when dealing with games with static content. With adaptive games, these can be designed to act differently according to several actions without developing different versions. Although this approach seems to solve all the designers' problems, creating these games can be very time-consuming, increasing the development time and budget. Besides, creating a generic approach that adapts content for every game genre or even different games inside the same genre is not trivial.

To tackle this problem, this work proposes a methodology to explore player adaptivity through level design that follows three phases: (i) the selection of a genre and a game, (ii) a first study to correlate players' profiles, preferences, and actions with single elements, and (iii) a second study, based on the previous, to evaluate the impact of generating and adapting game levels.

Using a version of a well-known platform game, *Super Mario Bros.*, the first results show it is not possible to correlate players' profiles with game elements. However, several preferences for levels with specific configurations were spotted and used for the second study's adaptation. Nevertheless, these adaptations did not sufficiently impact the players' overall experience in the second study.

Although the results do not seem encouraging, other deeper analyses of the study's data should be performed in the future using distinct algorithms and machine learning. Moreover, the same methodology should be applied to other games to ensure similar results are obtained. Last but not least, a study should be conducted to explore correlations between players' profiles and groups of game elements, especially obstacles around gaps.

Keywords: Game Adaptivity, Procedural Content Generation, Level Design, Player Profiling, Player Personality

Resumo

Pessoas diferentes experienciam sentimentos distintos quando jogam o mesmo jogo. Por exemplo, alguns jogadores gostam mais de jogos de ação, enquanto outros preferem puzzles. Criar um jogo que envolva todos não é trivial, e, em último caso, impossível. No entanto, gerar conteúdo adaptado aos jogadores poderia melhorar a sua experiência, maximizando assim a potencial base de jogadores.

O papel de um designer de jogos pode, portanto, ser uma tarefa difícil quando este lida com jogos com conteúdo estático. Com jogos adaptativos, estes podem ser concebidos para agir de forma diferente de acordo com várias acções, sem desenvolver versões diferentes. Embora esta abordagem pareça resolver todos os problemas dos designers, criar estes jogos pode ser muito demorado, aumentando o tempo de desenvolvimento e o seu orçamento. Além disso, criar uma abordagem genérica que adapte o conteúdo de cada género de jogo ou mesmo jogos diferentes dentro do mesmo género não é trivial.

Para abordar este problema, este trabalho propõe uma metodologia que explora a adaptividade ao jogador através da conceção de níveis que segue três fases: (i) a selecção de um género e de um jogo, (ii) um primeiro estudo para correlacionar os perfis, preferências e acções dos jogadores com elementos únicos, e (iii) um segundo estudo, com base no anterior, para avaliar o impacto da geração e adaptação dos níveis de jogo.

Utilizando uma versão de um conhecido jogo de plataforma, *Super Mario Bros.*, os primeiros resultados mostram que é não é possível correlacionar os perfis dos jogadores com os elementos do jogo. No entanto, várias preferências por níveis com certas configurações foram detetadas e utilizadas na adaptação do segundo estudo. No entanto, estas adaptações não tiveram impacto suficiente na experiência global dos jogadores no segundo estudo.

Embora os resultados não pareçam encorajadores, outras análises mais profundas dos dados do estudo devem ser realizadas no futuro utilizando outros algoritmos e aprendizagem computacional. Além disso, a mesma metodologia deve ser aplicada a outros jogos para garantir a obtenção de resultados semelhantes. Por último, mas não menos importante, deve ser realizado um estudo de modo a explorar correlações entre os perfis dos jogadores e grupos de elementos de jogo, especialmente obstáculos próximos a buracos.

Keywords: Adaptividade de Jogo, Geração Procedimental de Conteúdo, Conceção de Níveis, Perfil de Jogador, Personalidade de Jogador

Acknowledgments

As this thesis represents not only the end of a Master's degree but also the end of a challenging and exhausting period of my life, I would love to express my gratitude to everyone who participated directly and indirectly in it.

Firstly, thanks to my family for supporting me all my life and for staking in my dreams. A special thank you to my mother, who taught me how to play games when I was only three years old. Without her, this work would never have been done.

To my girlfriend, thank you for all the support, the encouragement, and overall for making me a better person. Completing this thesis and degree would have been much more difficult without her.

To my friends, thanks for the good times we have been through and all the games we played together. These moments of relaxation made this degree easier and helped me outperform my expectations.

To my supervisors, João Jacob and Rui Rodrigues, thank you for sharing your knowledge and guiding this work over the past year. I would also like to thank Pedro Evangelista, who has provided some great insights on how to analyze the data.

Last but not least, I would like to thank everyone who participated in both surveys conducted. Your time and comments contributed a lot.

Pedro Esteves

"It always seems impossible
until it's done."

Nelson Mandela

Contents

List of Figures

List of Tables

Listings

Abbreviations

DDA	Dynamic Difficulty Adjustment
EDPCG	Experience-driven Procedural Content Generation
EDRL	Experience-driven Procedural Content Generation via Reinforcement Learning
GDCML	Game Design via Creative Machine Learning
GEQ	Game Experience Questionnaire
GUID	Global Unique Identifier
IMB	Infinite Mario Bros
JS	JavaScript
JSON	JavaScript Object Notation
PCG	Procedural Content Generation
PCGML	Procedural Content Generation via Machine Learning
PEM	Player Experience Modeling
RL	Reinforcement Learning

Chapter 1

Introduction

The main goal of game designers is to create experiences. In the book *The Art of Game Design: A Book of Lenses*, Schell states that "without the experience, the game is worthless" [49, p. 10]. People play games because of experiences, so game designers need to know the preferences of their audience, listen to the players and "become intimate with their thoughts, their emotions, their fears, and their desires" [49, p. 116]. Designers would create games targeting individuals in an ideal world since each is unique. Although this is entirely impossible in the real world, there is a way to get as close as possible to this ideal world called Game Adaptivity.

This thesis explores player-oriented adaptivity by researching methods of adapting a level's design to players' profiles, focusing on their personalities. This chapter exposes this problem, showing its context and the motivation to solve it, followed by a discussion of this thesis' scope. Then, the research questions, the objectives, and the contributions of the work are described. Lastly, the document structure is analyzed, guiding the reader through the text.

1.1 Context and Motivation

Game Designers usually create experiences for a specific target audience, which can lead to the failure of some projects because even inside the chosen target group, people still have distinct experiences, and the "experience is all we [the game designers] care about" [49, p. 304]. Their job can be difficult when dealing with games with **static elements**, i.e., games where the content is predefined by designers, maintaining it testable and controllable, as mentioned by Lopes and Bidarra [37].

On the other hand, there are **adaptive games**, i.e., games with some adaptation to players, either through content creation adapted to their profiles or in adjusting parameters or content to these profiles [37]. With this approach, game designers can improve the players' experience and increase the target audience.

Although adaptive games seem to answer some of the designers' problems, creating them can be very time-consuming, increasing the planning and development time and the game's budget, as developers need to create **models** or **profiles** of players.

Since each individual is unique, game designers need a way to group players, which can be achieved through **player profiling**, "the categorization of players based on static information that does not alter during gameplay" [61, p. 2]. People can be grouped, for example, by age range or personality since some human beings share traits like extraversion or openness. These age or personality groups could be related to certain in-game behaviors or preferences for elements. For instance, users more open to experiences probably explore more of the game's map. However, not enough research relates these behaviors with the said personality traits.

Ultimately, automatically generating content adapted to the players could help game designers in the games' development process, making it faster to develop games and possibly with better results.

1.2 Scope

There are a plethora of distinct game genres, such as platform, role-playing, shooter, and survival, among others. Due to time constraints in the thesis' development and writing, the possible genres covered by this work needed to be limited. Only genres suitable for single-player modes were considered to avoid dealing with multiple players' preferences.

In each genre, some distinct elements can be adapted. According to Liapis et al. [35], six creative facets can categorize these elements: visuals, audio, narrative, levels, rules, and gameplay. The authors also add that the generation problem can be applied to each facet and should be treated independently for each of them.

Upon reviewing related works, as described in chapter 2, the **platform genre** and the levels facet (**level design**) were ultimately chosen. This genre was chosen due to the amount of research on game adaptivity within it, making it easier to find levels' generators and adapt them to fulfill the thesis' objectives.

1.3 Research Questions

Two research questions were posed to guide the research related to the present work:

- Do players' profiles correlate with their preference for specific platform game elements?
- Does automatically generating platform levels and adapting them to the overall preferences of players impact their experience?

The first question is related to Player Profiling and corresponds to an intermediate step in the present work's research. The expected outcome of this question was a correlation between players' profiles and some platform game elements, helping with the following one. A static prototype of a 2D platform game, adapted from Infinite Mario Bros [45], was developed to answer it. Twelve levels were developed, each with the number of distinct elements increased or decreased to infer these correlations. Although the results did not yield correlations with enough degree of certainty, as analyzed in section 5.4, they allowed continuing with a slightly different approach by adapting the number of specific elements, according to the same results, to investigate if these levels can improve the players' overall experience.

The second question is related to the main objective of the work, i.e., the generation of game levels according to players' models. It is a problem mainly related to Procedural Content Generation and Player Modeling and Profiling. This question is derived from the previous one. To answer it, the prototype was extended, allowing to generate levels and then adapt them, as detailed in chapter 6. Since it was impossible to adapt levels to the players' profiles, another solution was devised by adjusting levels according to the first study's results to improve the players' overall experience.

1.4 Objectives

As seen in the previous section (section 1.3), two research questions were determined to guide the present thesis' work. These questions were chosen to reach the thesis goal: to evaluate the **impact** of automatically **generating and adapting levels** in a **game**.

To reach the stated objective, a methodology was developed, as described in chapter 3. It consists of the following phases:

- Select a game genre (2D platform game in this work's case), and choose or create a game within it;
- Adapt the selected game to collect players' preferences, gather experimental data, and correlate players' profiles, actions, and preferences with game content;
- Create an adaptive version based on the previous prototype and the data collected, and gather experimental data to evaluate the impact of the generation and adaptation of game levels.

At the end of the work, with the experimental data gathered from the adaptive game, the impact of automatically generating and adapting levels according to the players' preferences was measured and outlined in section 6.3. A more detailed discussion of the studies' results is available in chapter 7.

1.5 Contributions

The present work encloses several contributions. With the literature review, an analysis was performed on platform levels' elements and ways to generate or adapt them, as presented in table 2.6.

Moreover, a new methodology was created to correlate players' profiles, actions, and preferences with game elements and evaluate the impact of generating and adapting game levels.

Furthermore, there is a Super Mario level generator, adapted from a *JavaScript* version[1] of Infinite Mario Bros [45]. This new version[2] allows the creation of small *JavaScript Object Notation* (JSON) files describing the levels and then importing them to the game, enabling game designers to automatically generate levels and iterate over their JSON description to reach the desired configuration. The code was refactored to reduce the description files' sizes, storing only the relevant information of a level, such as its sections and enemies, making it more human-readable than the data arrays needed to render and play them.

Besides the previous feature, a gameplay metrics collector was also developed. Game designers can deploy their games and collect users' data to improve their experience using this mechanism or anything similar. The metrics collector can also register every game element by ID, allowing designers to know which elements are not being collected or interacted with and draw conclusions on why. Both in chapters 5 and 6, there are examples of gameplay data collected from experimentations and methods to analyze them.

Last but not least, the present work created two datasets, allowing other researchers to iterate over them and investigate other correlations with distinct gameplay data. The first dataset [16], described in chapter 5, consists of data gathered from twelve previously generated levels with distinct configurations. The second dataset [15], detailed in chapter 6, was produced by combining randomly generated levels and an adaptation of them. Both datasets contain the levels' descriptions in a JSON format and the players' actions, profiles, and preferences, allowing researchers to rerun the experiences in batches to collect new data about gameplay.

1.6 Document Structure

The current document starts with this introductory chapter (chapter 1), describing the context and motivation, the work's scope, the research questions, the objectives, and contributions.

Following the introduction is a literature review chapter (chapter 2), focusing on topics related to the present thesis, such as Game Adaptivity, Level Design, and Player Modeling and Profiling.

Chapter 3 explains the work's methodology, developed to answer both research questions. Then, chapter 4 shows the game selection and the solution's architecture. The following two chapters (chapters 5 and 6) present the two studies conducted during the work. Chapter 7 lists several testers' comments, discussing how they could influence the results and what could be done to avoid users' mistakes while answering the survey.

Finally, some conclusions are presented in the final chapter (chapter 8), summarizing the work done and showing work that can and should be done in the future.

[1] https://github.com/OpenHTML5Games/games-mirror/tree/gh-pages/dist/mariohtml5
[2] https://github.com/pemesteves/game-adaptivity

Chapter 2

Game Adaptivity through Level Design: a review

Game Adaptivity has been an increasing topic of research in previous years. Proofs of that can be seen in several literature reviews, with emphasis on the work from Lopes and Bidarra [37], which already concluded in 2011 that adaptivity was "establishing itself as a rapidly maturing field regarding its purposes" [37, p. 12], and the work from Dhelim et al. [12], which surveyed recommendation systems in distinct areas, pointing some recent works on game recommendation systems.

Since this work focuses on creating an adaptive game by generating levels according to players' profiles, the present research explores academic and commercial works related to Game Adaptivity, Level Design, and Player Modeling and Profiling. Each section corresponds to one of these topics and is divided into three main parts:

- an **introduction**, where the background and the concepts are introduced;
- the **section body**, where works related to the section are analyzed;
- a **summary**, where the results most aligned with this thesis' aim are condensed, and possible knowledge gaps in the area are mentioned.

The review starts with **Game Adaptivity**, focusing on works related to the one proposed by this thesis, and evaluating their impact on the game development industry. Since this work approaches level adaptivity, this section focuses more on PCG as it is one of the more straightforward options. Then the works related to **Player Modeling and Profiling** are analyzed by surveying related projects in this field, types of recommendation systems, and personality questionnaires, which would allow categorizing players. Lastly, **Level Design**'s works are traversed, providing examples of adaptive elements and taxonomy.

2.1 Game Adaptivity

According to Lopes and Bidarra [37], Game Adaptivity consists of creating content or adjusting parameters, making the games more challenging, unpredictable, and player-oriented. As stated by the authors, there are two main types of adaptivity: offline and online.

Offline adaptivity consists of creating content adapted to some predefined model before the gameplay starts [37]. An example could be creating game configuration files adapted to the players' needs. Procedural Content Generation (PCG) can also fit in this adaptivity category since the content is mainly generated before the game or level starts, as seen in the works presented in the following subsection. Besides its application in games, PCG is also seen in other areas, such as in generating 3D urban environments [9], creating vegetation [26], and generating mathematical problems [60], which shows the potential of using this technique for content creation.

In contrast, online adaptivity consists of adjusting content during the gameplay, which can be approached by creating new content while the game progresses, such as new quests or enemies, or by tweaking some parameters, such as the game's difficulty [37]. Dynamic Difficulty Adjustment (DDA) is an example of a problem addressing online adaptivity, both in the industry and academic works, by tracking the player's ability and adapting to it, as described by Zohaib [63].

Both offline and online adaptivity of game worlds and scenarios can be influenced by gameplay expectations, learning preferences, or assessment data, as mentioned by Lopes and Bidarra [37]. This relation is shown in figure 2.2, where the authors show the open challenges for game adaptivity.

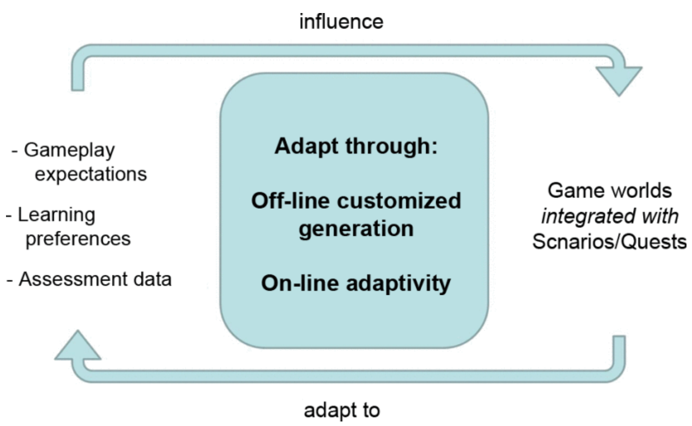

Figure 2.1: Open challenges for game adaptivity. Source: [37, Fig. 6]

This section reviews academic works on both types of adaptivity in level design and generation, focusing more on offline adaptivity through PCG. After reviewing these works, the impact on the Game Development industry is measured by describing games that address these problems. Finally, conclusions on how this thesis relates to these works are shown, pointing out possible knowledge gaps in the field.

2.1.1 Academic Works

As already established, research around Game Adaptivity has increased in the previous years. This literature review made it possible to check that this topic peaked in the previous decade, with several contributions by authors such as Georgio N. Yannakakis, Julian Togelius, Noor Shaker, and others.

Although there has been a peak in research on this topic, a recent study from Galdieri et al. [19] concluded, through a user study, that procedural level generation involves many risks and may even affect the users' satisfaction, leading to a decrease in the sales number and profit margin, which is what moves the industry. Their study was applied to a research puzzle game called *EscapeTower*, developed in the authors' previous studies. According to them, some users criticized the procedurally generated environments, categorizing them as too artificial and repetitive, which can be a source for their results.

Besides the work from Galdieri et al. [19], other works have been proposed over the years in the field of Game Adaptivity, most using PCG. An example is *Game-O-Matic*, from Treanor et al. [59], which aims at generating whole games according to ideas described by users. Other examples are related to Experience-Driven Procedural Content Generation (EDPCG).

The term EDPCG was introduced by Yannakakis and Togelius [62] in 2011. In their paper, they survey the four main components of EDPCG, i.e., player experience, content quality, content representation, and content generator, as shown in figure 2.2, providing a classification of approaches to each, and outlining their main research challenges. Besides showing three main approaches for modeling player experience (subjective, objective, and gameplay-based), which are more detailed in section 2.2, the authors show insights on how to evaluate the quality of game content items and optimize it for the experience.

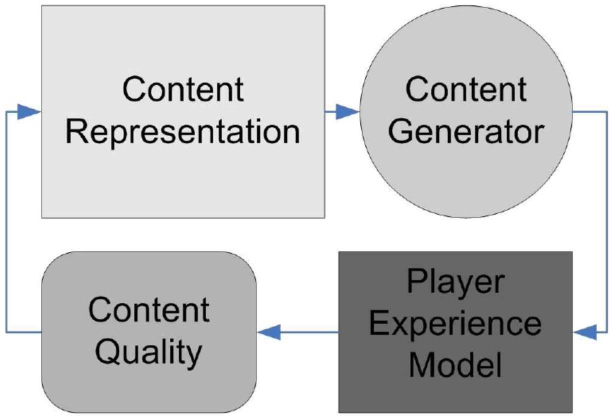

Figure 2.2: Main components of EDPCG [62, Fig. 1]

According to Yannakakis and Togelius [62], there is not a single solution to evaluate a game's content, leaving it up to the designer to decide how the content should be optimized - targeting, for instance, fun or frustration - and how to formalize it. Therefore, the authors distinguish three ways of assessing the generated content's quality:

- **Direct Evaluation**, where features are extracted from the content and mapped to a quality value. It can be of two types: **theory-driven**, in which the designer derives the mapping from intuition or a qualitative theory of emotion or experience, or **data-driven**, wherein data of the content's effect is collected via, for instance, questionnaires, using some automated tool to tune the mapping from content to experience and then to evaluation functions.

- **Simulation-based**, where an artificial agent plays a part of the game with the generated content. The agent, and consequently the quality value, can change during the game, making it a **dynamic** evaluation or not, which is called a **static** simulation-based evaluation.

- **Interactive**, where the content is scored according to the player interaction. It can be made **explicitly**, using questionnaires, or **implicitly** by measuring gameplay metrics.

Besides the previous work, other authors explored ways to analyze and evaluate PCG algorithms in Level Design. An example is a work from Biyik, and Sürer [6], applied to a dungeon-like escape-the-room game developed by the authors, called *The Haunted House*. The levels, generated using space partitioning algorithms, were evaluated using three main methods: (i) axial line analysis, (ii) connectivity or visibility graph, analysis, and (iii) quantitative analysis of the connection between spawning points and critical axes. The results show that connections between spawning points and critical axes may result in levels with better gameplay. The authors also conclude that these methods, called the Space Syntax approach, can be used by game designers to evaluate and prevent spatial configuration-related problems during the development process. However, these results were obtained from a different game genre with distinct spatial configurations, meaning that applying these methods to a platform can yield different results.

As mentioned above, Yannakakis and Togelius [62] show insights on how to optimize game content to improve the experience. To this end, they divide this problem into two main categories: (i) representation, and (ii) generation. Regarding content representation, the authors state that there are two main representations or a hybrid: (i) **symbolic**, when content is represented within data structures, such as graphs and (ii) **subsymbolic**, which allows greater variation and innovative creation, such as in artificial genotypes.

Encoding Type	**Proportionality between their size**	**How they map**
Direct	Linearly Proportional	Each part of genotype maps to a specific part of phenotype
Indirect	Does not need to be proportional	Nonlinearly

Table 2.1: Differences between the direct and indirect encodings. Source: [62]

Yannakakis and Togelius [62] also show an important distinction between direct and indirect encodings among content representation. These differences are represented in table 2.1, where they are compared based on the relation between genotypes, the structures represented by the content generator, and phenotypes, the structures or processes assessed by the evaluation function. Taking *Super Mario Bros.* [1] as an example, Yannakakis and Togelius use these definitions to show five distinct ways, from *directly* to most *indirectly*, to represent the game's levels, as shown in table 2.2. According to the authors, options 2 to 4 are suitable for searching for good platforming levels, making them helpful for this work.

Type	**Representation**
Directly	2D grid where each cell's content is specified separately
More Indirectly	List of positions and shapes of walls and ground, and another list of positions of enemies and items
Even More Indirectly	Repository of reusable patterns, and a list of how they are distributed across the level
Very Indirectly	List of desirable properties
Most Indirectly	Random number seed

Table 2.2: *Super Mario Bros.* [1] level representation. Source: [62]

Regarding the game's content generation, Yannakakis and Togelius [62] state that an ideal generator should "identify *if*, *how much*, and *how often* content should be generated for a particular player" [62, p. 11] since players can have distinct tastes regarding adaptation. The system should also recognize players who dislike adaptations. Besides, the authors show two techniques to generate content: (i) randomness, which creates unpredictability, and (ii) a search-based approach that allows designers to specify their desired properties.

Continuing with the EDPCG concept, a more recent work from 2021, performed by Shu, Liu, and Yannakakis [53], introduced a new framework called Experience-driven Procedural Content Generation via Reinforcement Learning (EDRL), which can be seen in figure 2.3. This framework was based on the previously explored EDPCG framework [62] and enables reinforcement learning (RL) agents to generate content adapted to players driven by experience-based functions. The authors used *Super Mario Bros.* [1] to test their approach. The levels were generated online and in segments, where the adjacent ones kept a moderate diversity level between them, following Koster's Theory of Fun [32], which the authors modeled to steer the adaptivity. However, the authors did not test the approach against human players.

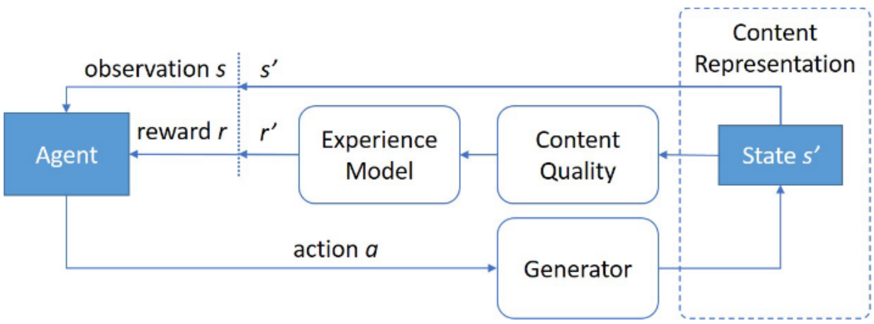

Figure 2.3: Overview of the EDRL framework. Source: [53, Fig. 1]

Shaker et al. [52] also applied Game Adaptivity to *Infinite Mario Bros* [45] by generating content adapted to some players' emotional states (engagement, frustration, and challenge), using player experience models constructed in an authors' previous study [50]. The authors used grammatical evolution, a grammar-based form of genetic programming, to represent the structure of the game's levels as design grammars. In their solution, the adaptation mechanics only considered the playing style exhibited by the player in the most recent level, assuming it was maintained during consecutive levels. Although the authors found "interesting relationships among the predictions of the emotional states" [52, p. 6], their approach was not tested against human players like the one mentioned above.

In a later study, Shaker, Yannakakis, and Togelius [51] investigated what makes a platform level engaging, challenging, or frustrating by extracting data (content, gameplay, and self-reported player experience) of a modified version of *Infinite Mario Bros* [45]. The modified level generator created levels according to six features:

- Number of gaps, i.e., spaces where the player may fall and lose the game;
- Average width of gaps;
- Number of enemies, which can be of two types;
- Enemy placement, which can be made in three distinct ways: around boxes, around gaps, or placed randomly;
- Number of powerups;
- Number of boxes, which may contain powerups or coins.

The levels were converted into sequences of numbers, representing each game item, to evaluate them. Besides these elements, gameplay data was mined through the recording of the following events and their timestamp:

- Level completion;
- Mario (player) death and its cause (enemy or gap related);
- Interaction with game items;
- Enemy kill and action that triggered it (jump on head, fire, or enemy throw);
- Mode changing (small, big, or fire);
- State changing (moving direction, jump, run, duck).

Finally, the authors relied on self-reported engagement, challenge, and frustration data by forcing the players to answer a questionnaire after small game sessions.

Differently from what was reviewed, Summerville et al. [58] defined, in 2018, the term Procedural Content Generation via Machine Learning (PCGML). According to the authors, PCGML differs from EDPCG since it models game content instead of experience, behavior, or preferences.

It also uses machine learning, but not for modeling content. The authors define five use cases for these techniques:

- **Autonomous Generation**, where content is completely generated without human input during this process;
- **Co-creative and Mixed-Initiative Design**, where an algorithm assists a human during the content creation process;
- **Repair**, which consists of algorithms that identify areas that are not playable;
- **Recognition, Critique, and Analysis**, in which the algorithms can evaluate the content;
- **Data Compression**, where the algorithms compress data to allow more efficient content storage.

In this thesis, PCGML could be used in the generation stage (autonomous generation) or to repair or evaluate the generated content.

From the previous concept, Sarkar and Cooper [48] introduced the term Game Design via Creative Machine Learning (GDCML), a subset of PCGML that enables creative machine learning applications for automated and mixed-initiative tools for game design. The models used by these techniques are trained on one or more games and are capable of several applications, such as game blending, the combination of levels and mechanics of two or more games into a new game, or level search, the generation of levels according to an input level and an objective.

Both the works from Summerville et al. [58] and Sarkar and Cooper [48] show a clear tendency to incorporate machine learning into game design. However, machine learning is out of this work's scope. Nevertheless, since most of these techniques avoid the human factor, they result in several open problems, such as ensuring games' playability, that must be explored in future work.

Besides the detailed works, others apply game adaptivity in distinct fields. For instance, Jacob et al. [27] apply player adaptivity in location-based games to reinforce the players' safety regarding their surroundings. Another example is the work from Nogueira et al. [42], where levels of a horror game are generated according to biofeedback. Although these works are off-topic with this thesis, they show the potential of game adaptivity to be used in various game genres.

2.1.2 Impact on the Game Development Industry

Besides academic works, game adaptivity has been used mainly with PCG in the game development industry.

It started in 1980 with *Rogue* (figure 2.4) [11], one of the first games using PCG. In this game, the developers created a dungeon generation, i.e., dungeons (or levels) are randomly generated every time the player descends a set of stairs. It inspired the roguelike genre, a subgenre of role-playing games with automatically generated levels, where games like *The Binding of Isaac* [14] can fit.

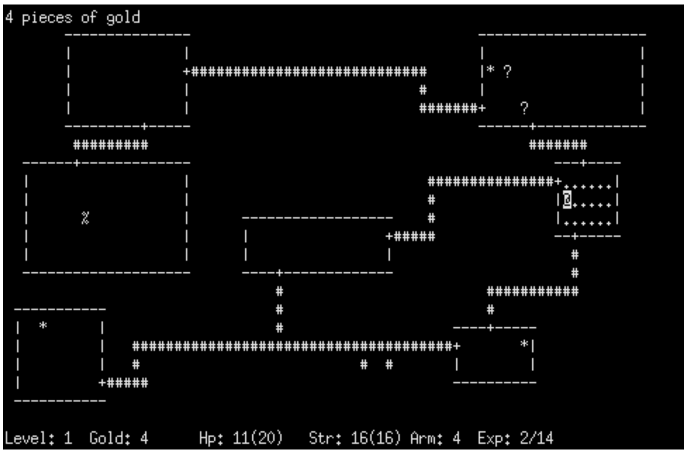

Figure 2.4: Rogue, 1980 [11]

Beyond the roguelike subgenre, other games use PCG in levels generation. One already seen in the previous subsection is *Infinite Mario Bros* [45], an open-source adapted version of *Super Mario Bros.* [1] developed by Markus Persson. Another example is *Spelunky* [39], a 2D action-adventure platform game where the levels are automatically generated.

Besides these examples, which have more than a decade, there are more recent games using PCG. The best example of these games is *No Man's Sky* [20]. In this game, terrains, enemies, ships, and even music, are procedurally generated, creating a world with infinite possibilities. This shows that PCG and, consequently, adaptivity have much potential in the game development industry and can be largely explored to create even better experiences for the players.

2.1.3 Summary

This section reviewed concepts regarding game adaptivity and how they can be used in academic and industry works. As shown here, it has great potential in the game development industry, especially using PCG.

There is also much research around this topic, which can help following the methodology proposed in chapter 3. The works reviewed [53, 52, 51] showed potential for using *Infinite Mario Bros* [45] as a case study. The work of Yannakakis and Togelius [62] showed how to assess the quality of generated content in three ways, which can contribute to the proposed generator. Shu, Liu, and Yannakakis [53] demonstrated how to define Koster's fun [32] as a function that can influence the content generator proposed. Shaker et al. [51] presented platform levels' features that can be adapted and gameplay events and questionnaires, which can be used to assess players' engagement, challenge, and frustration. Lastly, Summerville et al. [58] showed where and how to use machine learning with procedural content generation in games.

Although there was a spike around the early 2010s, with several papers targeting player experience using PCG algorithms, game adaptivity is still an area with much research. However, research trends have moved towards other subareas of game adaptivity and player modeling, such as player personalities, as described in the following subsection.

2.2 Player Modeling and Profiling

Works done in the Game Adaptivity area often define models of the player to generate or adapt their content. Several distinct works have been done in this field, such as modeling players' experience [52, 51, 53, 44] or their personalities [47]. In most researches, authors often distinguish between player modeling and profiling, referring to the first, Player Modeling, as the modeling of "complex *dynamic* phenomena during gameplay interaction" [61, p. 2] and the second, Player Profiling, as "the categorization of players based on *static* information that does not alter during gameplay," [61, p. 2] including "personality, culture background, gender and age" [61, p. 2].

The section starts by reviewing works related to player modeling and player profiling. Then, some recommendation systems and personality questionnaires are reviewed. Finally, it ends with a summary, concluding which works are most related to this thesis.

2.2.1 Player Modeling

As introduced above, Player Modeling consists of using artificial intelligence (AI) techniques to create computational models of players' behaviors, cognition, emotions, and other information beyond gameplay interaction (player profile), according to Yannakakis et al. [61].

Hooshyar, Yousefi, and Lim [25] show two approaches to build these computational models:

- **Theory-driven**, or model-based [61], where the model is based on social sciences, such as experimental psychology. Examples of these models are the emotional models, based on emotion theories.

- **Data-driven**, or model-free [61], which, in contrast, are based on natural sciences and computer science. These models are mainly built through game data mining by identifying players' behavioral patterns to predict their actions later.

Moreover, Yannakakis et al. [61] discuss the three main types of inputs a computational player model can take:

- **Behavioral data**, i.e., player actions collected during gameplay;

- **Objective data**, i.e., players' body responses to game stimuli;

- **Game context**, including player interactions and the type of content seen, played, or created.

Yannakakis and Togelius [62] have previously introduced these approaches, targeting player experience modeling (PEM). In this paper, they identified three approaches for PEM:

- **Objective PEM**, which consists of gathering data related to player body responses;

- **Gameplay-based PEM**, where data is obtained from players' interactions with the game;
- **Subjective PEM**, which consists of players' self-reported data and differs from the taxonomy seen above.

Besides player experience, there are other ways to model a player. Bakkes et al. [4] discussed the modeling of players' behaviors. In their work, the authors detail four approaches to model players' behavior:

- Modeling **player actions**, which generally tries to predict players' actions given the current game state, as seen, for instance, in AI for board games;
- Modeling **player tactics**, i.e., the short-term or local game behavior, composed of a series of actions;
- Modeling **player strategies**, which concerns the long-term or global game behavior composed of a series of player tactics that may include the behavior in a game, different playthroughs of the same game, or even across distinct games;
- **Player profiling**, which, as seen in figure 2.5, can guide the players' actions, tactics, and strategies and is looked at more in-depth in the following subsection.

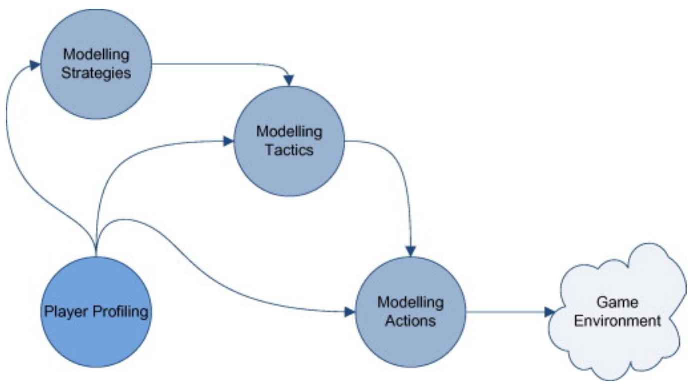

Figure 2.5: Behavioral modeling taxonomy. Source: [4, Fig. 1]

Moving from theory to practice, some player modeling works should be mentioned. The first one is Pedersen et al. [44], where the experience was modeled as fun, challenging, boring, frustrating, predictable, or anxious. To do that, the authors collected data from controllable game features (parameters used for level generation), how the users play the game, and their experience playing it. The approach was applied to *Infinite Mario Bros* [45], as other works seen in section 2.1, showing how each of the states mentioned correlates to the players' behavior in the game. Boredom, for instance, was associated with the average width of gaps in a way that wider gaps lead to a less boring experience. The correlation made by Pedersen et al. [44] can be explored in this thesis, leading to choosing certain elements to provoke certain types of experiences.

Another example, this in the field of narratives, is the work of Freilão [18], where the narrative was adapted to the players' emotional conditions. Although narratives are completely off-topic with the present work, their approach and, consequently, evaluation can be adapted to the platform genre. As seen in figure 2.6, their framework, i.e., the adaptivity system, takes two inputs: the player's emotional profile, drawn up from a questionnaire, and the default narrative. Then the framework adapted the default narrative according to the given profile. For instance, the environment colors would be faded if the player's sadness was greater than 10, on a scale ranging between 4 and 28. The players then played the default and adapted versions in a pseudo-random order, and after each version, they filled out a questionnaire about their experience in the current version. Lastly, after finishing the two versions, the players answered some general questions about the experience as a whole.

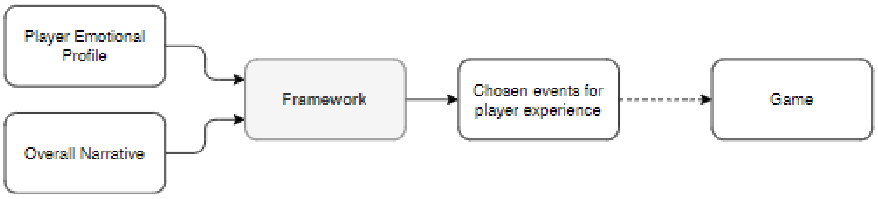

Figure 2.6: Solution developed by [18, Fig. 3.1]

Besides the presented works, there are other ways to create players' models and personalize game content. For instance, Camilleri, Yannakakis, and Dingli [8] crowdsourced rank-based annotations from users watching videos of characters controlled by AI and humans playing a 2D platform game to model believability, i.e., the ability to convince observers that a human player is controlling a character. Blom et al. [7] personalized levels of *Infinite Mario Bros* [45] according to players' facial expressions, i.e., during the gameplay, the experience adapted to the players' emotions inferred from the facial expression recognition. Fernandéz, Koji, and Kondo [17] adapted a 2D platform game difficulty by generating levels according to players' performance and electroencephalography data obtained from a biosensor while playing, based on DDA and Rhythm-Group Theory concepts. Spiel, Bertle, and Kayali [57] adapted *Tetris'* difficulty to players' performance and eye movements. Last but not least, Jacob et al. [28] applied player adaptivity to an exergame using biometric data and fitness models, ensuring the players were motivated to be physically pushed while avoiding harm.

2.2.2 Player Profiling

As seen before, Player Profiling categorizes players according to information that does not change during gameplay, such as personality, cultural background, or demographic data [61]. Bakkes et al. [4] add that player profiles "provide motives or explanations for observed behavior, regardless whether it concerns strategic behaviour, tactical behaviour, or actions" [4, p. 7].

One of the first examples of player profiling in the literature is Bartle's taxonomy [5]. In this taxonomy, demonstrated in figure 2.7, Bartle defines four types of players:

- **Killers**, which are players that want to demonstrate their superiority above other players by acting on them;
- **Achievers**, which act on the world by playing games to master them;
- **Socialisers**, whose main objective is to interact with other players;
- **Explorers**, which are players interested in interacting with the game world.

Bartle also demonstrates how the profiles interact with each other. For instance, the author states that killers do not like to play against or with other killers, except in pre-organized matches, such as gaming competitions. Although these interactions could help understand how distinct players react in a multiplayer game, it is off-topic with the present thesis since it looks into single-player platform games. Nevertheless, Bartle's taxonomy could be used to create players' profiles in an adaptive variation, excluding the socializer trait, which would only make sense in multiplayer games.

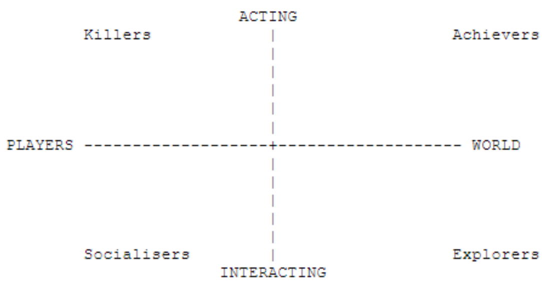

Figure 2.7: Bartle's taxonomy. Source: [5]

Another taxonomy to define players' profiles is the one presented in 2011 by Nacke, Bateman, and Mandryk [40]. In this work, the authors presented the *BrainHex* model, which defines seven types of players:

- **Seekers** are curious about the game world and enjoy moments of wonder;
- **Survivors** enjoy the intensity of the experience related to terror;
- **Daredevils** seek thrill, excitement, and taking risks;
- **Masterminds** enjoy solving puzzles and conceiving strategies;
- **Conquerors** enjoy struggling against opponents and beating other players;
- **Socializers** enjoy talking, helping, and hanging around with people they trust and get angry if others abuse their trust;

- **Achievers** love to reach goals and are motivated by long-term achievements.

Using the *BrainHex* model suffers from the same problem as the one defined above, i.e., using a socializer trait is impossible in a single-player game since there are no other users with whom the player can interact. Nevertheless, the taxonomy could be used, excluding this trait.

As seen in the previous subsection, Bakkes et al. [4] state that player profiling can guide the players' actions, tactics, and strategies by incorporating psychological features in player models. Furthermore, Lankveld et al. [33] investigated if games can be used to determine players' personalities, yielding positive results. However, the use of personality to personalize video games had not been used at least until 2014, as stated by Karpinskyj, Zambetta, and Cavedon [30].

Besides showing a lack of personality applications in video game personalization, the work from Karpinskyj, Zambetta, and Cavedon [30] also summarizes other examples of players' input data and their applications. Table 2.3 show examples retrieved from their work (note that the references were taken not to mislead the reader), where it can be seen that the examples in the personalization area are more focused on players' preferences, experience, and performance. That means that personality and in-game behavior have space to be explored in adaptivity works, especially in the platform genre.

Input Data	Gameplay Personalization
Preferences	Role-playing game maps Difficulty, weapon control, and objectives Platforming levels
Personality	No applications
Experience	Camera position Plot/story points Platforming levels Action game levels
Performance	Platforming levels Enemy type and count Dungeon structure Battle missions
In-game behavior	Quest structure Weapon behavior

Table 2.3: Some examples of video game personalization. Source: [30, Table 1]

More recent works have proposed frameworks for personality-based adaptivity following the previous work. In 2016, Nagle, Wolf, and Riener [41] related the Big Five personality traits (Openness, Agreeableness, Conscientiousness, Extraversion, and Neuroticism), introduced by John and Srivastava [29], with four distinct implementations of difficulty adjustment applied to a first-person shooter. The authors linked, with their research, the Big Five personality traits with some motivations to play games, which can be seen in table 2.4 (note that the references were taken to not mislead the reader, as in the previous table). However, no link was found with the Neuroticism trait, the fifth trait. Since their solution was applied to a first-person shooter, applying the same methodology to distinct genres can yield different results, as mentioned by the authors. Moreover, the motivations for playing outlined by the authors can help build a mechanism to correlate personality traits with a platform game's content, the target of this thesis.

Personality Trait	**Motivations to Play Games**
Openness	Actively seek flow experience Preferring balance of difficulty and skill
Agreeableness	Feelings of competence
Conscientiousness	Progress in games Time spent playing video games Less prone to boredom
Extraversion	Wanting to play well on difficult games Preference for difficult tasks

Table 2.4: Links between the Big Five personality traits and motivations to play games. Source: [41]

An even more recent work, by Lima, Freijó, and Furtado [10], associated the Big Five personality traits with general behavioral aspects and these with general in-game behaviors, which can be seen in table 2.5. Their approach was applied to narratives, influencing the work of Freilão [18] already mentioned in the previous subsection. The authors measure the players' personalities through questionnaires described in the last subsection of this chapter and generate a narrative using functions that connect each trait with in-game behavior associated with it. Although the narratives field is off-topic with this thesis, this methodology, and mainly the mapping between the Big Five factors and the in-game players' behaviors, can be of great use to the present thesis.

There are other ways to create players' profiles. For instance, Lima [13] created ten types of players, representing the player base of the game, based on the author's experience of playing the game and perception of difficulty. These players' types can be called personas, as seen, for instance, in the work of LeRouge et al. [34], since they are fictional character that represents a user group, including some likes, dislikes, background and expectations, and most importantly, the main goals for the user. For example, the first player type defined by Lima [13] is a player who wants to collect coins (the goal), and it represents players with experience in the target game who likes to go fast and collect everything on the level. The author's approach was applied to a 2D platform game, where agents would test the levels to maximize the persona's gaming experience. However, the author's solution was not tested with humans, making it impossible to know if this type of player profiling would succeed in an industry game.

The approach mentioned above, using personas to model the players, can be helpful for the present thesis, specifically if the results from the first study, described in chapter 3, do not yield the expected outcomes, i.e., if a correlation between players' personalities and their preferences for specific game elements could not be found.

Big Five Factors	**Behavioral aspects**	**In-game player behavior**
Openness	+ curious, interested, inquisitive	. Explores the environment . Collects all the available items
	- indifferent, incurious, uninterested	. Explores only indispensable parts of the environment . Collects only indispensable items
Conscientiousness	+ meticulous, efficient, systematic	. Rarely gets attacked by enemies . Rarely misses a shot . Collects and uses items only when they are needed
	- careless, chaotic, disorderly	. Frequently gets attacked by enemies . Frequently misses shots . Collects and uses items when they are not needed
Extraversion	+ sociable, talkative, active	. Frequently interacts with non-player characters . Interacts with non-player characters as soon as possible
	- reserved, shy, passive	. Rarely interacts with non-player characters . Postpones interactions with non-player characters
Agreeableness	+ friendly, altruistic, helpful	. Always tries to save non-player characters that are in danger
	- hostile, selfish, obstinate	. Rarely tries to save non-player characters that are in danger
Neuroticism	+ aggressive, nervous, unstable	. Tries to kill all enemies . Performs disordered movements
	- calm, relaxed, balanced	. Kills only threatening enemies . Performs only necessary movements

Table 2.5: General behaviors of the Big Five factors. Source: [10, Table 2]

2.2.3 Recommendation Systems

As the name says, recommendation systems, also called recommender systems, provide personalized recommendations to the users, which can be content or services [2]. According to Dhelim et al. [12], these systems usually have three phases: (i) **rating**, where the user states their preferences, (ii) **filtering**, where the results are filtered according to different strategies, and (iii) **recommendation**, where the filtered results are presented to the user.

Since this thesis' objectives are not to propose a recommendation system but a system that adapt itself without any user input, recommendation systems may seem off-topic. Nevertheless, the solution proposed could rely on a recommendation system since the adaptations could be proposed to the player instead of acting without the player's knowledge. Another option would be to start the adaptation from a potentially optimal solution, according to some ranking and matching techniques. Therefore, some recommendation systems were reviewed, especially those personality-aware.

Personality-aware recommendation systems are a subset of recommendation systems where the user's personality is measured, and the recommendations are given according to it. As stated by Dhelim et al. [12], these systems add two more phases to the general recommendation system: (i) **personality measurement**, before the rating stage, where the user's personality is assessed using questionnaires, and (ii) **personality matching**, which changes the filtering stage by matching items according to similar personality traits.

Dhelim et al. [12] recently surveyed several personality-aware recommendation systems across distinct areas, including games. The authors highlight six works in this field that connect personality traits with users' gaming preferences and behaviors. According to their research, the Five-Factor Model (or Big Five) [29] is the most used personality model in this area. However, there are some using HEXACO [3]. In other fields, such as movie or music recommendations, the Big Five [29] is also the most used model, so questionnaires related to it are reviewed in the following subsection.

2.2.4 Personality Questionnaires

As already mentioned, Karpinskyj, Zambetta, and Cavedon [30] concluded in 2014 that no gameplay personalization applications were using personality as input data to the adaptivity system. More recent studies, such as the one conducted by Dhelim et al. [12], proved that personality could be used in recommending of games. From the literature review conducted and explained in this chapter, it can be concluded that personality can be used for game adaptivity, and there is no application using personality as input to adapt platform games. The most similar approach to the one proposed in chapter 3 is the work of Lima, Feijó, and Furtado [10], which connects personality traits with players' in-game behavior. However, the authors' approach was applied to narratives and did not specify, for instance, how many enemies a calm person would kill on a level, among other relations.

Considering that, some personality questionnaires were reviewed during this literature review. The focus turned to questionnaires assessing the Big Five traits since, according to the research made by Dhelim et al. [12], it is the most used personality model. From the set of questionnaires, only the small were considered (maximum of twenty questions) since they need to be presented before the game starts, leading players to abandon the game if facing an extensive questionnaire. According to Dhelim et al. [12], the most prominent short questionnaires for the Big Five model are the BFI-10 [46] and the TIPI [22].

Both BFI-10 [46] and TIPI [22] consist of ten questions. The first is evaluated from 1 (disagree strongly) to 5 (agree strongly). In contrast, the second is evaluated in a range of 1 (disagree strongly) to 7 (agree strongly), adding a *disagree moderately* and an *agree moderately* options to the questionnaire evaluation, which can confuse the person being assessed. Both questionnaires can produce misleading results to the system, as mentioned by Dhelim et al. [12], since people can think of themselves differently from reality, depending on their surrounding environment. Nevertheless, although the authors of both questionnaires agree that short personality measures cannot substitute the regular assessments due to their inaccuracy, they can be used in some applications where the participant time is limited, such as in recommendation systems [22, 46, 12].

Lastly, albeit both questionnaires have similar results in terms of accuracy, the BFI-10 scores a little better than TIPI in most parameters, but it needs one more item to properly score the Agreeableness trait, as shown by Rammstedt and John [46].

2.2.5 Summary

This section reviewed player modeling and profiling concepts and their applications in game adaptivity and recommendation systems.

Player modeling and profiling sometimes are used as synonyms. However, in most works, player profiling is defined as a subset of modeling, in which players are categorized according to information that does not change during gameplay, such as personality, cultural background, or demographic data, as stated by Yannakakis et al. [61].

Hooshyar, Yousefi, and Lim [25] define two approaches to modeling players, theory-driven and data-driven. Yannakakis et al. [61] also establish three main types of inputs to these models, behavioral data, objective data, and game context. This thesis follows a **data-driven** approach since a map between the input and a player state representation was made. Moreover, the model takes behavioral data, and the game context as input since the previous mapping was made according to these data.

Considering the approaches for player experience modeling defined by Yannakakis and Togelius [62], it can be stated that the model follows an objective and subjective approach, although it does not model players' experience.

According to the work of Karpinskyj, Zambetta, and Cavedon [30], there was no game personalization application using personality until 2014. As seen from the literature review, meanwhile, there were some works exploring this application, such as the work from Lima, Freijó, and Furtado [10]. Most of these recent works model the Big Five factors [29], with the BFI-10 [46] and the

TIPI [22] questionnaires. However, there is no application in the platform's level design, which can be explored in the present thesis.

Moreover, the methodology proposed by this thesis, described in chapter 3, is similar to the one proposed by Freilão [18], by adapting content, in this case, levels, to the players given the player's personality and a default range of levels' parameters as input. Then the players play a default and an adapted version, in a pseudo-random order, answering questionnaires about their experience after each playing each one. Finally, the players answer a questionnaire about their general experience.

In conclusion, besides adapting levels, the proposed solution could also be used as a recommendation system by adjusting and suggesting content with or without the player's knowledge.

2.3 Video Games Level Design

Over the years, there have been attempts to categorize game content. In 2013, Henrikx et al. [24] proposed a taxonomy with six layers of game content: bits, space, systems, scenarios, design, and derived (including content from the other layers). Liapis, Yannakakis, and Togelius [36] created, in 2014, a classification of creative game facets, including visuals, audio, narrative, ludus, level architecture, and gameplay.

More recently, in 2019, Liapis et al. [35] defined a new taxonomy based on the previous work. According to the authors, games can be decomposed into six creative facets (figure 2.8): visuals, audio, narrative, levels, rules, and gameplay. The authors add that the problem of game content generation can be applied to each of these facets, and it should be treated independently.

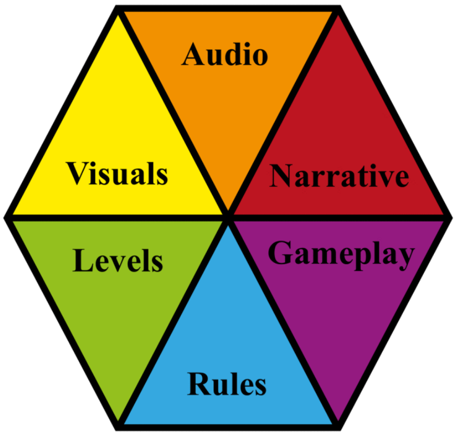

Figure 2.8: Game creative facets. Source: [35, Fig. 2]

Given these facts, the levels facet is considered in this thesis. As stated by Liapis et al. [35], levels are virtual spaces where the gameplay occurs, ranging from extremely simple to highly complex areas. The authors also add that levels need to combine form, allowing players to navigate "via memorable visible landmarks" [35, p. 3], and function, constraining the players' paths.

However, there could be exceptions to this combination, which is expected, for instance, in several horror games, where usually lights are turned off to provide better *jump scares*.

Searching for Level Design works, there are mainly two types of works that can be found: some are more focused on design, investigating, for example, levels' properties that would improve the player experience; others are more focused on the generation, proposing tools for generating them.

An example of a work related to the design of levels is the one proposed by Geslin, Jégou, and Beaudoin [21]. The authors investigate how color properties elicit emotions in video games. Their results show a significant correlation between luminance, saturation, and lightness and the emotions of joy, sadness, fear, and serenity. Although the work is off-topic with the present thesis, the results show that distinct colors can elicit opposite emotions, meaning players can perceive a different experience just with a change in the level's color. Due to the previous conclusion, the studies in this thesis should be conducted on levels with similar colors to avoid players experiencing different feelings, even if the levels have the same content.

Regarding level generation, Smith, Whitehead, and Mateas [54] proposed a tool to assist in platform level design using Procedural Content Generation. The tool, called *Tanagra*, creates levels autonomously, places and moves objects according to the designer's input, modifies the beat timeline (a mechanism to edit a level's pace) according to the designer's input, and also ensures the levels are playable. The approach followed by the authors, using the beat timeline, could be helpful in the generation stage of the methodology proposed in chapter 3. Nevertheless, *Tanagra* does not follow a player model in the generation stage, so it is only helpful for assisting designers, which is not intended for this thesis.

Similar to the previous tool, there is *Launchpad*, created by Smith et al. [55]. The authors propose an autonomous level generator based on a formal model of a 2D platform level design. In this work, the levels are generated with a grammar-based approach, using rhythm groups, i.e., the levels are generated using notions of rhythm and pacing of player actions, which, according to the authors, are important factors but not the only ones, to the players' enjoyment of levels. Although it is outside this work's scope, the authors mention a taxonomy for level components observed in industry platform levels, grouping them into five categories: (i) platforms, (ii) obstacles (e.g., gaps, enemies, stompers), (iii) movement aids (e.g., springs), (iv) collectible items (e.g., coins), and (v) triggers, which add puzzle elements into platform games.

A distinct approach to the one previously seen is the work of Sorenson et al. [56]. In this work, the authors present player enjoyment model dependent on rhythm groups and evaluate the model's ability to identify commercial-quality levels. The authors' approach was applied to *Infinite Mario Bros* [45] and *The Legend of Zelda* [38]. For both games, the authors identified their design elements, i.e., the elements and constraints for these elements necessary to build a level of these games, which can be helpful since *Infinite Mario Bros* is a platform game that can be used to construct the methodology proposed in chapter 3.

As seen above, there are distinct ways to approach level design in the literature, and other works should be mentioned. For example, Khalifa et al. [31] address levels' creation revolving around specific mechanics in the game by applying constrained evolutionary, and quality-diversity algorithms to *Super Mario Bros.* levels [1], using three approaches (Limited Agents, Punishing Model, and Mechanics Dimensions). Green et al. [23] extend the previous work by stitching together pre-generated levels containing specific mechanics, creating levels with a similar mechanical sequence to the target but with a different playthrough experience.

There are also different genres for which generation, or adaptivity, can be applied. Each of these genres has distinct elements, as seen, for instance, in work from Sorenson et al. [56]. The following subsection enumerates some elements of platform games and ways to generate or adapt them.

2.3.1 Adaptive Platform Levels' Elements

As already mentioned, there are distinct genres where generation and adaptivity can be used. As mentioned by Yannakakis and Togelius [62], game content affects gameplay, not related to NPC behaviors or the game engine, such as levels, dialogue, characters, rulesets, music, weapons, and others. This subsection focuses only on platform levels' content since they are also the focus of this thesis. Table 2.6 shows platform levels' elements seen in the literature and the parameters used to generate or adapt them.

According to this literature review, distinct elements can be generated or adapted in platform games. Most adaptations change objects' placement or numbers, as seen in table 2.6. In the case of gaps or holes, their width and height can also be found in related works [44, 54, 51] by changing their average, minimum, or maximum values. These properties, width, and height could also be applied to platforms since they are usually variable in the industry platform games. Other elements, such as stompers and springs, always have predefined sizes in platform games, such as in *Super Mario Bros.* [1], but varying them could improve the players' engagement. Thus, variable width and height could be applied to the stompers and springs, at least in an initial prototype, to study the players' reactions to these elements.

Other elements or ways to adapt them are not seen in the literature and could be explored in an initial stage of development. In this thesis, elements beyond those presented in table 2.6 can be explored. For instance, the enemies' movement speed could be varied to hamper or ease the players' progression in a level, or even the player's character movement speed. The game would probably lose some of its essence with these adaptations, so they should be used carefully.

Element	Ways to generate and adapt them
Blocks / Boxes	Number [51] Placement [56]
Coins	Number [55] Placement [55]
Enemies	Number [55, 51, 8, 17] Placement [55, 54, 8, 17]
Gaps / Holes, where the character may fall and die	Height [54] Number [44, 55, 51, 8, 17] Placement [44, 54, 55, 17] Width [44, 54, 51, 8]
Platforms	Movement Direction [55] Number [55] Placement [54, 55] Slope [55]
Powerups	Number [51]
Spawn Points	Number [51]
Spikes	Number [55, 17] Placement [55, 17]
Springs	Number [55] Placement [54, 55]
Stompers	Number [55] Placement [54, 55]

Table 2.6: Platform levels' elements, and ways to generate or adapt them.

Chapter 3

Exploring Player Adaptivity in Level Design: Methodology

To reach the objectives proposed in section 1.4 and answer the questions determined in section 1.3, a generic methodology was elaborated. In this thesis, it was implemented with a specific game genre and specific game elements, as mentioned in section 1.2, but it can and should, be applied to others.

This methodology is generic and can be applied to distinct game genres and players' profiling methods. As seen in figure 3.1, it can be divided mainly into three phases:

- **Game Selection**, where the game genre and game within it are selected;
- **First Study**, where the correlations between profiles and game elements are analyzed;
- **Second Study**, where an adaptation is created, and its impact is measured.

The following three sections describe each of the previous stages, allowing the reader to replicate it in their works. Then, the foundations for this methodology are presented, highlighting the works that helped create it.

3.1 Game Selection

The **Game Selection** corresponds to the first stage of the methodology. As the name suggests, the output of this stage is a game chosen for the subsequent phases.

The first step of this stage is the selection of a game genre. As mentioned in section 1.2, there are a plethora of game genres from which the researcher can choose. Some of them can share some properties, leading to results that can be transposed between these genres.

After selecting the genre, some research for games within the genre allowing for content generation must be done. If no game matches these requirements, researchers should create one of

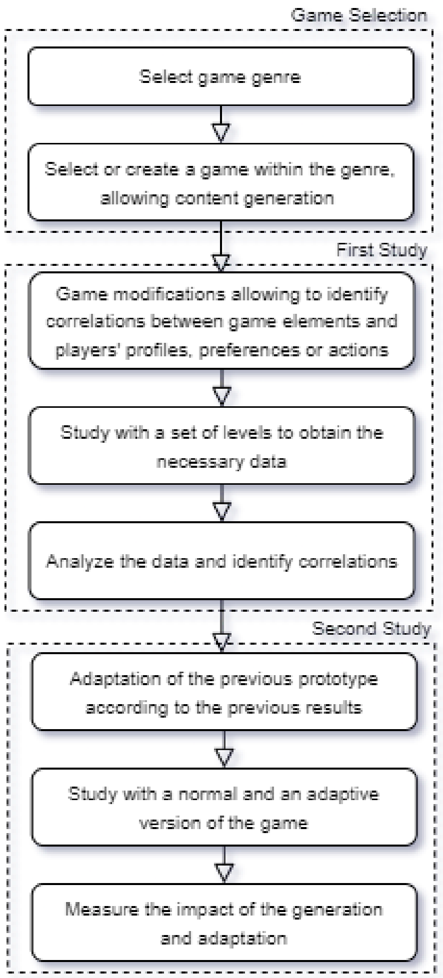

Figure 3.1: Flowchart with the proposed methodology split into phases.

their own, with the awareness that the content generator can influence people's experiences, both positively and adversely, not yielding the expected results. With this selection, researchers are ready to follow to the next stage.

3.2 First Study

Following the game selection, the **First Study** can be prepared. The study aims to correlate game content with players' profiles, preferences, or actions.

First, the game may need to be modified to reach its goal, i.e., to track game elements, gameplay metrics, and player actions. Moreover, the player profiles and preferences need to be measured, which can be done via questionnaires. These modifications result in a prototype ready to be used in the study.

After the prototype's development, a study with a set of pre-generated levels should be conducted, thus enabling researchers to control the data they want to obtain. The study should produce a new dataset with the needed information from the game and players to infer correlations.

With this dataset, researchers are ready to analyze the data, identify possible correlations, and answer the first research question pointed out in section 1.3.

Upon this analysis, the data collected must be ready to use for an adaptation of the study's prototype. Supposing these results are not enough for an adaptation, the study should be re-conducted, collecting more data about gameplay and players' profiles.

3.3 Second Study

The third and last stage of this methodology is the **Second Study**. This study aims to evaluate the impact of generating and adapting levels to players' overall experience.

As in the previous phase, the game may need to be modified to allow adaptations of levels. These modifications should be made according to the previous study's results.

Then, a study should be conducted, showing the players a standard and adaptive version of the game. This can be done with an A/B methodology, presenting two levels to the user in a pseudo-random order. The study should produce a new dataset with the needed information from the game and players to measure the impact of generating and adapting levels.

With this dataset, researchers are ready to analyze the data and answer the second research question posed in section 1.3. If the results prove that the adaptation impacts the players' experience, it can be used in industry games, helping game designers in the game development process.

3.4 Foundations

As reviewed in chapter 2, some works contributed to the methodology presented here. From these works, two can be highlighted, as they represent a significant contribution to each of the studies.

The first is the work of Shaker, Yannakakis, and Togelius [51]. The authors extract data from game elements, gameplay, and self-reported player experience and generate levels according to several features. In this methodology, these properties should be extracted in both studies, adding the measurement of players' profiles, which the authors do not target. This way, correlations between game elements and players' profiles, behaviors, and preferences can be found. Moreover, the impact of generating and adapting levels can be measured, and this data can help understand how each element influence the players' experience.

Another basis for this methodology is the work of Freilão [18]. This work presented standard and adapted narratives to players in a pseudo-random order, using an A/B methodology. After playing them, the users needed to answer a questionnaire about their experience. Although the work proposes a framework for narratives, where adaptations to specific emotional states needed to be previously described by the game designer, it helped create the second study stage. This way, players may be presented with standard and adapted levels in a pseudo-random order, answering questionnaires about their experience at the end.

Chapter 4

Selecting a Platform Game Allowing for Content Generation

As seen in chapter 3, the methodology created for this work may be used in different games and genres and can be divided into three stages.

This chapter describes the beginning of the platform case study created from it to infer correlations between players' personalities and game elements and measure the impact of levels' adaptations.

So, the first stage of the methodology, **Game Selection**, is presented in the following section by highlighting the games found to meet this work's requirements and the chosen game. Since both the first and second studies depend on the game selection, their common properties, i.e., the architecture of the selected game and the modifications to this architecture shared by both prototypes, are presented in the following sections.

After thoroughly reading this section, the reader can proceed to chapters 5 and 6 and better understand the studies conducted.

4.1 Game Selection

After selecting the game genre, for the reasons mentioned in section 1.2, research for platform games allowing for content generation was conducted.

As described throughout the state-of-the-art review (chapter 2), several games were found to meet part of the criteria, i.e., some generated content but were not platformers, and others generated content different from what was expected. For instance, *Rogue* [11], one of the first games using Procedural Content Generation (PCG), and *Diablo* [43] allowed to generate dungeons, or levels, for other game genres. Another example is *No Man's Sky* [20], which uses PCG to generate everything available in the game, i.e., galaxies, planets, and animals, but it represents the survival genre.

A smaller set of games were found to meet all the requirements[1], where two stand out: *Infinite Mario Bros* [45], an open-source version of Nintendo's *Super Mario Bros.* [1], developed by Markus Persson in 2006 (figure 4.1), and *Spelunky* [39]. Both games are well-known platformers allowing level generation and have been used in previous investigations in the game adaptivity field. However, *Infinite Mario Bros* was chosen since it already contained a port for *JavaScript*[2], allowing it to run on browsers, facilitating its distribution by players.

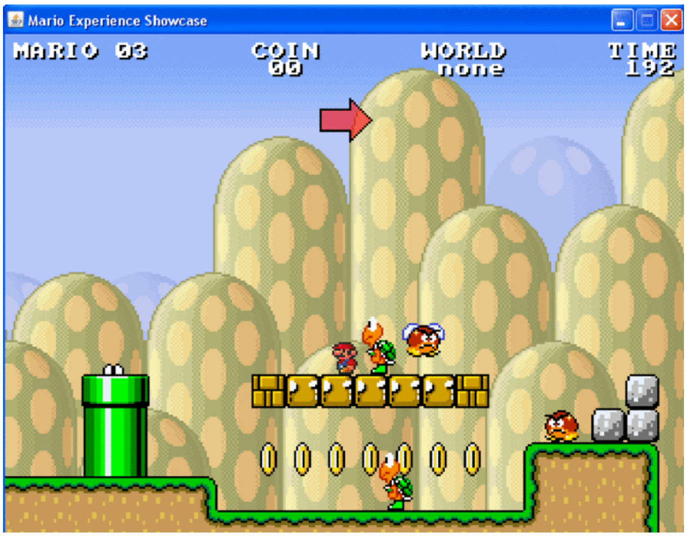

Figure 4.1: Infinite Mario Bros [45] level. Source: [51, Fig. 1].

4.2 Infinite Mario Bros Architecture

As mentioned in the previous section, a *JavaScript* (JS) port[2] of *Infinite Mario Bros* (IMB) [45] was selected for the experimentations.

Like most games, IMB uses an engine that abstracts most of its code. The game engine was created by the authors of the port and is available with the game. The engine was not modified for these studies since it provided everything they needed. The use of the game engine is not restricted to IMB since it is generic enough to develop other similar web games.

The game's code is separated from the engine, which can be distinguished by the class names since the engine's classes belong to the *Engine* prototype, which acts as an abstract class. Before the first study phase began, the game's code was refactored by transforming the prototypes into JS classes, simplifying the reading, editing, and extension into new types to create new behaviors.

Regarding the game itself, it is divided into the six states seen in figure 4.2:

[1] http://pcg.wikidot.com/genre:platform
[2] https://github.com/OpenHTML5Games/games-mirror/tree/gh-pages/dist/mariohtml5

- The **Loading** state starts the game, and, as the name says, it is where everything, from sprites to audio clips, is loaded into memory. The game automatically advances to the next state after the loading process is complete;
- The **Title** state shows the name and an image of the game. To proceed to the following state, the player needs to press S;
- The **Map** state represents the world's map, where players can navigate and access distinct levels. From this state, the players can enter levels by pressing S above a level's tile, or the game can finish if the player has already reached the eighth world, which requires completing most levels of the previous worlds;
- The **Win** state, as said above, can be reached by reaching the game's eighth world, showing a victory screen when it happens. Players can return to the title screen by pressing the S key.
- The **Level** state is where players can play the game by controlling a character, Mario, collecting items, and killing enemies. After beating the level or losing a life, players are redirected to the Map state, where they can choose to play the same or a different level. If the players lose all their three lives, they are forwarded to the Lose state.
- The **Lose** state is reached when the players lose their lives, showing a game over screen. By pressing S, the players can return to the title screen.

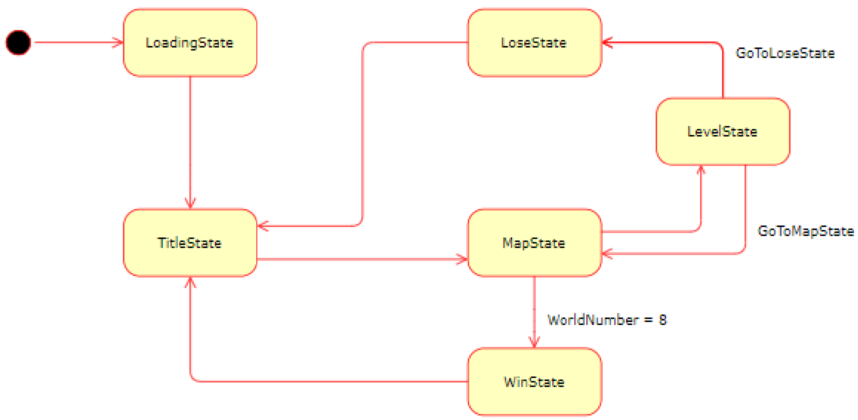

Figure 4.2: Infinite Mario Bros [45] game states.

During the Level state, several elements are rendered on the screen and can be used by players to reach the ending or score more points. These elements, seen in figure 4.3, extend the *Engine.Drawable* class, which allows rendering and updating their state. There are three major classes of drawable objects in the game: the (i) **LevelRenderer**, the (ii) **BackgroundRenderer**, and the (iii) **NotchSprite**. The classes extending the NotchSprite represent every sprite in the game and can be used in the adaptations. For example, in the first study conducted during this thesis work, described in chapter 5, the number of enemies, coins, and powerups (fire flowers and mushrooms) were adapted to create levels with different elements' configurations.

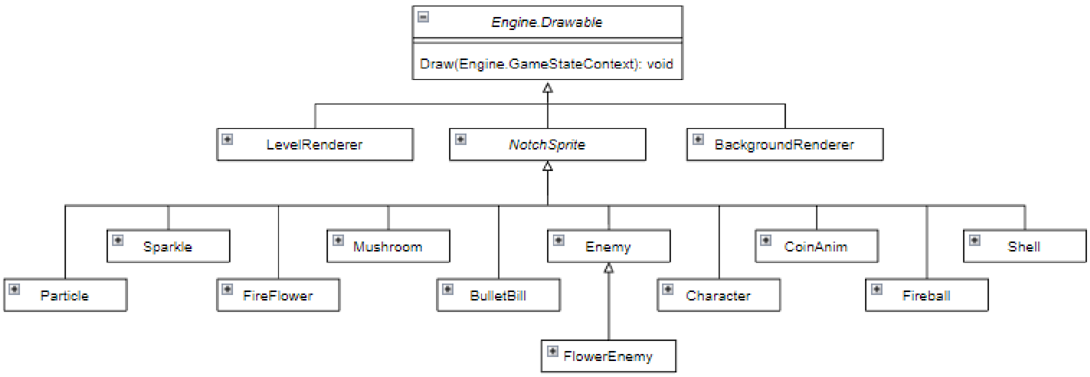

Figure 4.3: Infinite Mario Bros [45] drawable objects.

These game's states and drawable objects represent the overall architecture and composition of IMB, but not all were used for the studies reported in chapters 5 and 6. To answer the research questions pointed out in section 1.3, there was the need to modify part of this architecture by removing unnecessary states, changing the generator, and adding a way to track the players' actions, as detailed in the following subsection.

4.3 Game Architecture Modifications

After selecting the game and refactoring its code, it was almost ready to follow into the next phase of the methodology. However, some changes were needed to generate and import levels into the game, which were crucial for both studies.

As overviewed above, the original *Infinite Mario Bros* (IMB) [45] contained six states. Nevertheless, half of these states were unnecessary since the objective was to present a single level to the players each time. Therefore, the map, win, and lose states were removed from the game, resulting in a more straightforward sequence of states represented in figure 4.4.

The game states remain relatively similar, but with some changing in their names. The **Loading** state still loads everything into memory and continues to a new **Predefined Title** state. This state is an extension of the Title state seen in the previous section, allowing to show the game's controls (Arrows for moving, S for jumping, and A for moving faster and shooting) since some players tend to advance the surveys' explanations and try to learn the game's controls while playing. To move on to the **Predefined Level** state, the player must press the S key, entering one of the previously generated levels. Although this new state also acts as the normal Level state, it

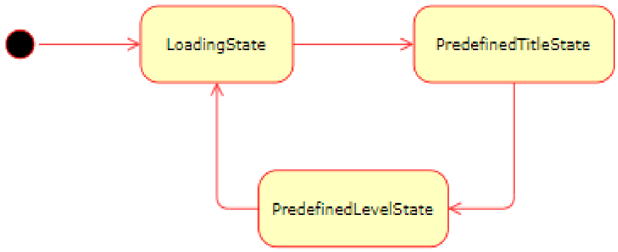

Figure 4.4: Prototypes' states.

only gives one life to the players, redirecting them to a new questionnaire or another level when completing or losing the level. Meanwhile, the game also moves to a new loading state, preventing players from playing the level again and avoiding possible errors in its execution.

While entering the Level state, IMB creates, with the LevelGenerator, a new Level with a random configuration. In these prototypes, these classes were also extended (figure 4.5), allowing not only to generate but also store and load levels to and from the *JavaScript Object Notation* (JSON) format. Furthermore, by decoupling the game sections (*JumpSection*, *TubeSection*, *StraightSection*, *HillStraightSection*, and *CannonSection*) and the enemies array from the levels map and data, which are almost enough to represent and render them, the generator could efficiently store the levels and send them with HTTP requests into a server, thus allowing investigators to reproduce the users' experimentations.

A crucial class in the game is *Mario*, which contains a reference to the character and the global map state. This class was slightly changed for these prototypes by adding a *PLAY_PROTOTYPE* boolean, allowing players to play the actual game or the prototypes. As can be seen in figure 4.6, the *MarioCharacter* variable points to a *Character* class, which was expanded by adding a new *gameplayMetrics* variable. This variable is of the type *GameplayMetrics*, one of the main contributions of this work. The *GameplayMetrics* class stores data from the gameplay, which is used to analyze players' behaviors, as described in sections 5.4 and 6.3, containing: (i) arrays with the jumps and landings distances to the nearest gaps, (ii) an array with the wall jumps positions, (iii) arrays with the IDs of coins, powerups, and enemies collected and killed, (iv) the number of coins, powerups, and enemies present in the level, (v) the players' cause of death, represented by the *CauseOfDeath* enumeration, (vi) the time left in the clock when the player leaves the level, by dying or completing it, (vii) an array with the players' actions, containing the keys, ticks and input events, which allows reproducing the experimentations, and (viii) the level state, to access the level's content.

The information about the players' actions is only possible to store and reproduce due to new classes developed for the prototypes, seen in figure 4.7: *Agent*, *PlayerAgent*, and *AIAgent*. The *Agent* class is abstract and allows to store the actions array during the game's execution, the

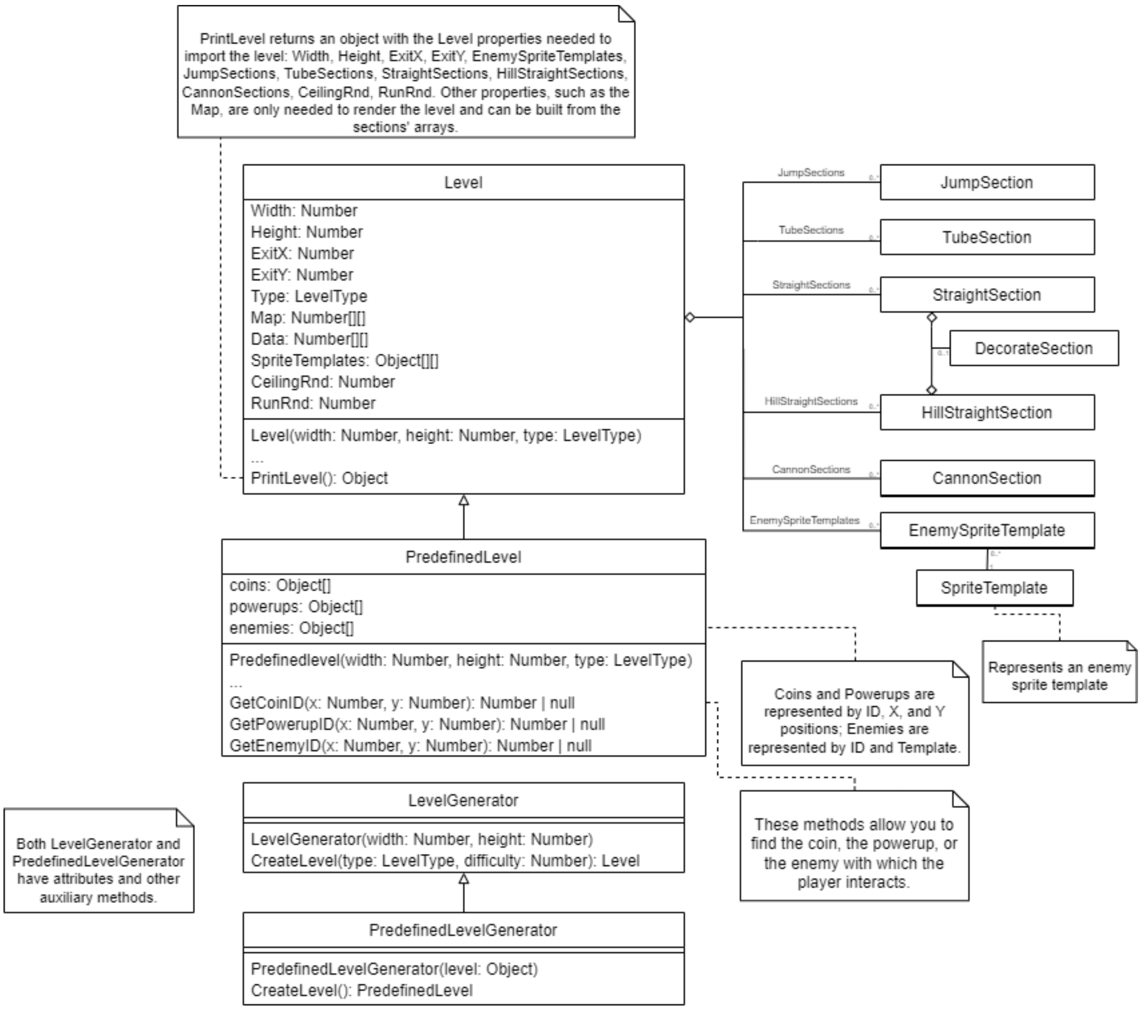

Figure 4.5: Prototype level and level generator.

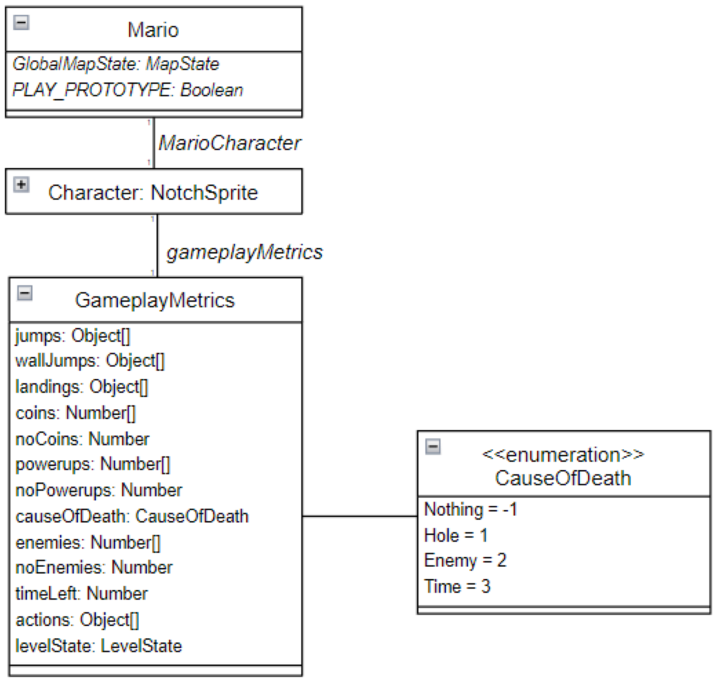

Figure 4.6: Mario abstract class.

number of ticks, and the time since the game started. Regarding the other two classes, they must be used in different runs. On the one hand, there is the *PlayerAgent*, used to record the players' actions by registering events of the game's input keys. On the other hand, the *AIAgent* is used to reproduce a previously recorded array of actions by triggering the specified events.

With the previously reported changes, the game is almost ready to be used in both studies, described in the following chapters (5 and 6), thus answering the research questions raised in section 1.3.

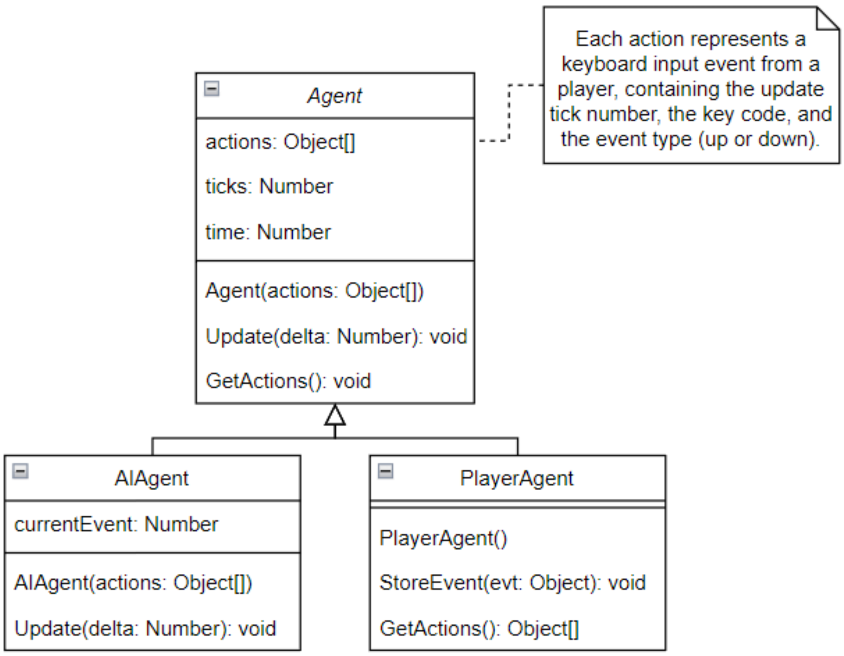

Figure 4.7: Agent classes, which allow to store and reproduce players' actions.

Chapter 5

Correlating Game Content with Players' Profiles

As mentioned in the chapter 3, the methodology proposed in this work consists of three phases. In the previous chapter (chapter 4), the game selection stage was described as well as the *Infinite Mario Bros* (IMB) [45] architecture and the modifications made to it.

The current chapter describes the second stage of the methodology (First Study) and answers the first research question raised in section 1.3. It starts by mentioning the preliminary work, mainly composed of the selection phase. Then the survey and the specific modifications to the game made for the prototype are described. Subsequently, the experience's deployment and the data collection methods are described. Finally, in the last section, the study's results are exposed, setting the way for the next step of the methodology, the second study.

Note that since the first study's results can be connected with the results from the second study, their discussion is made in chapter 7.

5.1 Preliminary Work

As mentioned above, most of the preliminary work made for this study was composed of the game selection stage, described in section 4.1. In that phase, the platform game genre was chosen as well as IMB since it included a level generator and a port for *JavaScript* (JS).

Besides the game selection, there was the need to create and distribute questionnaires about profiles and the experience in the levels. Since they should be answered during the playthrough, the questionnaires should be embedded in the game or vice-versa. After reviewing ways to do it, *SurveyJS*[1], a collection of libraries that allows creating JS surveys and forms, was chosen. With these libraries, the game's levels could be embedded in the survey's pages, combining the questionnaires with the levels.

[1] https://surveyjs.io/

In the following section, this integration is better explained, showing the content of each questionnaire and level.

5.2 Survey and Game Modifications

After selecting a game genre, and the game with a levels generator, a survey was created with *SurveyJS*. The purpose of the survey was to correlate players' profiles, particularly players' personalities, with game elements. It was divided into three main sections: (i) player profiling, where questionnaires regarding demographics, personality, and gaming experience, were presented to the player, (ii) levels, wherein the user played a set of pre-generated levels and answered questions concerning their experience in that level, and (iii) comments, in which players could provide feedback regarding the whole experience in the survey.

Each of the previous sections corresponds to an HTML page. The player profiling section is the index page, where the general instructions of the game and the survey are explained. Then the survey redirects to the game page, where users can play twelve pre-generated levels of IMB, which are loaded using the *PredefinedLevelGenerator* described in section 4.3. Finally, after playing all the levels, the survey redirects to the comments page, where users can provide feedback about their experience.

Each level evaluates a specific element by increasing or decreasing its amount. This study focused on the enemies' number and types, the width of gaps, and the number of powerups and coins. These elements, seen in table 5.1, were selected since they are common to most platform games, meaning that the results could be extrapolated to more games. Each of these elements was associated with an identifier, allowing to gather data about them and understand which influenced the players' experience more. Game designers can use this information to improve their games' level design.

			
Coin	Powerups	Enemies	Gaps

Table 5.1: Game Elements.

To show every level to the player and ensure the data was correctly collected, some values needed to be stored in the browser's local storage during the experience. These values consist of (i) a global unique identifier (GUID) that distinguishes users, (ii) the levels to avoid fetching them every time the player loads a page, (iii) the levels' order to ensure users play every level in a randomly generated order, and (iv) the current level users are playing. With these modifications, the survey was ready to be deployed.

The following subsections explain in detail each section of the survey. For a better view of the questionnaires, the reader should refer to appendix A. The survey's results are presented in section 5.4, but their discussion comes only in chapter 7.

5.2.1 Profile Questions

As mentioned above, the first section of the survey was related to profiling questions and included on the main page of the survey. Besides that, it also shows the general instructions of the survey, i.e., how to answer the questions, how to play the game, and how data was collected and stored during the study. The profiling questionnaire was divided into three pages, detailed below, and can be seen in appendix A.

5.2.1.1 Demographics

The first section of the questionnaire (appendix A.1) consists of demographic questions. These questions were designed to determine who answered the survey without compromising users' privacy and safety. Therefore, three questions were designed to collect the users' gender, age, and level of education and produce statistics presented in section 5.4.

5.2.1.2 Personality

The second section of this questionnaire (appendix A.2) contains questions to infer users' personality traits. To that end, the BFI-10 [46], described in section 2.2.4, was used. It consists of 11 questions that infer the players' Big Five traits: Agreeableness, Extraversion, Conscientiousness, Openness, and Neuroticism.

These questions were the central part of the questionnaire, allowing to correlate later the players' personalities with game elements, using the values for each trait.

5.2.1.3 Gaming Experience

The last section of the profiling questionnaire (appendix A.3) involves the users' gaming experience. These are also questions used to infer the type of people answering the survey and if their previous experience with games, namely *Super Mario Bros.*, could influence the outcome of the results. The questions focus on four key aspects: experience (i) with video games in general, (ii) with platform games, (iii) playing *Super Mario Bros.*, and (iv) with the *Super Mario Bros.* mechanics.

5.2.2 Levels

The second section of the survey was composed of twelve levels, each to evaluate a specific game element. The levels were automatically generated, by increasing or decreasing the chance to spawn the specific element to be evaluated, with an altered version of *Infinite Mario Bros* [45] and then stored in JSON files.

To quickly identify the content, each level was divided into distinct sections representing the background and foreground of the game, and the game elements were decoupled from these representations. With these modifications, other researchers can easily use the same levels or generate new ones, to conduct various investigations.

After playing a level, the users had to answer a small questionnaire (appendix A.4) with six questions related to their experience with enemies, enemy types, the width of gaps, coins, and powerups, in the level they had just played. If the users did not see the element, they had to answer that the element did not apply to the level, which allowed to exclude outliers from the analysis of results.

Table 5.2 describes the twelve pre-generated levels, their purpose, and the number of evaluated game elements, i.e., the number of enemies, enemies' types, gaps, powerups, and coins. The levels were automatically generated, by exaggerating the evaluated element, i.e., by significantly increasing or decreasing its amount or length. Although the levels are presented systematically, they were randomly shown to the user, i.e., the levels were shuffled before the user started the section.

5.2.3 Comments

The third and final section of the questionnaire was presented to the user after playing the twelve levels. It is composed only of a comments box, allowing testers to give feedback about the general experience with the experiment. Some of these comments will be highlighted in section 5.4 and discussed in chapter 7.

5.3 Experience Deployment and Data Collection

After the development of the survey, the game, developed using *JavaScript*, was deployed on an *itch.io* page[2]. This allowed people to run and answer it using only a browser, which is accessible to everyone that has a computer and an Internet connection. Users could also run and answer the survey on mobile devices but could not play the game since it required a keyboard.

After the deployment, the link for *itch.io* was spread around several communities, including *itch.io*, *Facebook*, *Discord*, *Twitter*, and the *University of Porto*. With this dissemination, the survey reached a good sample size of 282 people, as described in section 5.4.

Since the questionnaires were integrated with the game levels and *itch.io*, it was necessary to find a way to store their results and data collected during gameplay, including the cause of death, if applicable, the wall jumps position, the jumps, and landings distance to the nearest gap, the time left on the clock when the player finishes the level, by dying or winning, the identifiers of the coins and powerups collected, the identifiers of the enemies killed, and the players' actions, i.e., their inputs.

The most reliable way to collect and store the data was to use *Google Spreadsheets* and *Google App Scripts* to deploy a web application capable of receiving and storing *POST* and *GET* requests from the game using a script (detailed in appendix C.1). Each request needed to be correctly identified with the target sheet, allowing to separate each level's data, the demographics, and the

[2] https://pemesteves.itch.io/game-design-adaptivity

Level	Description	Enemies	Enemy Types	Gaps	Powerups	Coins
1 - Average Number of Enemies	Basis for enemy testing. Contains an average number of enemies and enemy types.	17	3	1	1	77
2 - Average Gap Width	Basis for gap width testing. It only contains gaps with an average width	0	0	25	0	0
3 - Fewer Enemies	Enemy testing. Fewer enemies than the basis (Level 1).	6	4	2	5	51
4 - Fewer Enemy Types	Enemy testing. Fewer enemy types than the basis (Level 1).	10	2	1	2	41
5 - More Coins, More Powerups	Collectibles testing. Higher number of powerups and coins	14	4	0	6	70
6 - More Enemies	Enemy testing. Higher number of enemies, and enemy types.	30	7	9	1	21
7 - More Enemy Types	Test enemy types by increasing them	27	7	3	0	54
8 - Narrower Gaps	Tests the gap width and, as level 2, it only has jump sections.	0	0	14	0	0
9 - No Coins, No Powerups	Every coin and powerup was removed.	13	5	1	0	0
10 - No Enemies	The enemies were removed.	0	0	3	6	41
11 - No Enemies, No Gaps	Similar to the previous, but the jump sections were also removed.	0	0	0	8	52
12 - Wider Gaps	Tests gaps, by increasing their width.	0	0	19	0	0

Table 5.2: First study pre-generated levels and their elements' configuration.

comments between different spreadsheets. Although the waiting to send the requests produced some delay between levels, it ensured data was saved correctly.

5.4 Results

After three weeks of collecting data, the dataset was analyzed. During that time, the questionnaire received 308 answers, where 26 were duplicates, possibly from people experiencing issues during the game. In other words, 282 people contributed to the study, where 115 completed the questionnaire, playing all the levels.

Of these 115, 37 commented on the overall experience. Most people liked the survey because it allowed them to play a game between the questionnaires instead of only answering an exhaustive questionnaire. Some people experience issues with the frame rate, stating that it should be increased, which can be just a *Mozilla Firefox* problem, as investigated afterward. Others just commented on the game mechanics and their answers to the previous questions.

Regarding the sample's gender, 192 (68%) were men, 83 (29.4%) were women, 6 (2.1%) did not answer, and one person did not identify with these genders. The sample's age varied between 11 and 51 years old, with an average of 22.954 and a standard deviation of 5.54. Regarding the sample's education degree, 129 (45.7%) had completed high school, 88 (31.2%) a bachelor's, 51 (18.1%) a master's, 7 (2.5%) a Ph.D., 4 (1.4%) were still in high school, and the other three people (1.1%) did not answer the question. Due to these varying results, it was impossible to find correlations with demographic data.

Due to an error in the data collection, data about the users' gaming experience were not stored for a large part of the population, more precisely 72.7% of the sample. The remaining percentage counted with people more experienced in video games, as seen in table 5.3, but conclusions cannot be drawn due to data loss. This sample contained 56 men (72.73%), 19 women (24.68%), and two people (2.6%) that did not answer the gender question. Their age also varied between 11 and 51 years old, but the average was 22.74, with a standard deviation of 6.012. Regarding their education, 42 (54.55%) had completed high school, 19 (24.68%) a bachelor's, 9 (11.69%) a master's, 3 (3.9%) a Ph.D., 1 (1.3%) person was still in high school, and the other three people (3.9%) did not answer the question.

Question	**Disagree strongly**	**Disagree a little**	**Neither agree nor disagree**	**Agree a little**	**Agree strongly**
Video Games (in general)	2 (2.60%)	3 (3.90%)	4 (5.19%)	20 (25.97%)	48 (62.34%)
Platformer	4 (5.19%)	7 (9.09%)	5 (6.49%)	21 (27.27%)	40 (51.95%)
Super Mario Games	5 (6.49%)	6 (7.79%)	0 (0.00%)	26 (33.77%)	40 (51.95%)
Super Mario Mechanics	4 (5.19%)	3 (3.90%)	3 (3.90%)	30 (38.96%)	37 (48.05%)

Table 5.3: Gaming experience data from the first study population.

As described above, during the levels, besides the questionnaires about player preferences, some gameplay metrics and data about elements were collected and used to produce statistics and correlate elements with players' profiles. These properties are (i) the cause of death, which could be -1, 1, 2, 3, if the player did not die, die in a hole, was killed by an enemy, or lost by time, respectively, (ii) the jumps array, with distances to the next gap while jumping, (iii) the wall jumps array, with the positions where these actions happened, (iv) the landing array, with distances to the previous gap while landing after a jump, (v) the time left when completing a level or dying, (vi) the number of coins collected and an array with their IDs, (vii) the number of powerups collected and an array with their IDs, (viii) the number of enemies killed and an array with their IDs, and (ix) the array with the player's actions.

In the following subsection, the results from each level are analyzed more in-depth. Then levels with the same purpose are compared. Finally, these results are summarized, answering the first research question raised in section 1.3, and setting the way for the second study.

5.4.1 Level Data

In this subsection, the results from each level are analyzed more extensively. The results were produced mainly with *RapidMiner*[3], a data and predictive analysis tool. Besides *RapidMiner*, a *Python* script, available in appendix C.2, was used to generate charts regarding the collection of certain game elements. These charts are available in appendix B.3, and game designers can use the process of gathering their data and creating them on their games to determine what elements have more impact on the game's experience.

After carefully analyzing each level's data [16], no correlation between players' personalities and game elements could be found, as seen through the tables in appendix B.2. However, some interesting results related to the players' stated preferences were deeply investigated and are presented in the following subsubsections. Each contains:

- A correlation matrix between the overall experience, several other game properties, and players' preferences. Only the most significant correlations are specified, i.e., correlations greater than 0.25 and lesser than -0.25;

- Data collected through the visualization of a decision tree (all decision trees are available in appendix B.1), i.e., the tree's leaves with a more significant flow;

- The accuracy of applying cross-validation to the previous decision tree to predict its values;

- Statistics about the gameplay data and the overall experience in the level.

Besides these analysis strategies, other analysis methods, such as the Principal Component Analysis and clustering with k-means and x-means, were explored. Still, they were discarded since they did not yield any relevant results, and machine learning was out of this work's scope.

[3]https://rapidminer.com/

Since the work's objectives were to improve the players' experience, there was no need to distinguish between a good and an excellent one. Although users needed to state their experience on a scale of 1 (Very dissatisfied) to 5 (Very satisfied), these values were grouped into three categories. The resulting categories are (i) **Dissatisfied**, which contained the "Dissatisfied" and "Very dissatisfied" answers, (ii) **Neutral**, which included the "Neutral" and "Not applicable to the previous level" responses, and (iii) **Satisfied**, which contained the "Satisfied" and "Very satisfied" answers.

5.4.1.1 Level 1 - Average Number of Enemies

During the study, level 1 was used as the basis for enemy testing, containing an average number of enemies and enemy types. As seen in table 5.4, the most promising correlations show that, in the players' opinion, liking the enemies' number is the essential factor in getting a better experience, followed by the number of coins, enemy types, and powerups. Spending more time in the level, i.e., having less time left in the clock, is slightly correlated to the overall experience, probably because some players got killed very early, worsening their experience. The numbers of coins and powerups collected are also slightly correlated with the overall experience, i.e., the more collectibles the players gather, the better their experience is.

Property	**Correlation Coefficient**
Feeling about the enemies' number	0.622
Feeling about the coins' number	0.597
Feeling about the enemy types' variety	0.571
Feeling about the powerups' number	0.415
Feeling about the gap's width	0.391
Number of coins collected	0.344
Number of powerups collected	0.335
Number of enemies killed	0.277
Time left	-0.364

Table 5.4: Level 1 - correlations with overall stated experience.

The decision tree shows a big cluster where the Satisfied value can be predicted, gathering 63 Satisfied, 48 Neutral, and 15 Dissatisfied answers. This happens with the following condition:

$$timeLeft \in]133.759, 197.079] \cap noCoins <= 54.5$$

With this tree, a cross-validation strategy could predict satisfaction with an accuracy of 43.75%, which is not enough to use in a recommendation system.

Finally, the statistics show that, from the 155 people who played and evaluated the level, 80 (51.61%) were satisfied, 17 (10.97%) were dissatisfied, and 58 (37.42%) were neutral about it.

5.4.1.2 Level 2 - Average Gap Width

Level 2 was used as the basis for the gap width testing, containing gaps, i.e., holes where the player may fall and die, with an average width. Since it was only made for testing the gaps, other elements, such as coins, powerups, and enemies, were removed. The only relevant correlation shows that people who liked the gap's width also had a satisfying experience, with a coefficient of 0.616.

The decision tree only shows a leaf with the most Dissatisfied values. This means that most people were dissatisfied with the level, and there are no ways to separate them from others using only the gameplay data and preferences provided. With this tree, a cross-validation strategy could predict satisfaction with an accuracy of 42.67%, which is the value for predicting the Dissatisfied value.

Finally, the statistics show that, from the 150 people who played and evaluated the level, 33 (22%) were satisfied with the level, 64 (42.67%) were dissatisfied, and 53 (35.33%) were neutral about it. As expected, most people disliked or did not have an opinion about the level since it lacked elements besides gaps.

5.4.1.3 Level 3 - Fewer Enemies

The third level tests the enemy number, having fewer enemies than the basis, i.e., level 1. As seen in table 5.5, the most promising correlations show that, in the players' opinion, liking the gap's width is the most crucial factor in getting a better experience, followed by the number of coins and enemies. This time, the time spent in the level has a less significant correlation coefficient, which can be derived from the fewer enemies, thus a more accessible level. The number of coins is also slightly correlated with the overall experience, i.e., the more coins the players gathered, the better their experience was.

Property	**Correlation Coefficient**
Feeling about the gap's width	0.566
Feeling about the coins' number	0.558
Feeling about the enemies' number	0.540
Feeling about the powerups' number	0.477
Feeling about the enemy types' variety	0.451
Number of coins collected	0.295
Time left	-0.291

Table 5.5: Level 3 - correlations with overall stated experience.

The decision tree shows a big cluster where the Satisfied value can be predicted, gathering 85 Satisfied, 25 Neutral, and only 8 Dissatisfied values. This happens with the following condition:

$$noCoins > 24.5 \cap noPowerups > 1.5$$

With this tree, a cross-validation strategy could predict satisfaction with an accuracy of 68.17%, which is better than its basis.

Finally, the statistics show that, from the 151 players who evaluated the level, 102 (67.55%) were satisfied with the level, 13 (8.61%) were dissatisfied, and 36 (23.84%) were neutral about it.

5.4.1.4 Level 4 - Fewer Enemy Types

Level 4 tests the enemy types by presenting a fewer number than level 1.

As seen in table 5.6, the most promising correlations show that, in the players' opinion, liking the coins' number is essential in getting a better experience, with an impressive correlation coefficient of 0.739. The feeling about the number of enemies, variety of enemy types, and the number of powerups come after, also with solid coefficients. The time spent at the level has a significant correlation coefficient, showing that spending more time at the level contributes to a better experience. The numbers of coins and powerups collected and the number of enemies killed somewhat correlate with the overall experience, i.e., the more the players interacted with these elements, the better their experience was.

Property	Correlation Coefficient
Feeling about the coins' number	0.739
Feeling about the enemies' number	0.692
Feeling about the enemy types' variety	0.638
Feeling about the powerups' number	0.563
Feeling about the gap's width	0.501
Number of coins collected	0.455
Number of enemies killed	0.432
Number of powerups collected	0.375
Time left	-0.455

Table 5.6: Level4 - correlations with overall stated experience.

The decision tree shows two leaves with a large flow. The first predicts Satisfied, from a set with 20 Satisfied and 3 Neutral values, with the following condition:

$$noEnemies <= 7.5 \cap timeLeft <= 129.431$$

The second predicts Neutral aggregating 27 Satisfied answers, 12 Dissatisfied, and 39 Neutral, by using the constraint:

$$noEnemies <= 6.5 \cap timeleft > 160.873 \cap nopowerups <= 0.5$$

This last leaf does not have much interest in recommending or adapting levels since it predicts a neutral experience, and the objective with adaptivity is to improve the experience or engagement of players. With this tree, a cross-validation strategy could predict satisfaction with an accuracy of 52.64%.

Finally, the statistics show that, from the 135 testers who evaluated the level, 66 (48.89%) were satisfied with the level, 17 (12.59%) were dissatisfied, and 52 (38.52%) were neutral about it.

5.4.1.5 Level 5 - More Coins, More Powerups

The fifth level is the first for testing the collectibles, i.e., the powerups and coins, presenting a higher number of them. As expected and viewed in table 5.7, the most promising correlations show that the players' experience is hand-to-hand with the feeling about the collectible, i.e., people who like the number of coins enjoyed the level, and vice-versa. For the first time until now, the number of coins was correlated with satisfaction with a coefficient greater than 0.5, showing that the more coins players' collected, the better their experience in the level was. As already discussed in other levels, the time left in the game's clock also significantly correlates with the overall experience, which can be explained by people dying too soon disliking the level.

Property	**Correlation Coefficient**
Feeling about the coins' number	0.791
Feeling about the enemy types' variety	0.753
Feeling about the enemies' number	0.736
Feeling about the powerups' number	0.700
Number of coins collected	0.588
Number of powerups collected	0.485
Feeling about the gap's width	0.374
Time left	-0.506

Table 5.7: Level 5 - correlations with overall stated experience.

The decision tree of this level found a big cluster for the Satisfied values, gathering 94 Satisfied answers, 5 Dissatisfied, and 10 Neutral, by using the following constraint:

$$timeLeft <= 197.558 \cap noCoins > 9.5$$

This shows that if the players collected ten or more coins and spent more than 2.5 seconds in the game, they most probably liked the level. The cross-validation strategy could predict satisfaction with an accuracy of 74.78%, which is a great result.

Finally, the statistics show that, from the 134 people who evaluated the level, 98 (73.13%) were satisfied with the level, 11 (8.21%) were dissatisfied, and 25 (18.66%) were neutral about it. This means that almost everyone liked the level or had no opinion, showing that increasing the number of collectibles can improve the players' experience.

5.4.1.6 Level 6 - More Enemies

The sixth level also tested the enemies' number, increasing it and, consequently, increasing the variety of types. As seen in table 5.8, the most promising correlations show that the players have a better experience when they like the enemies' number, followed by the number of enemy types, coins, and powerups. At this time, the time left has a low correlation coefficient, which can be explained by the higher level's difficulty, leading players to lose too soon.

Property	Correlation Coefficient
Feeling about the enemies' number	0.705
Feeling about the enemy types' variety	0.655
Feeling about the coins' number	0.427
Feeling about the powerups' number	0.356
Feeling about the gap's width	0.344
Number of enemies killed	0.290
Number of coins collected	0.252
Time left	-0.267

Table 5.8: Level 6 - correlations with overall stated experience.

The decision tree's outcome is irrelevant because it only aggregates a high number of Neutral answers, resulting in poor accuracy of 37.82% using a cross-validation strategy.

Finally, the statistics show that, from the 127 players who evaluated the level, 53 (41.73%) were satisfied with the level, 31 (24.41%) were dissatisfied, and 43 (33.86%) were neutral about it.

5.4.1.7 Level 7 - More Enemy Types

Level 7 also tests enemies, increasing the number of enemy types, although it also increases the number of enemies. Table 5.9 shows that most people need to like the enemies' numbers and the variety of enemies to have a good experience.

Property	Correlation Coefficient
Feeling about the enemies' number	0.612
Feeling about the enemy types' variety	0.593
Feeling about the coins' number	0.449

Table 5.9: Level 7 - correlations with overall stated experience.

The decision tree shows two groups where Satisfied values can be predicted. The first group gathers 24 Satisfied values, 19 Neutral and 7 Dissatisfied, under the following condition:

$$noCoins \in]8.5, 25.5] \cap noEnemies \in]1.5, 5.5] \cap timeLeft <= 194.761$$

The second group aggregates 11 Satisfied answers, 9 Neutral and 3 Dissatisfied, under the following constraint:

$$noCoins \in]8.5, 25.5] \cap noEnemies <= 0.5 \cap timeLeft <= 194.761$$

This means that collecting between 9 and 25 coins, spending more than 5 seconds in the game, and killing between 2 and 5 enemies or none makes the experience better in this level. However, a cross-validation strategy applied to this tree could only predict satisfaction with an accuracy of 44.56%.

Finally, the statistics show that, from the 139 testers who evaluated the level, 67 (48.2%) were satisfied with the level, 22 (15.83%) were dissatisfied, and 50 (35.97%) were neutral about it.

5.4.1.8 Level 8 - Narrower Gaps

The narrower gaps level, as the name says, tests the gap width by decreasing it, and, as in level 2, it only has jump sections. As in level 2, the feeling about the gap's width is the only correlation found with the overall experience, with a coefficient of 0.622, which is in line with the level's purpose.

Since most people disliked the level and there were few parameters on the level, the decision tree could not predict any outcome except the Dissatisfied value. However, the cross validation's accuracy was only 54.47%.

Finally, the statistics show that, from the 128 people evaluating the level, 24 (18.75%) were satisfied with the level, 71 (55.47%) were dissatisfied, and 33 (25.78%) were neutral about it.

5.4.1.9 Level 9 - No Coins, No Powerups

In level 9, collectibles were deleted, removing any score mechanic from the game. In Table 5.10, the most promising correlations show that, in the players' opinion, liking the enemies' number is the most crucial factor to get a better experience, followed by the enemy types variety and the gaps' average width. However, the results show that people liked coins and powerups, meaning they did not pay attention to the survey's instructions or the level's elements. The numbers of enemies killed are also slightly correlated with the overall experience, i.e., the more enemies the players kill, the better their experience is, probably because it inserts some excitement in the game.

Property	Correlation Coefficient
Feeling about the enemies' number	0.682
Feeling about the enemy types' variety	0.639
Feeling about the gap's width	0.601
Feeling about the coins' number	0.496
Feeling about the powerups' number	0.424
Number of enemies killed	0.308
Time left	-0.344

Table 5.10: Level 9 - correlations with overall stated experience.

The level's decision tree found a cluster that it can predict as Satisfied, gathering 48 Satisfied answers, 43 Neutral, and 16 Dissatisfied, with the following condition:

$$timeLeft <= 195.298$$

This means that the more time the players take in this level, the better their experience, which makes sense since less than 5 seconds of playtime means the player was killed almost instantaneously. The cross-validation strategy could predict satisfaction with an accuracy of 44.55%, which is also not enough to use in a recommendation system.

Finally, the statistics show that, from the 129 players who evaluated the level, 48 (40.34%) were satisfied with the level, 20 (16.81%) were dissatisfied, and 41 (42.86%) were neutral about it.

5.4.1.10 Level 10 - No Enemies

In level 10, the enemies were deleted to test if this type of challenge was needed Table 5.11, the most promising correlations show that, in the players' opinion, liking the average width of gaps is the most crucial factor to getting a better experience at this level, which makes sense since it creates some challenge to them. As in other levels, spending more time in it is slightly correlated to the overall experience. The numbers of coins and powerups collected are also slightly correlated with the overall experience, i.e., the more collectibles the players gather, the better their experience is.

Property	**Correlation Coefficient**
Feeling about the gap's width	0.552
Feeling about the powerups' number	0.55
Feeling about the coins' number	0.542
Number of coins collected	0.428
Number of powerups collected	0.377
Time left	-0.428

Table 5.11: Level 10 - correlations with overall stated experience.

In level 10, the decision tree created three big groups, where two predict the Satisfied answer and one the Neutral. The first prediction of Satisfied aggregates 15 Satisfied answers, 1 Dissatisfied, and 6 Neutral, by applying the following constraint:

$$timeLeft <= 142.73 \cap noPowerups > 5.5$$

The second prediction groups 15 Satisfied answers, 15 Dissatisfied, and 6 Neutral with the following condition:

$$timeLeft <= 142.73 \cap noPowerups \in]2.5, 4.5]$$

The cross validation's performance vector shows an accuracy of 41.21%.

Finally, the statistics show that, from the 131 players evaluating the level, 40 (33.53%) were satisfied with the level, 44 (33.59%) were dissatisfied, and 47 (35.88%) were neutral about it, meaning that the level divided players probably by their gaming experience or tastes for gaming challenges.

5.4.1.11 Level 11 - No Enemies, No Gaps

Level 11 does not contain enemies like the previous level, and the gaps were deleted, completely removing all challenges.

Table 5.12 shows that the feeling about the coins and powerups numbers is the most crucial factor in getting a better experience at these levels.

Property	**Correlation Coefficient**
Feeling about the coins' number	0.557
Feeling about the powerups' number	0.532

Table 5.12: Level 11 - correlations with overall stated experience.

The level's decision tree can predict both the Satisfied and the Dissatisfied. Regarding the Satisfied cluster, it gathers 10 Satisfied answers, 3 Dissatisfied, and 3 Neutral, according to the following constraint:

$$noPowerups > 7.5 \cap timeLeft <= 126.904$$

The Dissatisfied cluster aggregates 15 Satisfied answers, 24 Dissatisfied, and 14 Neutral, with the condition:

$$noPowerups <= 2.5$$

The tree could predict satisfaction through cross-validation with an accuracy of 34.13%.

Finally, the statistics show that, from the 141 testers evaluating the level, 48 people (34.04%) were satisfied with the level, 55 (39.01%) were dissatisfied, and 38 (26.95%) were neutral about it. These results are similar to the ones of the previous level, showing similar values for every satisfaction level. However, the dissatisfaction degree increased a little, which can be due to the lack of challenges at the level.

5.4.1.12 Level 12 - Wider Gaps

The last level tests gaps by increasing their width. As in the other gaps' testing levels, it only contains jump sections. As seen in table 5.13, the players' experience was related to the feeling about the gap's width, which was expected since the levels do not contain any other element. Spending more time in the level also correlates a little with the overall experience since players who performed more jumps probably liked the level better.

Property	**Correlation Coefficient**
Feeling about the gap's width	0.749
Time left	-0.320

Table 5.13: Level 12 - correlations with overall stated experience.

The decision tree shows, as expected, a Dissatisfied cluster, gathering 12 Satisfied, 34 Neutral, and 61 Dissatisfied answers under the following condition:

$$timeLeft > 180.461$$

This means that most players who cannot spend more than 20 seconds in the level disliked it. The cross-validation strategy could predict the satisfaction with an accuracy of 53.41% due to the high dissatisfaction in the previous cluster.

Finally, the statistics show that, from the 118 people who played and evaluated the level, only 19 (16.1%) were satisfied with the level, 65 (55.08%) were dissatisfied, and 34 (28.81%) were neutral about it.

5.4.2 Level Comparison

After exposing the results for each level individually, they can be compared by their purpose. In the following subsubsections, these comparations are made regarding certain elements. For instance, the first subsubsection compares levels 2, 8, and 12 since they aimed to measure the preference for the average width of gaps. In each comparison, the satisfaction and dissatisfaction degrees, i.e., the amount of satisfied and dissatisfied answers divided by the total number of responses, are exposed, as well as a possible explanation for the results.

5.4.2.1 Gaps

As stated above, the gaps testing comprised levels 2, 8, and 12. Table 5.14 represents the satisfaction and dissatisfaction levels of each of these levels. By analyzing the table, it can be concluded that the basis for this type of testing produced greater satisfaction and lesser dissatisfaction. This could be due to the lack of challenge with small gaps and a massive challenge with big holes, which need a great degree of expertise. Levels that only contain gaps are not good in general, as can be seen by the difference between the levels of satisfaction and dissatisfaction. These levels need more elements such as enemies, collectibles, and sections without jumps to make the players feel they are playing an actual Super Mario level.

	8 - Narrower	**2 - Average Width**	**12 - Wider**
Satisfaction/Total	0.188	0.220	0.161
Dissatisfaction/Total	0.555	0.427	0.551

Table 5.14: Comparison between the levels that tested the gaps' width.

5.4.2.2 Number of Enemies

The number of enemies testing was mainly composed of levels 1, 3, and 6, but level 10 was added to this section since it did not contain enemies. Table 5.15 represents the satisfaction and dissatisfaction levels of each of these levels. By analyzing the table, it can be concluded that decreasing the number of enemies produces more satisfaction in the players. However, removing every enemy from the game makes the experience neutral. Players need challenges in a game, but they cannot be too exaggerated. So, it can be concluded that it is good to have some challenges by introducing enemies in levels. Still, they should not be exaggerated, i.e., their number must remain relatively low.

	10 - None	**3 - Fewer**	**1 - Average Number**	**6 - More**
Satisfaction/Total	0.305	0.675	0.516	0.417
Dissatisfaction/Total	0.336	0.086	0.107	0.244

Table 5.15: Comparison between the levels that tested the enemies' number.

5.4.2.3 Number of Enemy Types

The variety of enemy types was tested with levels 1, 4, and 7. Table 5.16 represents the satisfaction and dissatisfaction levels of each of these levels. Analyzing the table, there does not seem to be much difference between each level. However, an average level of enemy types produced a slightly higher satisfaction and lower dissatisfaction. More types usually mean greater difficulty, and fewer types mean the opposite, which can be better for experienced and non-experienced players. However, this variation is not substantial.

	4 - Fewer	**1 - Average Number**	**7 - More**
Satisfaction/Total	0.489	0.516	0.482
Dissatisfaction/Total	0.126	0.107	0.158

Table 5.16: Comparison between the levels that tested the enemy types.

5.4.2.4 Number of Coins and Powerups (Collectibles)

The number of collectibles was tested with levels 1, 5, and 9. Table 5.17 represents the satisfaction and dissatisfaction levels of each of these levels. By analyzing the table, it can be concluded that increasing the number of enemies produces more satisfaction in the players. A possible explanation is that coins are fun to collect and a goal whereby people play the game, and the powerups make the game easier by helping defeat the enemies. This can potentially help newbies and also motivate experienced players to continue playing the game.

	9 - None	**1 - Average Number**	**5 - More**
Satisfaction/Total	0.403	0.516	0.731
Dissatisfaction/Total	0.168	0.107	0.082

Table 5.17: Comparison between the levels that tested the collectibles' number.

5.4.2.5 Level with no challenges (no enemies and no gaps)

The last test was made with level 11, the level without enemies and gaps, which everyone could quickly complete. By analyzing table 5.18, which represents its levels of satisfaction and dissatisfaction, it can be concluded that not having challenges leads to a neutral experience since players only need to run through the level to complete it. This can be good for new players to learn the basic game's mechanics (moving, jumping, and collecting items) but bad for the experienced ones that want to be challenged by the game.

	11 - No Enemies, No Gaps
Satisfaction/Total	0.340
Dissatisfaction/Total	0.390

Table 5.18: Data from the level without challenges.

5.4.3 Summary

In this section, the first study's results were presented level by level and as a comparison of levels by purpose. With its completion and the analysis of its results, some conclusions could be drawn to try to answer the first research question raised in section 1.3: Do players' profiles correlate with their preference for specific game elements?

Although several analyses were made with the data collected from the study, they could not answer this question. So, it was not possible to correlate players' profiles with their preferences for specific game elements.

Since this work needed the answer to this question to continue, some modifications to the original objectives were made. This way, the second study, described in chapter 6, focused on generating levels that improved the overall experience instead of developing them according to the players' profiles.

Some of the results presented in this chapter were crucial, mainly when comparing levels by their purposes. As concluded, the players' overall experience did not improve when changing the variety of enemy types and gaps' average width. However, the experience improved when the number of enemies decreased, without eliminating them all, and when the number of collectibles increased. These results were used for the second study as thoroughly detailed in the following chapter.

Several results were described in this chapter; some did not yield the expected outcome and should be better discussed. Therefore, chapter 7 outlines the study's results by discussing possible reasons for those results and additional ways to analyze the data collected in future studies.

Chapter 6

Improving Player Experience through Offline Adaptivity

As mentioned in chapter 3, the methodology proposed in this work consists of three phases: (i) the selection stage, described in section 4.1, (ii) the first study, detailed in chapter 5, and (iii) the second study, explained in this chapter.

As concluded in the first study, the players' experience improved when the number of enemies decreased, without eliminating them all, and when the number of collectibles increased. These results were used in the second study by developing an adaptive version of the game, which forced to modify the previous prototype to generate and adapt levels.

So, the current chapter describes the third and last stage of the methodology (Second Study) and answers the second research question raised in section 1.3. It starts by explaining the survey and the specific modifications to the game, allowing the implementation of an A/B methodology with regular and adapted versions of levels. Subsequently, the experience's deployment and the data collection methods are briefly described since they are similar to what was made in the previous study. Finally, the study's results are exposed in the last section, concluding the research and this thesis' work.

Note that since the second study's results can be connected with the results from the previous research, their discussion is made in chapter 7.

6.1 Survey and Game Modifications

As for the previous study, a survey was created with *SurveyJS* to measure the impact of generating and adapting levels tho the overall players' preferences for game elements. The survey was divided into three main sections: (i) player profiling, where questionnaires regarding demographics, personality, and gaming experience, were presented to the player, (ii) levels, wherein the user played an automatically generated level and an adapted version of it, and (iii) preferences and comments, in which players could choose the level they preferred and provide feedback regarding the whole experience in the survey.

Each of the previous sections corresponds to an HTML page. The player profiling section is the index page, where the general instructions of the game and the survey are explained. Then the survey redirects to the game page, where users can play the regular and adapted levels, loaded using the *PredefinedLevelGenerator* described in section 4.3. Finally, after playing the two levels, the survey redirects to the last page, where users need to answer questions about their preferences for the levels, and they can also provide feedback about their experience.

As in the previous study, some values needed to be stored in the browser's local storage during the experience to show the same level to the players twice, in regular and adapted versions. These values consist of (i) a global unique identifier (GUID) that distinguishes users, (ii) an identifier fetched from *CountAPI*[1], using *experienceID* as the key for the *pemesteves.itch.io* namespace, allowing for tracking the number of players and assigning an order to each, i.e., even ID numbers played the regular level first, while the odd numbers played the adapted version first, (iii) the current level number users are playing, allowing to distinguish the first and second levels, and (iv) the level itself, allowing to present the same level twice and modify it in the adapted version.

Besides the previous changes to the architecture described in section 4.3, the *Predefined-LevelGenerator* was modified to generate and adapt a level according to the identifier explained above. This class generated the level the first time it was created and stored it in the browser's local storage. If the identifier corresponded to an odd number, the generator deleted one-third of the enemies, added coins to the straight and hill sections if they could be decorated, and increased the number of coins in boxes by changing empty blocks for collectible blocks with a 2/3 chance. With these modifications, the survey was ready to be deployed.

The following subsections explain in detail each section of the survey. For a better view of the questionnaires, the reader should refer to appendix A. The survey's results are presented in section 6.3, but their discussion comes only in chapter 7.

6.1.1 Profile Questions

As mentioned above, the first section of the survey was related to profiling questions and included on the survey's main page. Besides that, it shows the survey's general instructions, i.e., how to answer the questions, how to play the game, and how data was collected and stored during the study. Since the profiling questionnaire is the same presented in section 5.2.1, only with some changes in the survey's description, the reader should refer to this section to know more about it or to appendix A, where the entire questionnaire is shown.

6.1.2 Levels

The second section of the survey consisted of the two levels the users needed to play:

[1] https://api.countapi.xyz/hit/pemesteves.itch.io/experienceID

- The **regular** level, automatically generated by the game's level generator and shown to the user;

- The **adapted** level, created from the previous level, by deleting one-third of the enemies, adding coins to the straight and hill straight sections, if they could be decorated, and increasing the number of coins in boxes by changing empty blocks for collectible blocks with a 2/3 chance.

To ensure the playing order does not affect the A/B testing results, an identifier was fetched from *CountAPI*, as already described above. Players with an odd identifier would play the adapted version first, and people with an even identifier would play the standard level first. To assure both levels were the same, except for the small changes in the number of elements, a JSON version of the level was stored in the browser's local storage.

The level questionnaire represents the primary difference between this and the first study levels' sections. In this study, the questionnaire was not used, leaving the questions related to the level for the end, as described in the following subsection.

6.1.3 Preferences and Comments

The third and final section of the questionnaire was presented to the user after playing the two levels. It contains questions about which level had more of a particular element and which level the player preferred. It also includes a comments box, allowing testers to give feedback about their general experience with the questionnaire.

These questions were the most crucial for answering the second research question raised in section 1.3. The comments also allowed to understand how the generation of levels impacted the users' experience, and some will be highlighted in section 6.3 and discussed in chapter 7.

6.2 Experience Deployment and Data Collection

After the development of the survey, the game, developed using *JavaScript*, was also deployed on an *itch.io* page[2]. This allowed people to run and answer it using only a browser, which is accessible to everyone that has a computer and an Internet connection. Users could also run and answer the survey on mobile devices but could not play the game since it required a keyboard.

As for the other study, the link for *itch.io* was spread around several communities, including *itch.io*, *Facebook*, *Discord*, *Twitter*, and the *University of Porto*. With this dissemination, the survey could reach a good sample size of 178 people, as described in section 6.3, although smaller than the first study's sample, which counted 282 people.

To collect the questionnaires' answers, and gameplay data, *Google Spreadsheets* and *Google App Scripts* were used by deploying a web application capable of receiving and storing *POST* and *GET* requests from the game. The script used was the same as the first study, which is

[2] https://pemesteves.itch.io/game-design-adaptivity-2

generic enough to allow its replication in other investigations. Each request needed to be correctly identified with the target sheet, allowing to separate each level's data, the demographics, and the comments between different spreadsheets. Although the waiting to send the requests produced some delay between levels, it assured data was saved correctly.

6.3 Results

After two weeks of collecting data, the dataset was analyzed. During that time, the questionnaire received 189 answers, where 11 were duplicates, possibly from people experiencing issues during the game. In other words, 178 people contributed to the study, where 142 completed the questionnaire, playing both levels and answering the final questionnaire.

Of these 178, 28 commented on the overall experience, allowing to filter some answers according to these comments. For instance, someone commented, "didn't get too far through either level, so my results may not be accurate," which shows this player's answers would be outliers. Some people explained their preferences, and others just commented on the game. For example, a person complained about the character's movement, stating that "the acceleration is too high." These comments highlight possible issues related to the generator and the game's mechanics. Thus, they will be addressed in chapter 7.

As seen in table 6.1, the adapted level worked as expected, consistently decreasing the enemies' number and increasing the number of collectibles, coins, and powerups, except for one case where the powerups' number was not increased.

Element	Minimum	Maximum	Average	Standard Deviation
Coins	19	77	44.605	14.688
Enemies	-5	-17	-9.111	2.208
Powerups	0	13	4.407	2.519

Table 6.1: Difference in the number of elements between the levels.

During the levels, besides the questionnaires about player preferences, some gameplay metrics and data about elements were collected and used to produce statistics and remove outliers. These properties are (i) the cause of death, which could be -1, 1, 2, 3, if the player did not die, died in a hole, was killed by an enemy, or lost by time, respectively, (ii) the number of update ticks when completing a level or dying, (iii) the time left when completing or failing a level, (iv) the number of coins collected, (v) the number of powerups collected, (vi) the number of enemies killed, and (vii) the array with the player's actions. Due to a data collection error, the player's time left on the clock was not collected. However, this issue was solved with the number of update ticks, which counts the update function calls, with a frame rate of 30, i.e., in each second, there are 30 calls to this function. Furthermore, on level 1, the standard level, its JSON description was collected. On level 2, the adapted level, the new enemies, the straight sections, the straight hill sections, and the collectible's data arrays were collected.

To remove outliers, other methods were also used. For starters, both levels were analyzed according to the ticks data. Experiences with values greater than or equal to 6000 and lesser than the first quartile (368 in level 1, 937.25 in level 2) were deleted since these corresponded to people who died by reaching the time limit or died too soon in a level, respectively. This resulted in a sample of 81 people, which was used for the other analyses.

Regarding the sample's gender, 51 (62.96%) were men, 25 (30.86%) were women, 2 (2.47%) did not answer, and three people (3.7%) did not identify with these genders. The sample's age varied between 15 and 46 years old, with an average of 24 and a standard deviation of 6.348. Regarding the sample's education degree, 38 (46.91%) had completed a bachelor's, 26 (32.1%) high school, 11 (13.58%) a master's, 3 (3.7%) a Ph.D., 2 (2.47%) were still in high school, and the other person (1.23%) did not answer the question. Due to these varying results, it was impossible to draw conclusions concerning demographics. This time, the users' gaming experience was well stored for every user. The survey counted on people more experienced in video games, as seen in table 6.2, possibly because some had already participated in the first study.

Question	Disagree strongly	Disagree a little	Neither agree nor disagree	Agree a little	Agree strongly
Video Games (in general)	1 (1.23%)	2 (2.47%)	5 (6.17%)	22 (27.16%)	62 (62.96%)
Platformer	4 (4.93%)	1 (1.23%)	7 (8.64%)	30 (37.04%)	39 (48.15%)
Super Mario Games	1 (1.23%)	2 (2.47%)	7 (8.64%)	29 (35.80%)	42 (51.85%)
Super Mario Mechanics	3 (3.70%)	5 (6.17%)	8 (9.88%)	24 (29.63%)	41 (50.62%)

Table 6.2: Gaming experience data from the second study population.

Besides the statistics described above, a *RapidMiner* process produced others about the preference for each level, the players' feelings about the difference in the elements' amounts, and the real difference made by the adaptation. Table 6.3 shows the preference and the levels that players felt had more of certain elements. This table shows unexpected results, especially on the gaps questions, since these elements were not changed. Moreover, regarding the adapted elements, i.e., coins, powerups, and enemies, testers seem to know more or less what levels contain more of these elements. However, some could not identify these differences, which can be due to death in different parts of the levels.

Question	Level 1 - Regular	Level 2 - Adapted	None
Level Preference	36 (44.44%)	35 (43.21%)	10 (12.35%)
Feel had more Coins	12 (14.81%)	49 (60.49%)	20 (24.69%)
Feel had more Enemies	69 (85.19%)	7 (8.64%)	5 (6.17%)
Feel had more Gaps	18 (22.22%)	21 (25.93%)	42 (51.85%)
Feel had more Powerups	4 (4.94%)	70 (86.42%)	7 (8.64%)

Table 6.3: Players' preferences and feeling about the elements' amount.

Besides statistics, a correlation matrix for each preference was generated. It can be seen in table 6.3, highlighting correlation coefficients with an absolute value greater than or equal to 0.25. The table shows that there is no significant coefficient. However, it suggests that, for example, people identifying with the female gender tend to prefer the adapted level. In contrast, those identifying with the male gender tend to prefer the standard level.

Attribute	**Preference = 1**	**Preference = 2**	**Preference = none**
Gender = Male	0.22	**-0.26**	0.05
Gender = Female	**-0.27**	**0.39**	-0.17
Gender = NotAnswered	0.18	-0.14	-0.06
Gender = other	-0.04	-0.17	0.32
Education = Bachelor	0.01	0.03	-0.05
Education = HighSchool	-0.03	0.04	-0.02
Education = Master	-0.14	0.02	0.18
Education = PhD	0.09	-0.04	-0.07
Education = SomeHighSchool	0.18	-0.14	-0.06
Education = NotAnswered	0.12	-0.10	-0.04
Age	0.10	-0.19	0.13
Experience-VideoGames	0.17	-0.30	0.19
Experience-Platformer	0.20	**-0.32**	0.18
Experience-SuperMario	0.08	-0.12	0.07
Experience-Mechanics	0.16	**-0.28**	0.18
Extraversion	0.05	-0.14	0.15
Agreeableness	-0.07	**0.26**	**-0.29**
Conscientiousness	-0.10	**0.25**	-0.22
Neuroticism	0.11	-0.11	0.00
Openness	0.17	- 0.13	-0.06

Table 6.4: Correlation matrix of player preferences.

As in the first study, a decision tree was created to understand if a similar model could be used as a recommendation system and if there was any data cluster from which conclusions could be drawn. Then a cross-validation strategy was applied to get the model's prediction accuracy. The tree to evaluate the players' preferences is illustrated in figure 6.1, showing some intriguing clusters and including personality traits to differentiate between them. Although the tree isolated several groups and got most of these predictions right, it only reached an accuracy of 53.19% in the cross-validation test.

With this analysis of the second study's results, some conclusions could be drawn to answer the second research question raised in section 1.3: Does automatically generating levels and adapting them to the overall preferences of players impact their experience?

As can be concluded by looking at table 6.3, the adaptation did not improve or worsen the overall players' experience. In other words, adapting levels did not impact players' overall experience.

However, some players reported they preferred the adapted level since it was more accessible. Still, others said the opposite, liking the standard level since the other was too easy. Some of these

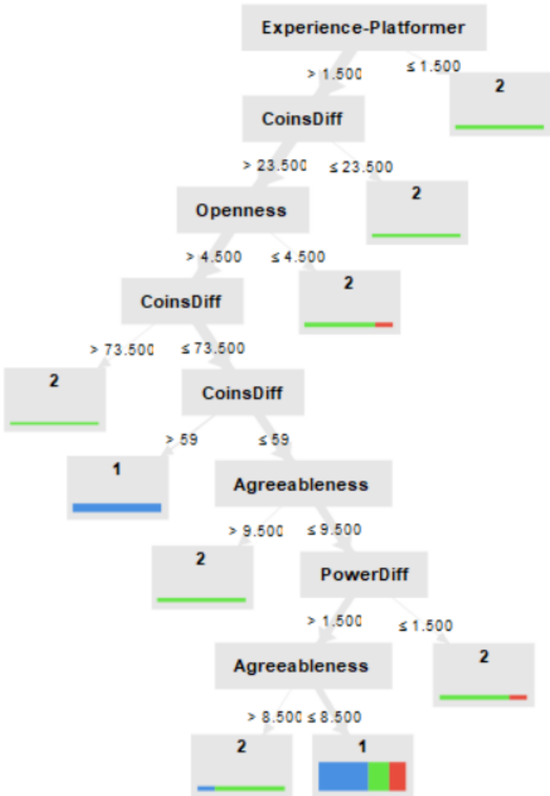

Figure 6.1: Players' preferences decision tree (Blue = Regular Level; Green = Adapted Level; Red = None).

results may be due to the game's level generator, mechanics, or the players' profile and gaming habits. Therefore, as stated at the beginning of this chapter, chapter 7 outlines the study's results by discussing possible reasons for the unexpected results and other ways to analyze the data collected in future studies.

Chapter 7

Discussion

During the previous two chapters, chapter 5 and chapter 6, several results were presented, answering the two research questions raised in section 1.3.

Since some results were unexpected, this chapter presents possible reasons for it, targeting both studies and the selected game. The following sections discuss these results, considering the comments provided by players in the two surveys conducted. At the end of the chapter, a summary is presented, concluding the discussion and setting the way for future work.

7.1 Infinite Mario Bros

For both studies, a *JavaScript* (JS) port of *Infinite Mario Bros* [45] was chosen to create surveys with pages where users could play the game. Since the game was not developed solely to reach this work's objectives, several issues were spotted by players while answering the survey. It should be noted that the generator was only changed to allow storing and loading levels, and the chances of creating each section or element remained the same. Some people noticed the use of this generator. "Both maps looked like they came from the Mario level generator. They were both very bland and effectively empty," commented a tester in the second study. This means that even the generator had problems that could have affected the players' experience.

Regarding JS, some browsers, such as *Mozilla Firefox*, do not run at the specified frame rate (30 frames per second). The low performance in these browsers can affect the players' experience, especially since it is a popular browser nowadays, and 37 out of 189 testers, i.e., 19.6% of the population, used it in the second prototype. Some users even commented on this lack of performance, stating that "the game runs really poorly," and asking to "increase the framerate."

Besides the generation and frame rate, several people complained about the game's mechanics, mainly Mario's acceleration. A person stated that "the mechanics were slightly different than the original [game] - speed and acceleration were much higher" (translated from Portuguese), and another said "it felt like an old mario sped up a bit, like 1.2x the speed." This last person also told "the gravity is slightly too strong to make precise jumps." These last two comments are related to the feedback provided by other two players: "The gaps can be annoying, because sometimes the own game doesn't register your input, which is more important if it's the jump input," and "Took too long to stop walking." Besides the movement mechanics, some people spotted some "missing [...] mechanics, like picking up shells and being able to throw them," which in the opinion of another tester "meant enemy variety barely mattered."

However, not every mechanic worsens the players' experience. A tester pointed out that the "wall sliding and jumping were a nice touch," which, according to a Mario wiki[1], was only present in the original game as a glitch and not a real mechanic. However, another player said he felt the instructions "should have told us we [the players] have a wall jump," which means that the player only discovered the mechanic too late in the survey, and it probably could saved the user in a level before.

Last but not least, many people complained about the game's controls, which were "not much intuitive at first." Some testers also claimed they died because they "forgot that jumping was with the S" and "it is very hard to use letters and arrows at the same time." Moreover, players gave some hints about what should be done instead: the game "should let the player choose or arrows or letters for the buttons," or "introduce the space key as an option for jumping."

All these issues could be reviewed before the experimentations. However, the focus of this work was to understand how the game's elements impacted the players' experience and if this impact could be correlated to the players' profiles. Thus, although some mechanics might influence the overall experience, the questionnaire was made to ignore their existence and focus on the game elements. Some people may have given wrong answers just because of these mechanics, so a more detailed analysis of the studies' data should be conducted later by eliminating more outliers and expanding it to the machine learning area.

7.2 First Study

Focusing on the first study, some comments can be highlighted. The most common were related to the lives given in each level. Several people complained about the only life given, saying that if they died, they could not evaluate the level and asking for an option to restart the levels. Another tester gave an interesting suggestion: "the first game (or first try) should be a test, only to test the time of response of the game." That could be done with a button redirecting to a new page where players could try a level before starting the survey. With this, probably some people would be more acquainted with the game's mechanics and would not die instantly.

Regarding the questionnaire, some users gave some feedback about how they felt they should be answering it. For instance, a person said that the questions would be intuitive on the gaming experience's page if they were yes or no questions. Another person said the level's questionnaire should "ask for if there are too little or too many of each element instead of a general feeling of satisfaction." These ideas were thought through for both questionnaires before the survey's

[1] https://www.mariowiki.com/Wall_Jump

deployment. For the first, yes or no questions cannot distinguish a person who plays games daily from a person who plays once a month. Moreover, stating if the elements were too little or too many does not say if people liked them. For instance, someone could have liked the elements, even saying they were too many because they helped pass the level. This question could also be added instead of substituting the satisfaction question, but the questionnaire would be too big to retain players for the whole survey.

Furthermore, people also commented on the levels' elements. Some just gave general feedback about their preferences, for example, "levels with platforms that didn't even need to be reachable felt a lot fuller and more fun to play," and "all the levels without enemies were very boring and not enjoyable." From the set of comments related to the elements, a person said that "some lines of blocks with enemies underneath felt too narrow to be worth going under," and the "frequency of gaps and obstacles around them affect difficulty more than the width." This is an intriguing point of view, and although the element's evaluation did not consider these cases, it was discussed before the study. However, the work's objectives were to find correlations with single elements; thus, this analysis can and should be made in future work.

Moreover, some people commented on how well they performed and how they answered the questionnaire, some with mistakes. These comments were not used to spot outliers since others could make the same mistakes without noticing and telling it in the comments section. Others said what their satisfaction meant in terms of the elements' amount. Therefore, the data analysis can have some outliers that any method should remove, and then other analyses should be made.

Despite these adverse comments, there were people enjoying the survey, stating they "liked to answer this survey in this way" (translated from Portuguese), and even it was "the best survey ever." This may be because the survey incorporated a game without forcing people to download and play it on their computers. Moreover, the survey was not too dense, having only a few questions for each level.

To conclude, as stated in section 5.4, it was not possible to correlate players' profiles with their preferences for specific game elements, answering the first research question raised in section 1.3. However, as seen above, groups of elements should be considered, and a question for future research can be raised here: Do players' profiles correlate with their preference for groups of game elements? Nevertheless, this work continued with the same game and a similar survey but relied on the players' overall preferences for game elements since, overall, the testers liked it.

7.3 Second Study

Revisiting the second study, the most common comments were related to the lives given in each level, as in the previous study. Of the 28 people who commented on this study's survey, 16 complained about the only life given, saying they lost too early on one of the levels or even in both. Although this also happened in the previous survey, the idea was that the players would play as usual instead of trying to master the mechanics before they started the survey. However, as highlighted in the previous section, a new page where players could try a level before starting the survey could be a helpful idea to prevent this kind of mistake. Besides, a person commented that the game should give three lives for each level, allowing players to experience the whole level even losing once. However, supposing this was done in the first survey, most players who completed it could give up early, not completing the twelve levels due to the survey's length. Moreover, since the second survey followed the same structure as the previous, this detail remained unchanged. These answers were considered outliers during the level's results analysis, as described in section 6.3. Nevertheless, a survey with three lives should also be conducted to conclude if this issue is relevant for evaluating the experience.

Beyond the lives issue, seven testers commented on their preferences, showing a preference for the standard level. Some people felt the adapted level was "a little poorly designed," or "non-challenging, and [...] less enjoyable." Others said the standard level was "more challenging due to having more enemies and, consequently, more fun." However, two people said both levels "very bland and effectively empty," and players "could just run through and ignore." Since each player experienced a different level this time, this could be an issue related to these players' levels, or the adapted level could not fit the players' skills.

To conclude, as stated in section 6.3, it was possible to answer the second research question raised in section 1.3: generating and adapting levels to the overall players' preferences does not impact their experience. However, as seen above, some people disliked the level for being simpler and having lesser challenges. Further analyses should be made with these people by applying different methods, eliminating more outliers, and expanding them to the machine learning area.

7.4 Summary

In this section, the comments from both surveys conducted and described in chapters 5 and 6 were overviewed.

By reviewing these comments, some issues related to the chosen game and both studies stand out. For starters, the game should include the classical mechanics of *Super Mario Bros.* [1], such as picking up and throwing the enemies' shells. The character's movement speed and acceleration should also be reviewed.

Another way around this issue would be to select an unknown game to avoid failed expectations from the players. Moreover, the first study's survey should be reviewed and, in later studies, include questions about the quantity of the elements evaluated.

Nevertheless, both research questions, raised in section 1.3, could be answered during the work:

- Do players' profiles correlate with their preference for specific game elements?
  No, these properties are not correlated.

- Does automatically generating levels and adapting them to the overall preferences of players impact their experience?

It does not impact their experience, i.e., it does not improve or worsen the players' experience.

During these sections, future work was also proposed. The chosen game could be changed to make it more similar to what the players expect, or another game could be selected, and the same methodology could be employed. Other analyses could be made to the datasets produced in these experimentations, especially those related to machine learning. Lastly, instead of correlating specific game elements with players' profiles, a deeper analysis should consider groups of elements, especially obstacles around gaps.

Chapter 8

Conclusions

Throughout this work, the field of Game Adaptivity was addressed through the platform levels' generation problem. Solving this problem could help game designers in the games' development process, enabling faster development and possibly better results.

To propose a solution to this problem, works related to Game Adaptivity, Level Design, and Player Modeling and Profiling were reviewed. This review established a lack of research in platform games adaptivity using players' personalities as input to the players' models. Moreover, there is much research around platform games, emphasizing *Infinite Mario Bros* [45], providing level generators that could be modified to fit the present thesis' needs. These works also highlight elements that can be adapted or generated and ways to do it, and there is a possibility of adding more depending on the game genre and the level itself.

Upon reviewing related works, a methodology was designed to answer the previously stated research questions. This methodology is sufficiently generic to be extrapolated to other platform games and genres. It is divided into three main stages: (i) game selection, where the game genre and the game itself are chosen; (ii) first study, where the elements are selected, and the game is modified, allowing to conduct a study to correlate the elements with players' profiles, preferences, or actions; (iii) second study, based on the previous, where an adaptive version is compared to the baseline version, measuring the impact of the generation and adaptation of levels.

With this methodology, a solution was created based on a *JavaScript* port of *Infinite Mario Bros* [45]. The game's architecture and the changes made to it were thoroughly analyzed to understand how the game works and how to conduct studies correlating game elements with players' profiles, preferences, or actions and measuring the impact of generating or adapting levels.

After the game selection, a study was conducted to correlate game content with players' profiles, answering the first research question. A survey was developed by integrating *SurveyJS* with the game engine, resulting in a prototype launched at *itch.io*[1] that could be played on browsers. The survey counted 282 responses, of which 115 completed every level. Although the survey had

[1] https://pemesteves.itch.io/game-design-adaptivity

a large sample, it was impossible to find correlations between single game elements and players' profiles, thus answering the first research question with an unexpected result. Nevertheless, the survey's responses showed intriguing results when comparing levels with the same testing purpose, i.e., levels where the same element was evaluated. With these results, it could be concluded that overall the players' experience improved when decreasing the number of enemies, without eliminating them all, and increasing the number of collectibles. So, the methodology could be followed by targeting these properties in a second study.

With the previous results, a second study was conducted to improve players' experience through offline adaptivity. The survey was done along the lines of the previous, being launched at *itch.io*[2] and counting with 178 responses, where 142 completed both levels and answered about their preference. However, its results show it did not improve or worsen the overall players' experience, thus answering the second research question with a negative answer, i.e., the adaptation did not impact the players' experience.

After both studies were presented, possibilities that led to those unexpected results were discussed by going through the players' comments on the overall experience. Several issues were spotted with this discussion, mainly on the selected game. Since several users were acquainted with the game and its mechanics, some missed a few mechanics from the original game and noticed differences in the character's movement. Other players thought some questions might be asked differently. These details could improve in future work, as described in the following section.

8.1 Future Work

As seen above, during the discussion, several issues were spotted. In the future, these issues should be tackled, and a new iteration with the same studies could be conducted to ensure the data was adequately collected and matches the players' feelings and beliefs. Another option is to select another similar game and conduct the same experiments to determine if the game influences the collected data.

Moreover, these studies' results could be analyzed more in-depth by applying other algorithms and machine learning. With these approaches, possibly other clusters could be found, and a different result could be achieved.

Last but not least, as a person commented on one of the surveys, the "frequency of gaps and obstacles around them affect difficulty more than the width," i.e., the players' experience can be more influenced by groups than single elements. These connections between elements can and should be explored in future work, possibly producing different results than analyzing single elements.

[2] https://pemesteves.itch.io/game-design-adaptivity-2

References

[1] Nintendo Research & Development 4. Super mario bros., 1985.

[2] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17:734–749, 6 2005.

[3] Michael C Ashton and Kibeom Lee. Empirical, theoretical, and practical advantages of the hexaco model of personality structure. *Personality and Social Psychology Review*, 11:150–166, 2007.

[4] Sander C.J. Bakkes, Pieter H.M. Spronck, and Giel van Lankveld. Player behavioural modelling for video games. *Entertainment Computing*, 3:71–79, 2012.

[5] Richard Bartle. Hearts, clubs, diamonds, spades: Players who suit muds. *Journal of MUD research*, 1:19, 1996.

[6] Emin Alp Biyik and Elif Sürer. Developing a space syntax-based evaluation method for procedurally generated game levels. *Mugla Journal of Science and Technology*, 12 2020.

[7] Paris Mavromoustakos Blom, Sander Bakkes, Chek Tien Tan, Shimon Whiteson, Diederik Roijers, Roberto Valenti, and Theo Gevers. Towards personalised gaming via facial expression recognition. In *AIIDE'14: Proceedings of the Tenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, pages 30–36, 2014.

[8] Elizabeth Camilleri, Georgios N. Yannakakis, and Alexiei Dingli. Platformer level design for player believability. In *IEEE Conference on Computatonal Intelligence and Games, CIG*, volume 0. IEEE Computer Society, 7 2016.

[9] M Carrozzino, F Tecchia, and M Bergamasco. Urban procedural modeling for real-time rendering. In *Proceedings of the 3rd ISPRS International Workshop 3D-ARCH*, 2 2009.

[10] Edirlei Soares de Lima, Bruno Feijó, and Antonio L. Furtado. Player behavior and personality modeling for interactive storytelling in games. *Entertainment Computing*, 28:32–48, 12 2018.

[11] A.I. Design. Rogue, 1980.

[12] Sahraoui Dhelim, Nyothiri Aung, Mohammed Amine Bouras, Huansheng Ning, and Erik Cambria. A survey on personality-aware recommendation systems. *Artificial Intelligence Review*, 2021.

[13] João Augusto dos Santos Lima. Adaptivity in single player video games. Master's thesis, Faculdade de Engenharia da Universidade do Porto, 2021.

[14] Florian Himsl Edmund McMillen. The binding of isaac, 2011.

[15] Pedro Esteves. Players' profiles and preferences in standard and adapted levels: Dataset. https://doi.org/10.5281/zenodo.6677476, 2022.

[16] Pedro Esteves. Players' profiles and satisfaction for game elements across levels: Dataset. https://doi.org/10.5281/zenodo.6623375, 2022.

[17] Henry D. Fernández, Mikami Koji, and Kunio Kondo. Adaptable game experience based on player's performance and eeg. In *Proceedings - 2017 NICOGRAPH International, NICOInt 2017*, pages 1–8. Institute of Electrical and Electronics Engineers Inc., 9 2017.

[18] Matilde Freilão. Affective narratives for engagement in digital games. Master's thesis, Faculdade de Engenharia da Universidade do Porto, 2020.

[19] Riccardo Galdieri, Alessandro Longobardi, Michele De Bonis, and Marcello Carrozzino. Users' evaluation of procedurally generated game levels. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 12980 LNCS, pages 44–52. Springer Science and Business Media Deutschland GmbH, 2021.

[20] Hello Games. No man's sky, 2016.

[21] Erik Geslin, Laurent Jégou, and Danny Beaudoin. How color properties can be used to elicit emotions in video games. *International Journal of Computer Games Technology*, 2016, 2016.

[22] Samuel D. Gosling, Peter J. Rentfrow, and William B. Swann. A very brief measure of the big-five personality domains. *Journal of Research in Personality*, 37:504–528, 2003.

[23] Michael Cerny Green, Luvneesh Mugrai, Ahmed Khalifa, and Julian Togelius. Mario level generation from mechanics using scene stitching. In *2020 IEEE Conference on Games (CoG)*, pages 49–56. IEEE, 8 2020.

[24] Mark Hendrikx, Sebastiaan Meijer, Joeri Van Der Velden, and Alexandru Iosup. Procedural content generation for games: A survey. *ACM Transactions on Multimedia Computing, Communications and Applications*, 9, 2 2013.

[25] Danial Hooshyar, Moslem Yousefi, and Heuiseok Lim. Data-driven approaches to game player modeling: A systematic literature review. *ACM Computing Surveys*, 50, 1 2018.

[26] INC IDV. Speedtree – 3d vegetation modeling and middleware. `https://store.speedtree.com/`, 2020.

[27] João Jacob, Ana Lopes, Rui Nóbrega, Rui Rodrigues, and António Coelho. Player adaptivity and safety in location-based games. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10714 LNCS, pages 219–238. Springer Verlag, 2018.

[28] João Jacob, Ana Lopes, Rui Nóbrega, Rui Rodrigues, and António Coelho. Towards player adaptivity in mobile exergames. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10714 LNCS, pages 278–292. Springer Verlag, 2018.

[29] Oliver P. John and Sanjay Srivastava. The big five trait taxonomy: History, measurement, and theoretical perspectives. In *Handbook of personality: Theory and research*, pages 102–138. Guilford Press, 1999.

[30] Stephen Karpinskyj, Fabio Zambetta, and Lawrence Cavedon. Video game personalisation techniques: A comprehensive survey. *Entertainment Computing*, 5:211–218, 9 2014.

[31] Ahmed Khalifa, Michael Cerny Green, Gabriella Barros modlai Maceió, Julian Togelius, and Gabriella Barros. Intentional computational level design. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 2019.

[32] Raph Koster. *Theory of Fun for Game Design*. O'Reilly Media, Inc., 2nd edition, 2013.

[33] Giel Van Lankveld, Pieter Spronck, Jaap Van Den Herik, and Arnoud Arntz. Games as personality profiling tools. In *2011 IEEE Conference on Computational Intelligence and Games, CIG 2011*, pages 197–202, 2011.

[34] Cynthia LeRouge, Jiao Ma, Sweta Sneha, and Kristin Tolle. User profiles and personas in the design and development of consumer health technologies. *International Journal of Medical Informatics*, 82:e251–e268, 11 2013.

[35] Antonios Liapis, Georgios N. Yannakakis, Mark J. Nelson, Mike Preuss, and Rafael Bidarra. Orchestrating game generation. *IEEE Transactions on Games*, 11:48–68, 3 2019.

[36] Antonios Liapis, Georgios N Yannakakis, and Julian Togelius. Computational game creativity. In *5th International Conference on Computational Creativity (ICCC) 2014*, 2014.

[37] Ricardo Lopes and Rafael Bidarra. Adaptivity challenges in games and simulations: a survey. *IEEE Transactions on Computational Intelligence and AI in Games*, 3:85–99, 2011.

[38] S. Miyamoto, T. Nakago, and T. Tezuka. The legend of zelda, 1986.

[39] Mossmouth. Spelunky, 2008.

[40] Lennart E. Nacke, Chris Bateman, and Regan L. Mandryk. Brainhex: Preliminary results from a neurobiological gamer typology survey. In Junia Coutinho Anacleto, Sidney Fels, Nicholas Graham, Bill Kapralos, Magy Saif El-Nasr, and Kevin Stanley, editors, *Entertainment Computing – ICEC 2011*, pages 288–293, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[41] Aniket Nagle, Peter Wolf, and Robert Riener. Towards a system of customized video game mechanics based on player personality: Relating the big five personality traits with difficulty adaptation in a first-person shooter game. *Entertainment Computing*, 13:10–24, 3 2016.

[42] Pedro A. Nogueira, Vasco Torres, Rui Rodrigues, Eugénio Oliveira, and Lennart E. Nacke. Vanishing scares: biofeedback modulation of affective player experiences in a procedural horror game. *Journal on Multimodal User Interfaces*, 10:31–62, 3 2016.

[43] Blizzard North. Diablo, 1997.

[44] Christopher Pedersen, Julian Togelius, and Georgios N. Yannakakis. Modeling player experience for content creation. *IEEE Transactions on Computational Intelligence and AI in Games*, 2:54–67, 3 2010.

[45] Markus Persson. Infinite mario bros, 2006.

[46] Beatrice Rammstedt and Oliver P. John. Measuring personality in one minute or less: A 10-item short version of the big five inventory in english and german. *Journal of Research in Personality*, 41:203–212, 2 2007.

[47] Carlos Pereira Santos, Kevin Hutchinson, Vassilis Javed Khan, and Panos Markopoulos. Profiling personality traits with games. *ACM Transactions on Interactive Intelligent Systems*, 9, 3 2019.

[48] Anurag Sarkar and Seth Cooper. Towards game design via creative machine learning (gdcml). In *2020 IEEE Conference on Games (CoG)*, pages 744–751. IEEE, 8 2020.

[49] Jesse Schell and Jesse Schell. *The Art of Game Design*. CRC Press, 8 2008.

[50] Noor Shaker, Georgios N. Yannakakis, and Julian Togelius. Feature analysis for modeling game content quality. In *2011 IEEE Conference on Computational Intelligence and Games, CIG 2011*, pages 126–133, 2011.

[51] Noor Shaker, Georgios N. Yannakakis, and Julian Togelius. Crowdsourcing the aesthetics of platform games. *IEEE Transactions on Computational Intelligence and AI in Games*, 5:276–290, 2013.

[52] Noor Shaker, Georgios N Yannakakis, Julian Togelius, Miguel Nicolau, and Michael O'neill. Evolving personalized content for super mario bros using grammatical evolution. In *Eighth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2012.

[53] Tianye Shu, Jialin Liu, and Georgios N. Yannakakis. Experience-driven pcg via reinforcement learning: A super mario bros study. In *2021 IEEE Conference on Games (CoG)*, pages 1–9, 6 2021.

[54] Gillian Smith, Jim Whitehead, and Michael Mateas. Tanagra: A mixed-initiative level design tool. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games - FDG '10*, pages 209–216. ACM Press, 2010.

[55] Gillian Smith, Jim Whitehead, Michael Mateas, Mike Treanor, Jameka March, and Mee Cha. Launchpad: A rhythm-based level generator for 2-d platformers. *IEEE Transactions on Computational Intelligence and AI in Games*, 3:1–16, 3 2011.

[56] Nathan Sorenson, Philippe Pasquier, and Steve DiPaola. A generic approach to challenge modeling for the procedural creation of video game levels. In *IEEE Transactions on Computational Intelligence and AI in Games*, volume 3, pages 229–244, 9 2011.

[57] Katta Spiel, Sven Bertel, and Fares Kayali. Adapting gameplay to eye movements – an exploration with tetris. In *CHI PLAY 2019 - Extended Abstracts of the Annual Symposium on Computer-Human Interaction in Play*, pages 687–695. Association for Computing Machinery, Inc, 10 2019.

[58] Adam Summerville, Sam Snodgrass, Matthew Guzdial, Christoffer Holmgård, Amy K. Hoover, Aaron Isaksen, Andy Nealen, and Julian Togelius. Procedural content generation via machine learning (pcgml). *IEEE Transactions on Games*, 10:257–270, 9 2018.

[59] Mike Treanor, Bryan Blackford, Michael Mateas, and Ian Bogost. Game-o-matic: Generating videogames that represent ideas. In *Proceedings of the The third workshop on Procedural Content Generation in Games - PCG'12*, pages 1–8. ACM Press, 2012.

[60] Yi Xu, Roger Smeets, and Rafael Bidarra. Procedural generation of problems for elementary math education. *International Journal of Serious Games*, 8:49–66, 6 2021.

[61] Georgios N Yannakakis, Pieter Spronck, Daniele Loiacono, and Elisabeth Andre. Player modeling. *Artificial and Computational Intelligence in Games*, 2013.

[62] Georgios N. Yannakakis and Julian Togelius. Experience-driven procedural content generation. *IEEE Transactions on Affective Computing*, 2:147–161, 7 2011.

[63] Mohammad Zohaib. Dynamic difficulty adjustment (dda) in computer games: A review. *Advances in Human-Computer Interaction*, 2018:1–12, 11 2018.

Appendix A

Questionnaires

A.1 Demographics

1. What gender do you identify as?

   ◯ Male

   ◯ Female

   ◯ Prefer not to answer

   ◯ Other (describe)

2. What is your age? ___

3. What is the highest degree or level of education you have completed?

   ◯ Some High School

   ◯ High School

   ◯ Bachelor's Degree

   ◯ Master's Degree

   ◯ Ph.D. or higher

   ◯ Trade School

   ◯ Prefer not to answer

A.2 Personality

4. I see myself as someone who...

	Disagree Strongly	Disagree a little	Neither agree nor disagree	Agree a little	Agree strongly
...is reserved.	○	○	○	○	○
...is generally trusting.	○	○	○	○	○
...tends to be lazy.	○	○	○	○	○
...is relaxed, handles stress well.	○	○	○	○	○
...has few artistic interests.	○	○	○	○	○
...is outgoing, sociable.	○	○	○	○	○
...tends to find fault with others.	○	○	○	○	○
...does a thorough job.	○	○	○	○	○
...gets nervous easily.	○	○	○	○	○
...has an active imagination.	○	○	○	○	○
...is considerate and kind to almost everyone.	○	○	○	○	○

A.3 Gaming Experience

5. Please answer every question.

	Disagree Strongly	Disagree a little	Neither agree nor disagree	Agree a little	Agree strongly
I have experience with video games (in general).	○	○	○	○	○
I have experience with the platform game genre.	○	○	○	○	○
I have experience playing Super Mario Bros.	○	○	○	○	○
I am acquainted with Super Mario Bros. mechanics.	○	○	○	○	○

A.4 First Study - Level Questionnaire

How to answer this questionnaire?

As mentioned before, you only need to say if the elements improved or not the overall experience in the level by stating your **level of satisfaction**. If you did not see the said element, either because you lost before seeing it or won without seeing it, please state that **it does not apply to the previous level**.

Please answer as honestly as possible.

1. In the previous level, how did you feel about...

	Not applicable to the previous level	Very dissatisfied	Dissatisfied	Neutral	Satisfied	Very satisfied
...the number of enemies?	○	○	○	○	○	○
...the width of gaps?	○	○	○	○	○	○
...the variety of enemies' types?	○	○	○	○	○	○
...the number of coins?	○	○	○	○	○	○
...the number of powerups?	○	○	○	○	○	○
...the overall experience?	○	○	○	○	○	○

A.5 Second Study - Overall Experience

How to answer this questionnaire?

In the following questionnaire, you must state which level you feel had **MORE** of each element. If you think that both levels had the same amount of the element, you should answer with **NONE**.

Then, you need to state your preference between the two levels, and if you do not have any, you should answer **NONE**.

Finally, you can leave some comments regarding the whole experience and some justification for your answers if you want.

Please answer as honestly as possible.

1. Which level did you feel had MORE...

	Level 1	Level 2	None
...coins?	○	○	○
...powerups?	○	○	○
...enemies?	○	○	○
...gaps?	○	○	○

2. Which level did you prefer?

   ◯ Level 1

   ◯ Level 2

   ◯ None

3. Please leave any comment if you want...

Appendix B

First Study - Decision Trees, Correlation Matrices, and Charts

B.1 Levels' Decision Trees

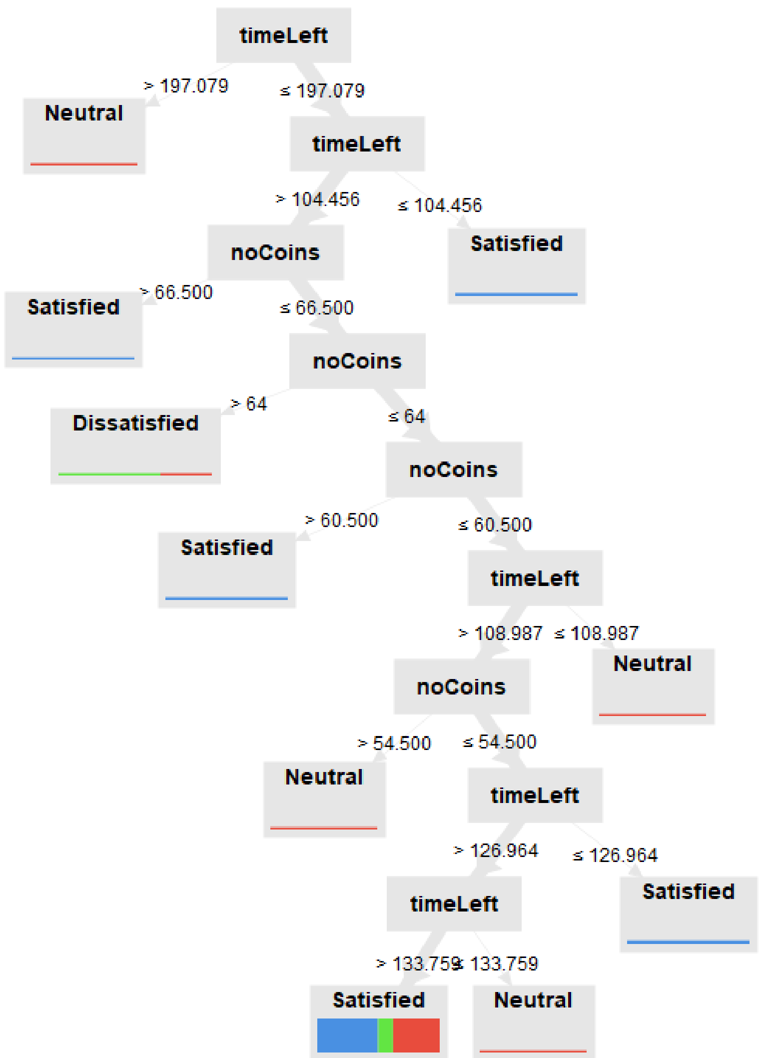

Figure B.1: Level 1 - Decision Tree.

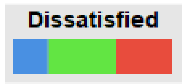

Figure B.2: Level 2 - Decision Tree.

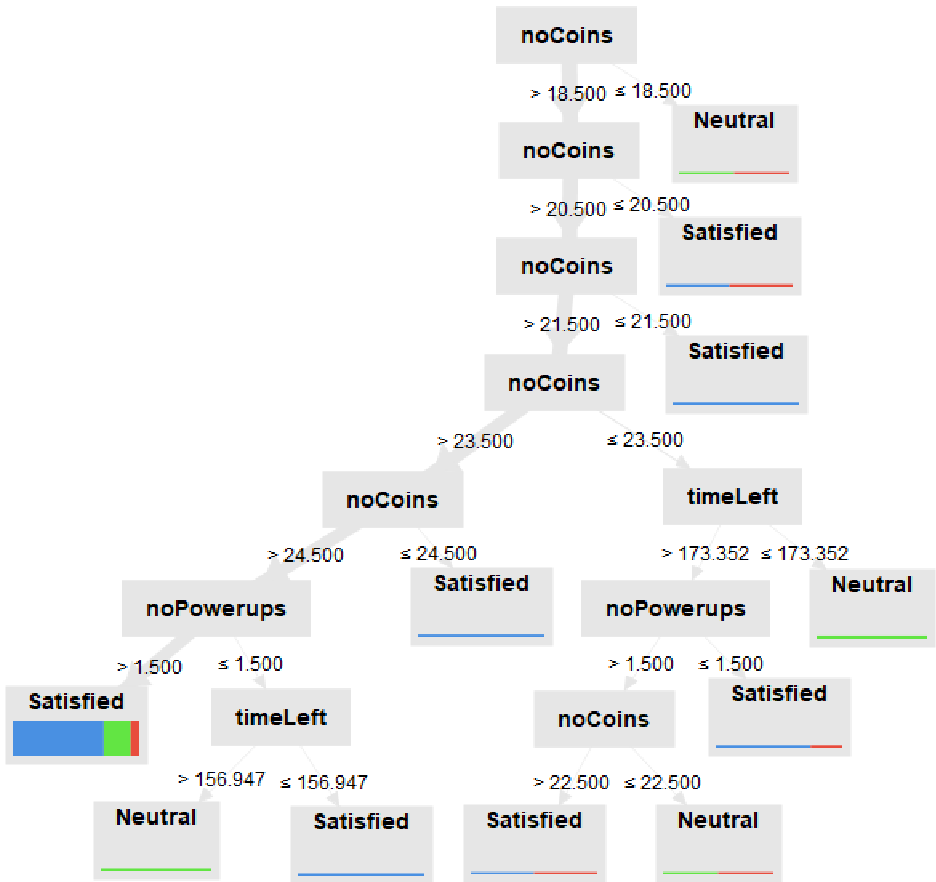

Figure B.3: Level 3 - Decision Tree.

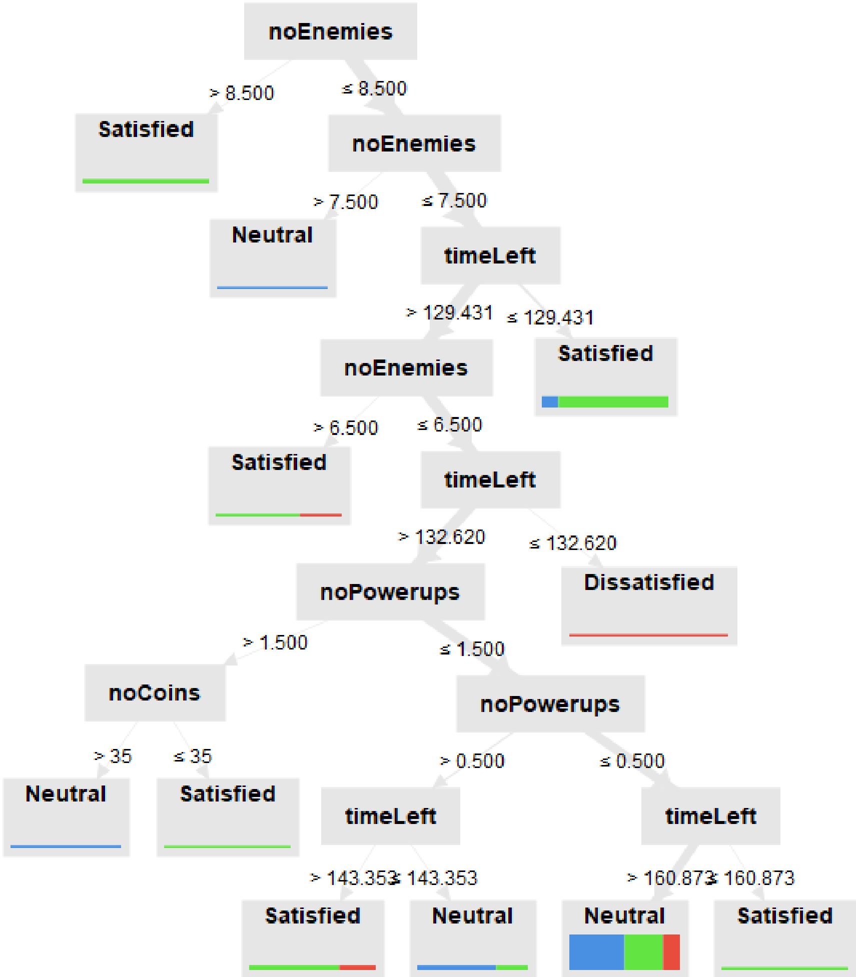

Figure B.4: Level 4 - Decision Tree.

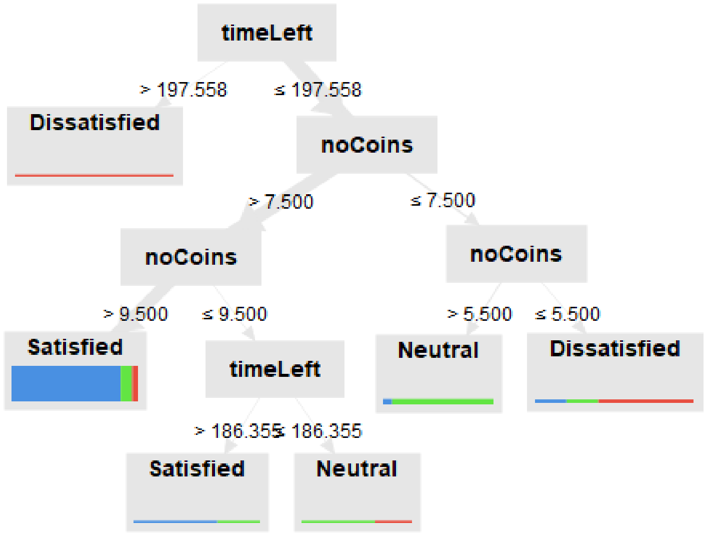

Figure B.5: Level 5 - Decision Tree.

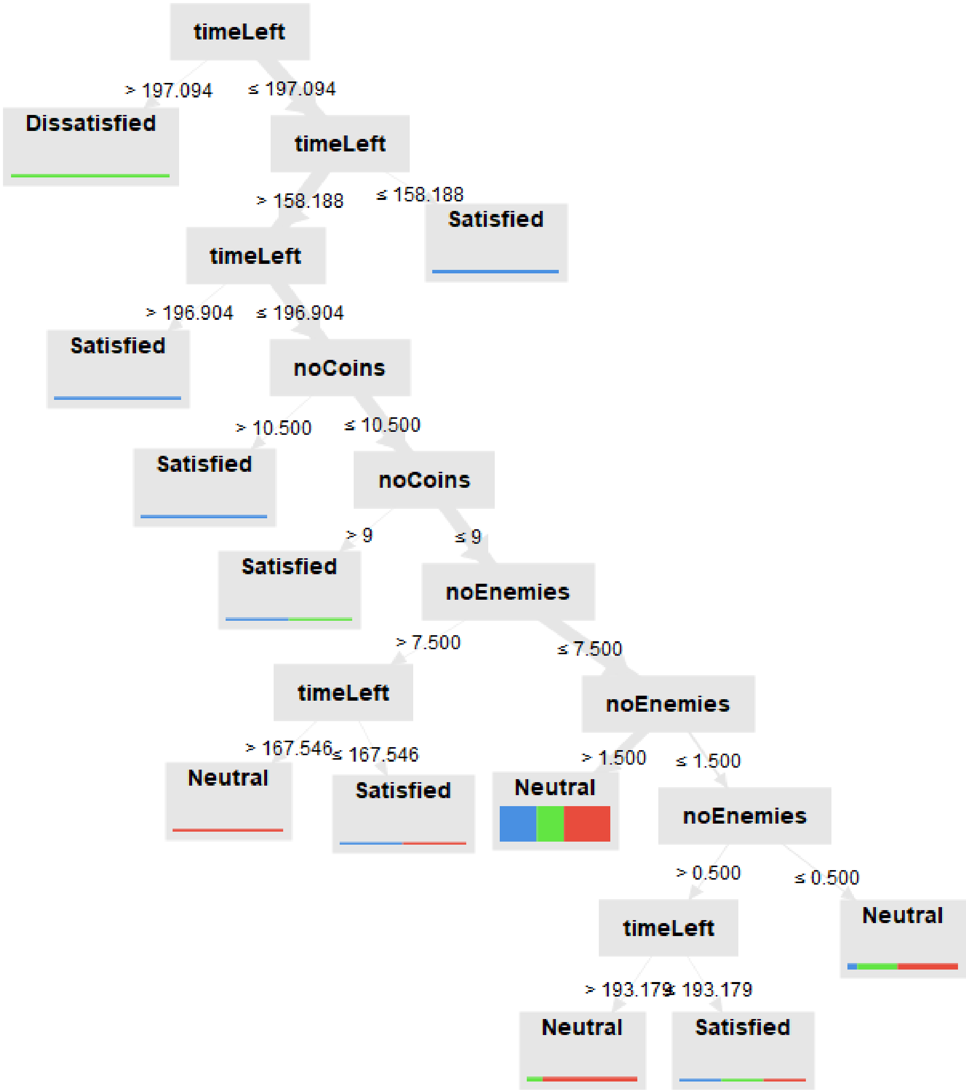

Figure B.6: Level 6 - Decision Tree.

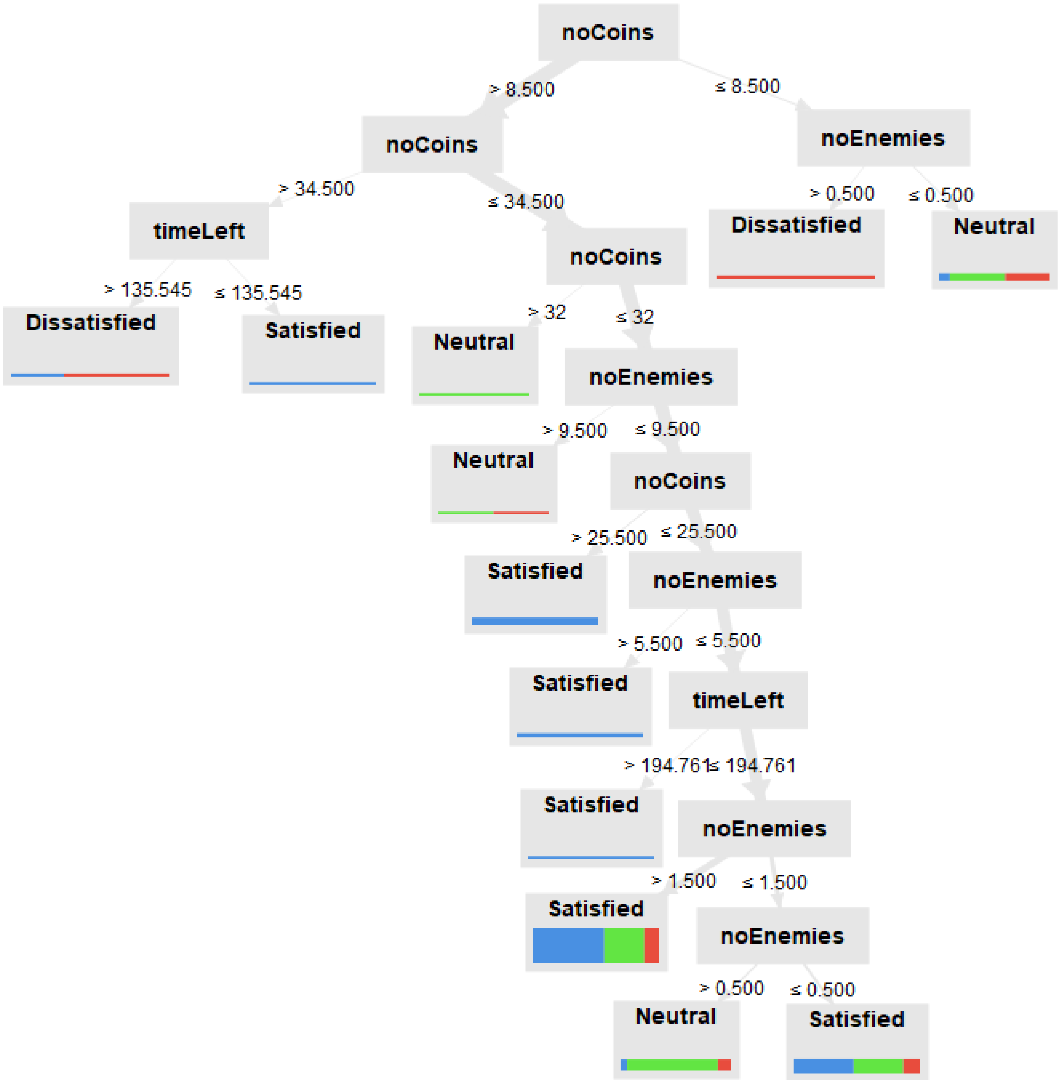

Figure B.7: Level 7 - Decision Tree.

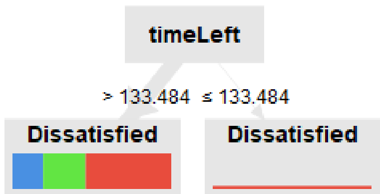

Figure B.8: Level 8 - Decision Tree.

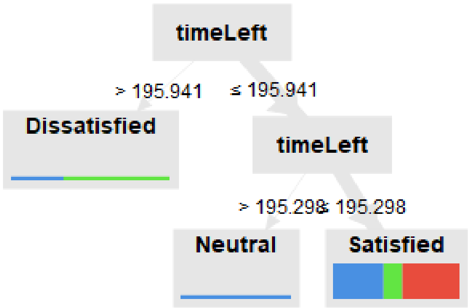

Figure B.9: Level 9 - Decision Tree.

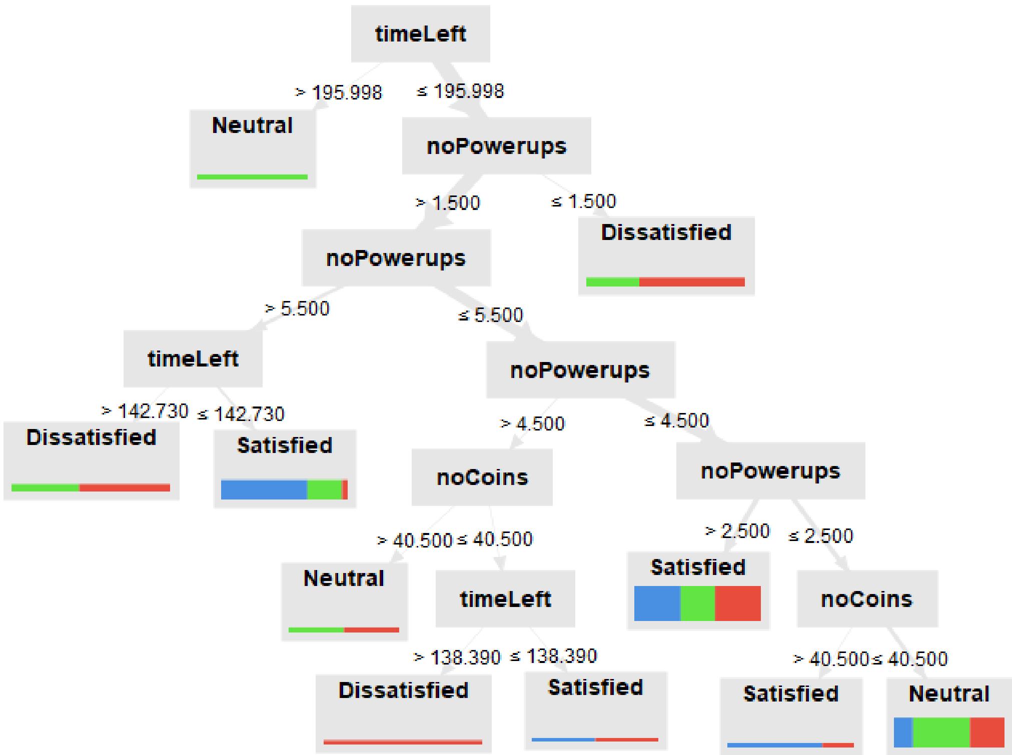

Figure B.10: Level 10 - Decision Tree.

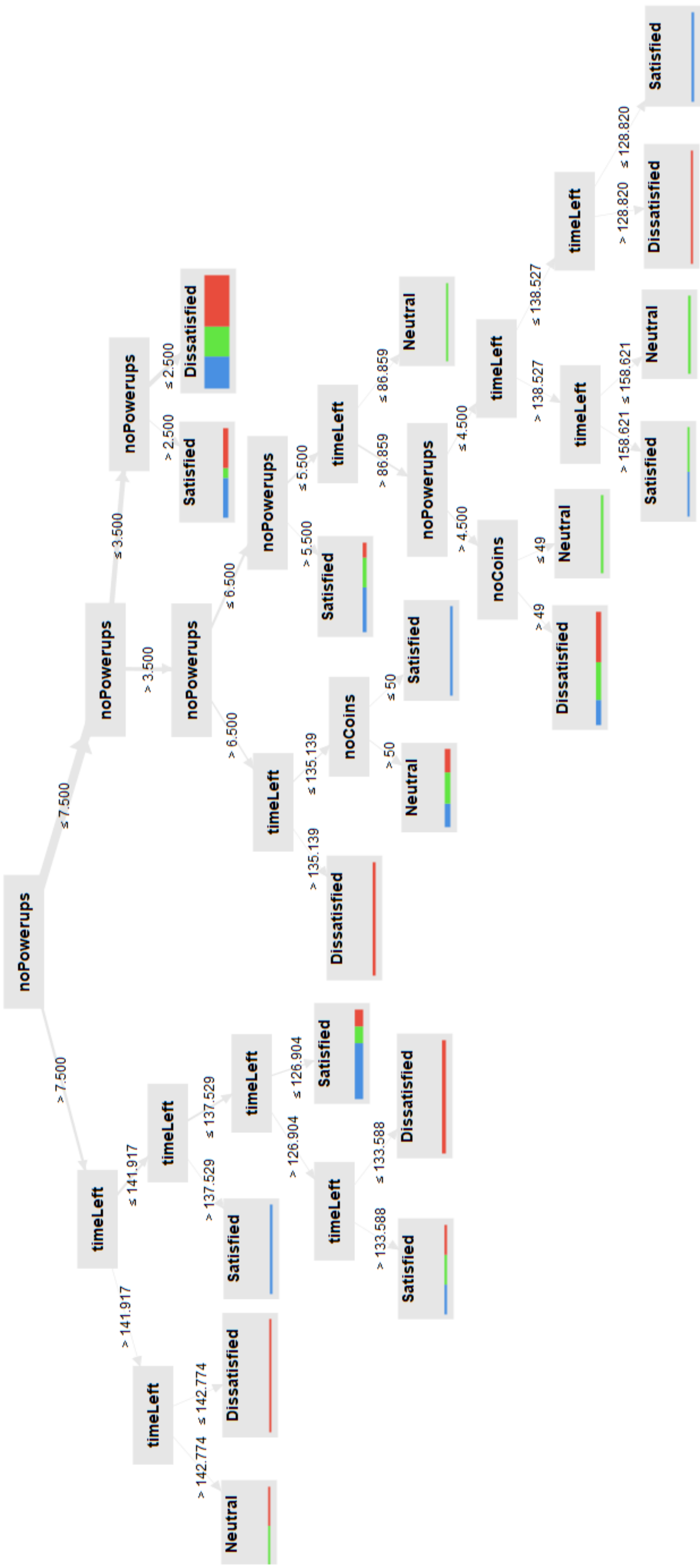

Figure B.11: Level 11 - Decision Tree.

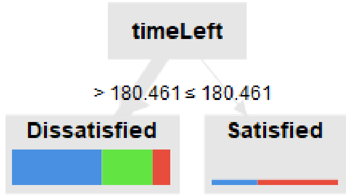

Figure B.12: Level 12 - Decision Tree.

B.2 Correlations between Levels' Elements and Players' Profiles

	Gender	Educa-tion	Age	Extra-version	Agree-ableness	Conscien-tiousness	Neuro-ticism	Open-ness
EnemiesNumber	0.04	0.00	0.08	-0.05	0.04	0.14	-0.06	-0.04
GapWidth	0.04	-0.05	-0.05	-0.01	0.07	0.06	-0.08	-0.06
EnemiesTypes	0.16	-0.01	0.02	-0.07	0.13	0.16	-0.08	0.02
Coins	0.09	-0.07	-0.03	0.02	0.10	0.11	-0.06	-0.02
Powerups	0.06	0.03	0.03	0.00	0.07	0.08	-0.07	-0.03
OverallExperience	0.11	0.01	0.06	-0.14	0.11	0.08	-0.07	-0.04
causeOfDeath	-0.03	0.06	0.01	-0.11	-0.02	0.07	0.00	0.02
timeLeft	-0.16	0.07	0.06	-0.13	-0.03	0.00	-0.01	0.02
noCoins	0.11	-0.07	-0.09	0.11	0.02	-0.01	0.00	-0.02
noPowerups	0.14	-0.08	-0.10	0.07	0.02	-0.01	0.04	0.02
noEnemies	0.06	-0.02	-0.06	0.11	0.06	0.00	-0.04	0.01

Table B.1: Level 1 - correlations between players' profiles and game elements

	Gender	Educa-tion	Age	Extra-version	Agree-ableness	Conscien-tiousness	Neuro-ticism	Open-ness
EnemiesNumber	-0.08	0.10	0.10	-0.08	0.09	-0.03	-0.08	0.05
GapWidth	-0.10	-0.05	-0.03	0.07	0.08	0.04	-0.01	-0.02
EnemiesTypes	-0.06	0.09	0.09	-0.07	0.05	0.06	-0.04	0.02
Coins	-0.09	0.07	0.04	-0.02	0.04	-0.08	-0.04	0.00
Powerups	-0.09	0.12	0.08	-0.03	0.08	-0.03	-0.03	0.02
OverallExperience	0.05	0.03	0.04	0.05	0.08	0.18	-0.03	-0.06
causeOfDeath	0.05	-0.07	0.06	0.05	-0.07	0.10	-0.03	0.07
timeLeft	0.04	-0.09	0.07	0.00	-0.15	0.00	-0.06	0.05

Table B.2: Level 2 - correlations between players' profiles and game elements

	Gender	Educa-tion	Age	Extra-version	Agree-ableness	Conscien-tiousness	Neuro-ticism	Open-ness
EnemiesNumber	-0.09	-0.05	-0.11	0.06	0.04	-0.04	0.03	0.02
GapWidth	0.07	-0.02	0.02	-0.10	0.09	-0.07	-0.02	-0.01
EnemiesTypes	-0.15	-0.11	-0.14	0.04	0.12	-0.16	-0.09	-0.05
Coins	0.18	0.06	0.11	-0.01	0.01	-0.07	0.02	-0.01
Powerups	0.06	0.08	0.08	-0.10	0.00	0.01	0.00	0.05
OverallExperience	0.12	0.07	0.04	-0.03	0.03	-0.07	0.09	-0.08
causeOfDeath	0.29	0.15	0.15	-0.03	-0.04	0.08	0.08	0.06
timeLeft	0.03	0.17	0.10	0.12	-0.02	0.05	0.08	-0.03
noCoins	-0.28	-0.15	-0.22	0.02	0.01	-0.19	-0.11	0.02
noPowerups	-0.19	-0.06	-0.14	-0.01	-0.10	-0.12	0.04	-0.03
noEnemies	-0.30	-0.18	-0.19	0.00	-0.07	-0.14	-0.05	0.01

Table B.3: Level 3 - correlations between players' profiles and game elements

	Gender	Educa-tion	Age	Extra-version	Agree-ableness	Conscien-tiousness	Neuro-ticism	Open-ness
EnemiesNumber	0.12	0.10	0.16	0.06	-0.15	0.00	0.09	-0.12
GapWidth	0.06	-0.01	-0.05	0.00	-0.03	-0.11	0.06	0.00
EnemiesTypes	0.07	0.03	0.03	-0.04	0.01	0.05	-0.07	-0.03
Coins	-0.03	0.05	0.03	-0.06	0.00	-0.10	-0.01	0.01
Powerups	-0.04	0.12	0.06	0.05	-0.16	-0.01	-0.04	-0.10
OverallExperience	0.06	0.09	0.05	0.02	-0.03	-0.07	0.13	-0.04
causeOfDeath	0.14	0.07	0.13	-0.04	-0.04	0.06	0.01	0.09
timeLeft	0.06	0.01	0.09	0.04	-0.01	0.01	0.02	0.11
noCoins	-0.17	0.02	-0.10	0.05	-0.03	-0.11	-0.01	-0.09
noPowerups	-0.09	-0.01	-0.10	0.05	-0.07	-0.02	0.01	-0.12
noEnemies	-0.06	0.01	-0.07	-0.03	0.05	-0.12	0.00	-0.03

Table B.4: Level 4 - correlations between players' profiles and game elements

	Gender	Education	Age	Extraversion	Agreeableness	Conscientiousness	Neuroticism	Openness
EnemiesNumber	-0.02	0.01	-0.01	0.10	0.00	-0.02	0.02	-0.07
GapWidth	-0.03	-0.02	-0.02	0.01	-0.02	0.08	-0.03	0.11
EnemiesTypes	-0.04	0.01	-0.02	-0.01	0.15	0.08	-0.02	-0.03
Coins	-0.02	0.01	-0.02	0.08	0.03	0.12	0.11	-0.05
Powerups	-0.02	-0.02	-0.03	0.03	0.05	0.20	0.08	0.03
OverallExperience	-0.01	0.02	-0.07	0.10	-0.03	0.04	0.09	-0.04
causeOfDeath	0.11	-0.02	0.11	-0.06	-0.05	0.02	0.01	0.03
timeLeft	-0.05	0.00	0.02	0.01	-0.06	-0.08	-0.01	0.03
noCoins	-0.12	-0.01	-0.09	0.04	0.06	0.01	-0.02	-0.01
noPowerups	-0.12	0.02	0.00	0.02	0.00	0.02	0.04	0.01
noEnemies	-0.14	0.01	-0.10	0.04	0.06	-0.02	-0.03	-0.02

Table B.5: Level 5 - correlations between players' profiles and game elements

	Gender	Education	Age	Extraversion	Agreeableness	Conscientiousness	Neuroticism	Openness
EnemiesNumber	-0.02	-0.09	0.02	-0.23	0.09	0.06	-0.04	0.14
GapWidth	-0.09	-0.11	-0.02	-0.04	0.02	0.00	-0.17	0.04
EnemiesTypes	0.13	-0.09	0.03	-0.07	0.13	-0.05	0.05	-0.04
Coins	0.00	-0.08	0.06	-0.03	0.09	-0.08	-0.11	0.04
Powerups	0.00	0.00	0.10	-0.12	0.07	-0.04	-0.03	0.10
OverallExperience	0.06	-0.07	0.04	-0.05	0.16	-0.01	0.06	0.07
causeOfDeath	0.05	-0.10	0.02	0.06	-0.09	0.05	-0.08	-0.04
timeLeft	0.01	0.05	0.18	0.05	-0.10	0.10	-0.10	0.00
noCoins	-0.02	0.00	-0.07	0.07	0.13	-0.10	0.06	-0.01
noPowerups	-0.05	0.02	-0.03	-0.07	0.10	-0.11	0.10	0.09
noEnemies	-0.11	-0.09	-0.13	0.08	0.08	-0.17	0.09	0.01

Table B.6: Level 6 - correlations between players' profiles and game elements

	Gender	Education	Age	Extraversion	Agreeableness	Conscientiousness	Neuroticism	Openness
EnemiesNumber	-0.10	0.04	0.08	-0.12	0.05	-0.08	-0.07	0.12
GapWidth	-0.07	-0.05	0.06	-0.13	0.06	0.01	-0.09	0.16
EnemiesTypes	-0.10	-0.06	-0.08	-0.10	-0.08	-0.16	-0.02	0.03
Coins	-0.10	0.13	0.18	-0.04	0.08	-0.04	-0.01	0.01
Powerups	-0.05	0.03	0.17	-0.08	0.00	0.02	-0.16	0.10
OverallExperience	-0.03	0.06	0.07	-0.05	0.14	-0.10	0.05	0.11
causeOfDeath	0.11	0.09	0.09	0.04	-0.15	0.09	0.07	-0.16
timeLeft	0.11	0.05	0.05	0.04	-0.13	0.11	0.06	-0.10
noCoins	-0.27	-0.06	-0.12	0.01	0.05	-0.14	-0.04	0.17
noEnemies	-0.13	-0.01	-0.10	0.05	0.02	-0.15	-0.09	0.07

Table B.7: Level 7 - correlations between players' profiles and game elements

	Gender	Education	Age	Extraversion	Agreeableness	Conscientiousness	Neuroticism	Openness
EnemiesNumber	-0.04	-0.09	-0.09	0.00	-0.09	-0.11	-0.05	0.00
GapWidth	0.24	-0.07	-0.04	-0.03	-0.02	0.05	0.13	-0.03
EnemiesTypes	0.03	-0.06	-0.05	-0.03	-0.08	-0.06	-0.07	-0.07
Coins	-0.05	-0.06	-0.06	0.04	-0.06	-0.03	0.00	-0.04
Powerups	-0.05	-0.09	-0.07	-0.03	-0.05	0.01	-0.06	0.03
OverallExperience	0.32	0.06	-0.03	0.01	-0.02	0.15	0.21	0.03
causeOfDeath	0.09	0.02	0.08	-0.12	-0.13	0.12	-0.04	0.14
timeLeft	-0.07	0.08	0.10	-0.05	-0.07	0.06	-0.05	0.04

Table B.8: Level 8 - correlations between players' profiles and game elements

	Gender	Education	Age	Extraversion	Agreeableness	Conscientiousness	Neuroticism	Openness
EnemiesNumber	-0.11	-0.13	-0.14	-0.09	0.03	-0.07	0.07	-0.01
GapWidth	0.04	0.02	-0.02	0.01	-0.01	-0.10	-0.06	-0.13
EnemiesTypes	-0.10	-0.16	-0.11	-0.10	0.05	0.07	0.03	-0.06
Coins	0.00	0.00	-0.02	-0.06	0.06	0.08	0.04	-0.05
Powerups	-0.03	-0.08	-0.08	-0.12	0.09	0.11	0.09	0.05
OverallExperience	-0.03	-0.04	-0.03	-0.04	0.14	0.04	0.03	-0.04
causeOfDeath	0.12	0.02	-0.08	-0.05	0.07	-0.02	0.05	-0.09
timeLeft	0.13	0.03	-0.06	0.05	0.09	-0.10	0.03	0.02
noEnemies	-0.25	-0.05	-0.11	-0.02	-0.12	-0.02	-0.02	0.02

Table B.9: Level 9 - correlations between players' profiles and game elements

	Gender	Educa-tion	Age	Extra-version	Agree-ableness	Conscien-tiousness	Neuro-ticism	Open-ness
EnemiesNumber	-0.10	-0.01	-0.06	-0.09	0.00	-0.18	-0.01	0.11
GapWidth	-0.01	0.06	0.12	0.04	0.02	0.04	0.03	0.07
EnemiesTypes	-0.13	-0.01	-0.04	-0.05	0.04	-0.01	-0.03	0.17
Coins	0.01	0.01	0.03	-0.07	0.11	-0.01	0.13	0.07
Powerups	0.14	0.06	0.02	-0.02	0.07	0.08	0.08	0.02
OverallExperience	0.15	0.05	0.08	-0.05	0.09	-0.01	0.13	0.02
causeOfDeath	0.10	-0.03	0.00	0.03	0.05	0.08	-0.17	-0.10
timeLeft	-0.03	0.02	0.02	0.13	-0.04	-0.01	-0.11	-0.06
noCoins	-0.06	-0.05	-0.07	-0.10	0.01	-0.05	0.16	0.12
noPowerups	-0.08	0.02	0.02	-0.01	0.01	0.08	0.19	0.10

Table B.10: Level 10 - correlations between players' profiles and game elements

	Gender	Educa-tion	Age	Extra-version	Agree-ableness	Conscien-tiousness	Neuro-ticism	Open-ness
EnemiesNumber	-0.08	-0.01	-0.10	0.02	-0.05	-0.10	0.05	0.05
GapWidth	0.06	-0.05	-0.06	-0.05	0.04	0.05	-0.13	0.14
EnemiesTypes	-0.08	0.12	0.04	-0.07	0.01	-0.04	0.01	0.07
Coins	0.11	0.00	0.02	0.01	0.04	-0.08	0.07	0.01
Powerups	0.17	0.00	0.00	-0.08	-0.02	0.15	0.02	0.16
OverallExperience	0.27	-0.03	-0.03	-0.06	0.11	0.04	0.19	0.14
timeLeft	-0.35	0.02	-0.09	0.06	0.02	-0.01	-0.14	0.21
noCoins	0.21	-0.01	-0.10	0.09	-0.02	-0.03	0.18	-0.20
noPowerups	-0.02	0.02	0.06	0.09	-0.12	-0.01	0.11	-0.12

Table B.11: Level 11 - correlations between players' profiles and game elements

	Gender	Educa-tion	Age	Extra-version	Agree-ableness	Conscien-tiousness	Neuro-ticism	Open-ness
EnemiesNumber	-0.04	-0.13	-0.08	-0.07	-0.14	-0.13	-0.06	0.01
GapWidth	-0.18	-0.04	-0.09	0.06	-0.12	-0.10	-0.13	-0.01
EnemiesTypes	-0.04	-0.09	-0.05	-0.07	-0.18	-0.10	-0.05	0.04
Coins	0.00	-0.09	-0.03	-0.09	-0.16	-0.02	-0.02	0.06
Powerups	0.01	-0.10	-0.03	-0.06	-0.14	-0.02	-0.06	0.01
OverallExperience	-0.03	-0.17	-0.16	-0.12	-0.03	-0.09	-0.04	0.12
causeOfDeath	0.10	-0.10	0.05	0.13	-0.05	-0.01	0.10	0.16
timeLeft	0.22	0.07	0.12	0.12	-0.18	0.02	0.16	0.03

Table B.12: Level 12 - correlations between players' profiles and game elements

B.3 Interaction with Levels' Elements

B.3.1 Enemies

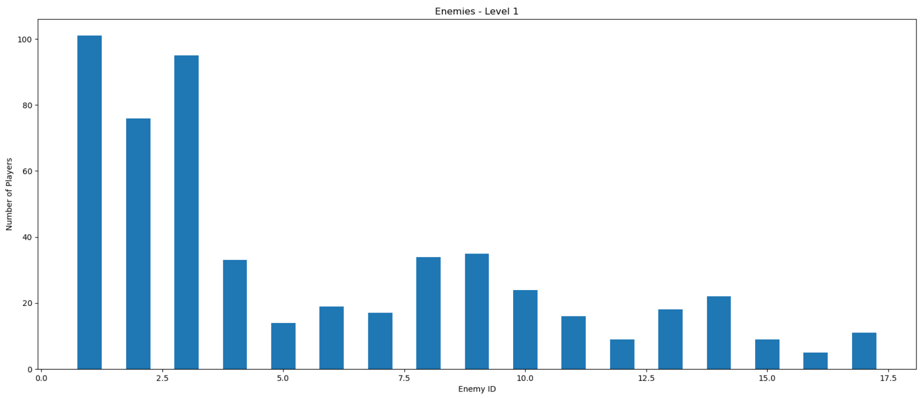

Figure B.13: Level 1 - Enemy kill by ID.

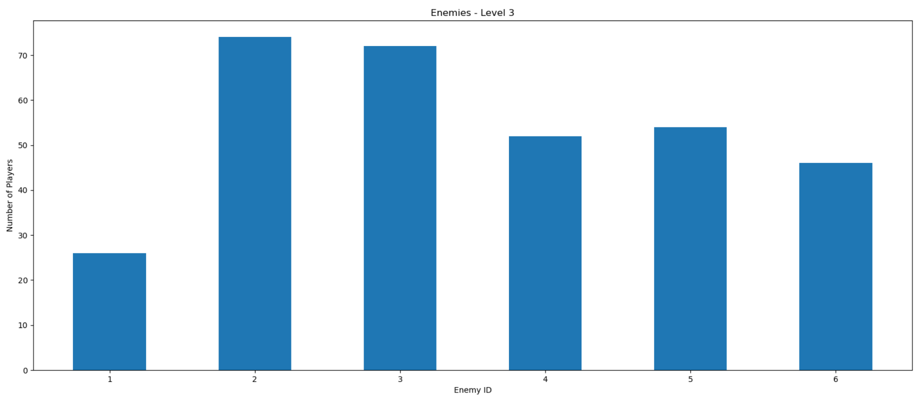

Figure B.14: Level 3 - Enemy kill by ID.

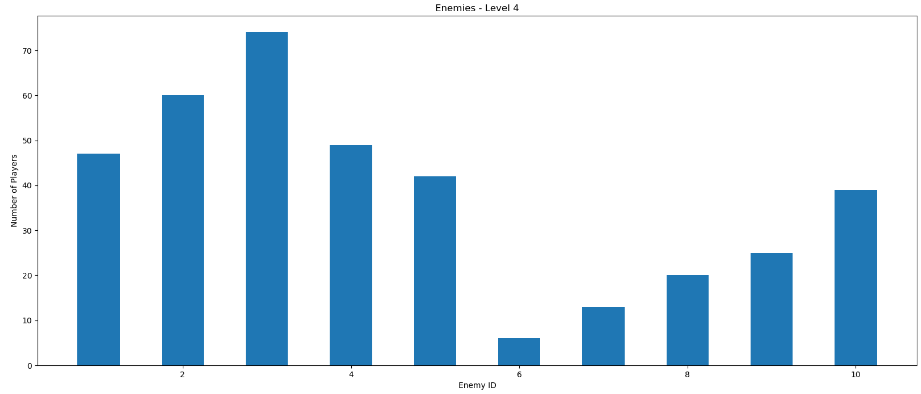

Figure B.15: Level 4 - Enemy kill by ID.

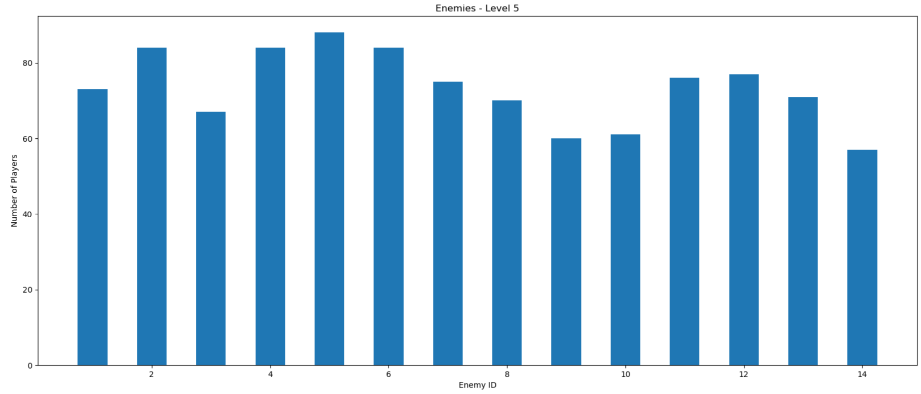

Figure B.16: Level 5 - Enemy kill by ID.

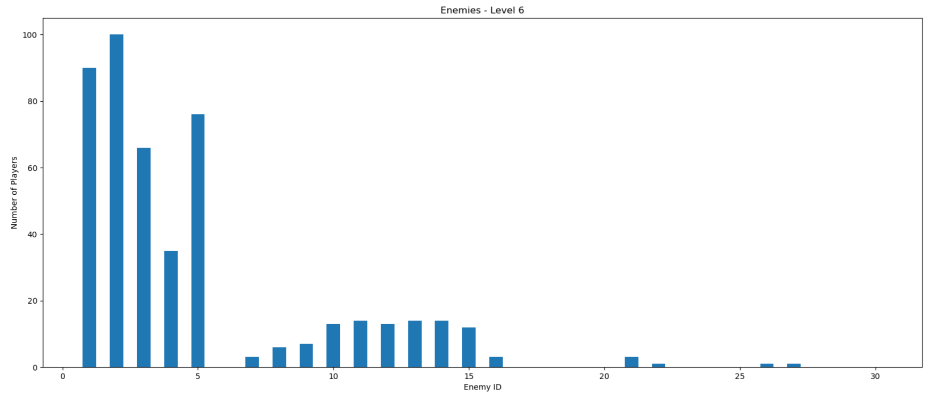

Figure B.17: Level 6 - Enemy kill by ID.

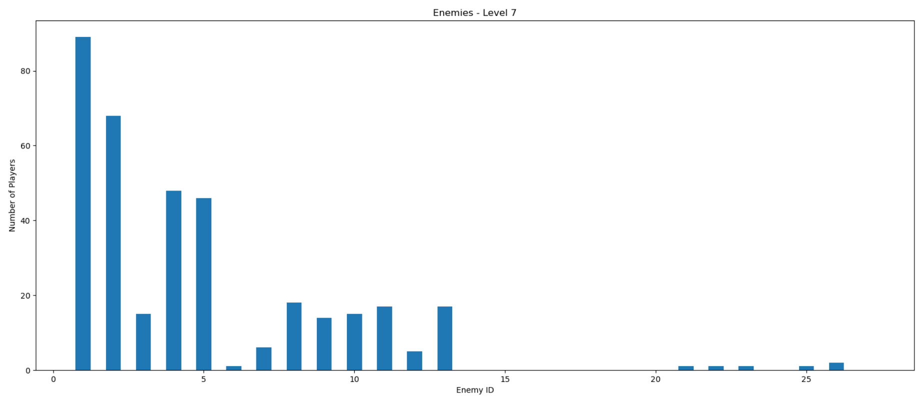

Figure B.18: Level 7 - Enemy kill by ID.

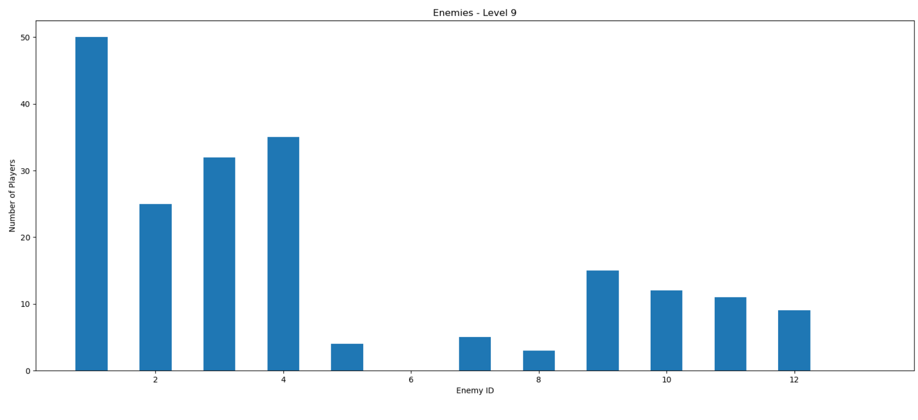

Figure B.19: Level 9 - Enemy kill by ID.

B.3.2 Coins

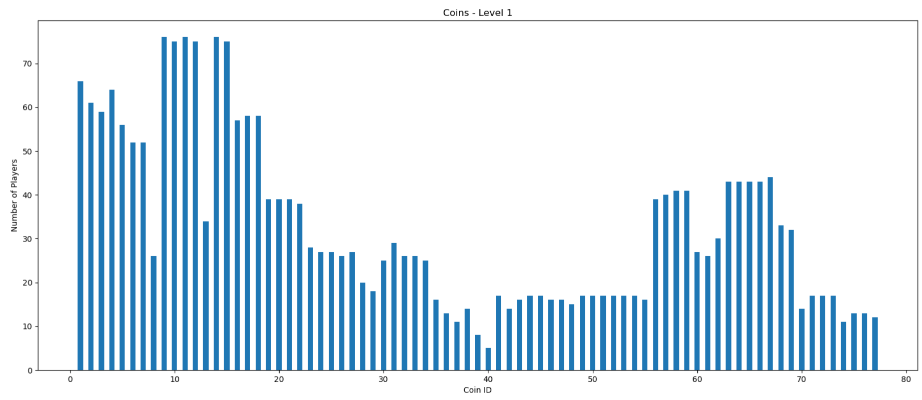

Figure B.20: Level 1 - Coin collection by ID.

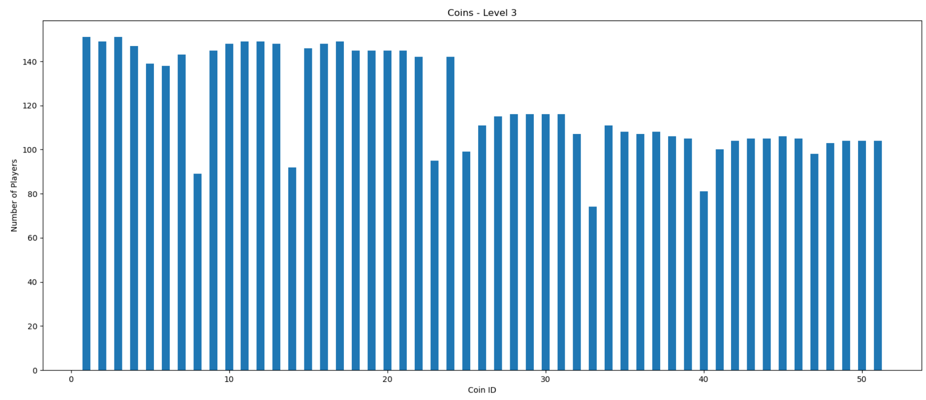

Figure B.21: Level 3 - Coin collection by ID.

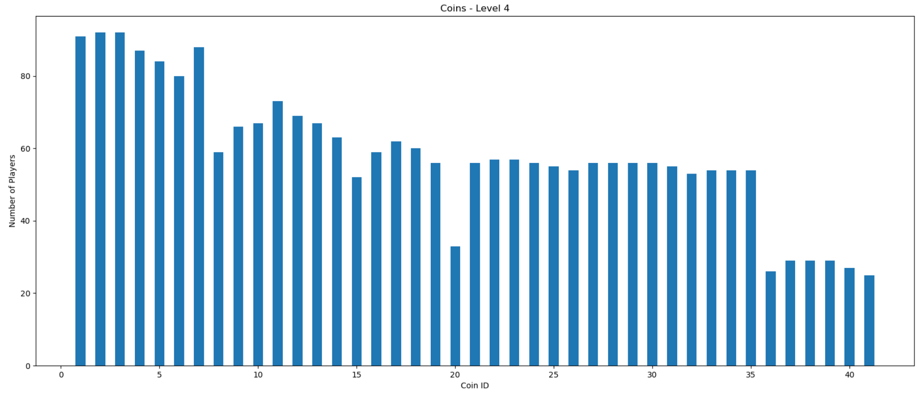

Figure B.22: Level 4 - Coin collection by ID.

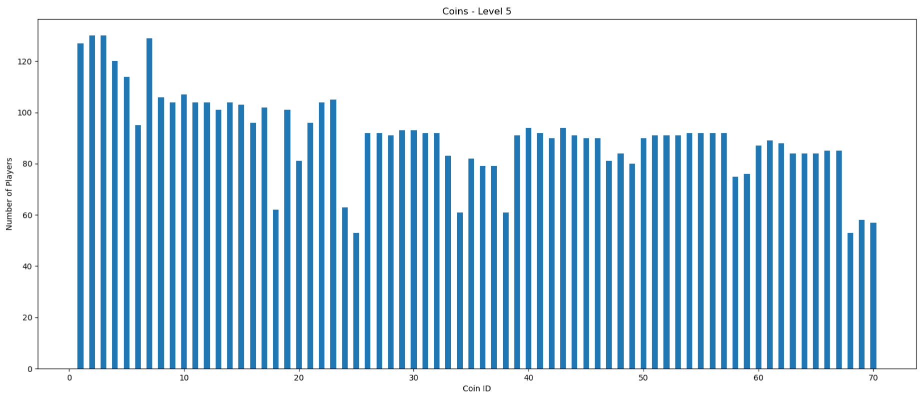

Figure B.23: Level 5 - Coin collection by ID.

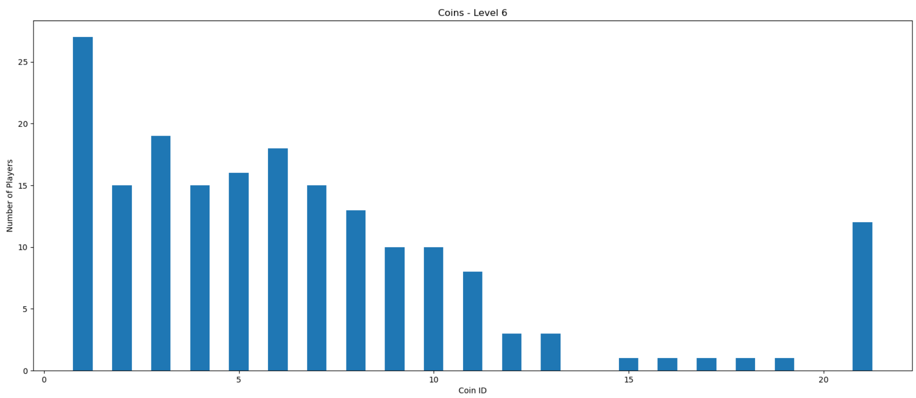

Figure B.24: Level 6 - Coin collection by ID.

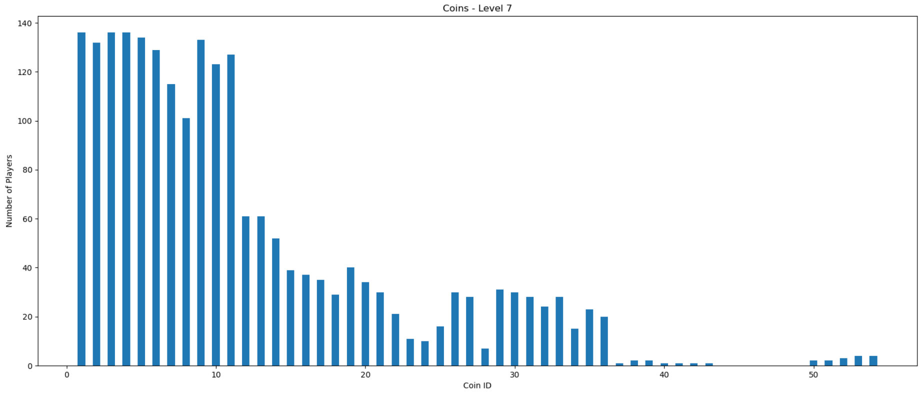

Figure B.25: Level 7 - Coin collection by ID.

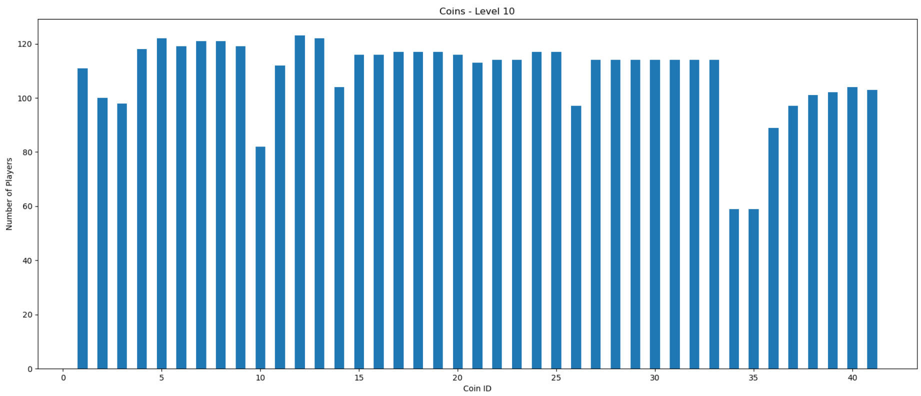

Figure B.26: Level 10 - Coin collection by ID.

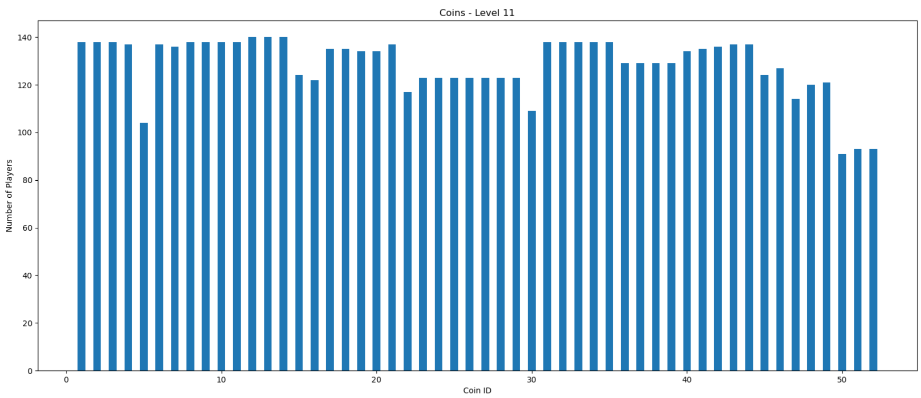

Figure B.27: Level 11 - Coin collection by ID.

B.3.3 Powerups

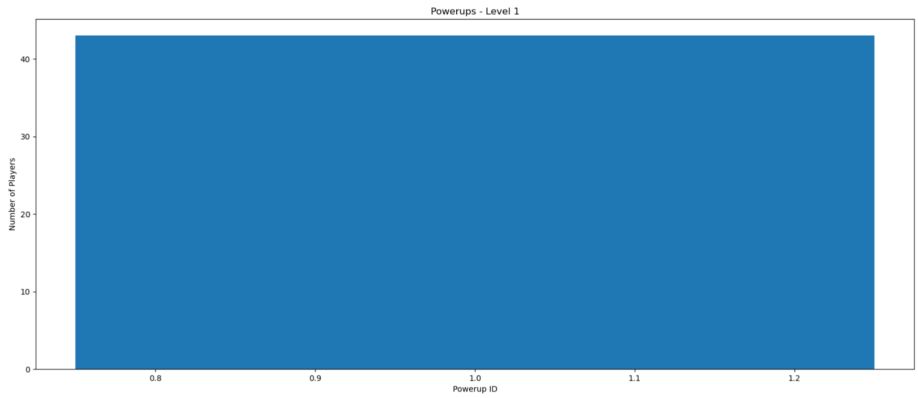

Figure B.28: Level 1 - Powerup collection by ID.

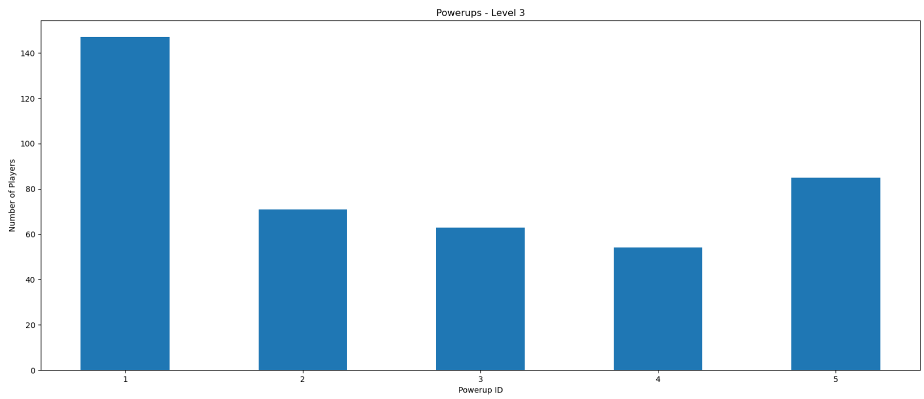

Figure B.29: Level 3 - Powerup collection by ID.

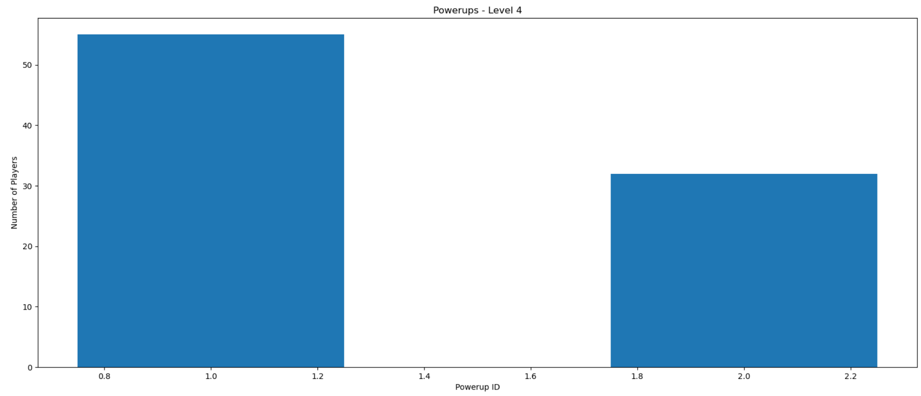

Figure B.30: Level 4 - Powerup collection by ID.

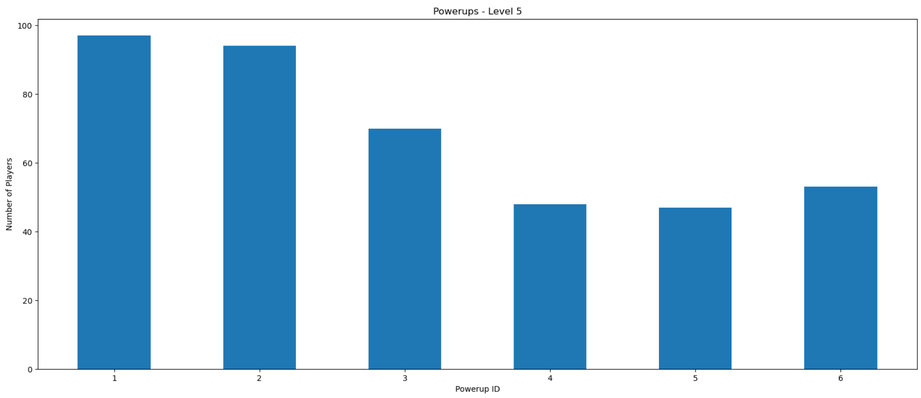

Figure B.31: Level 5 - Powerup collection by ID.

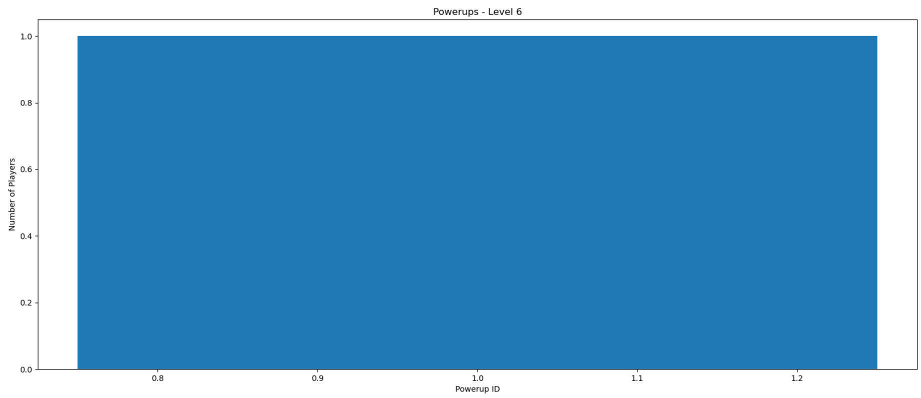

Figure B.32: Level 6 - Powerup collection by ID.

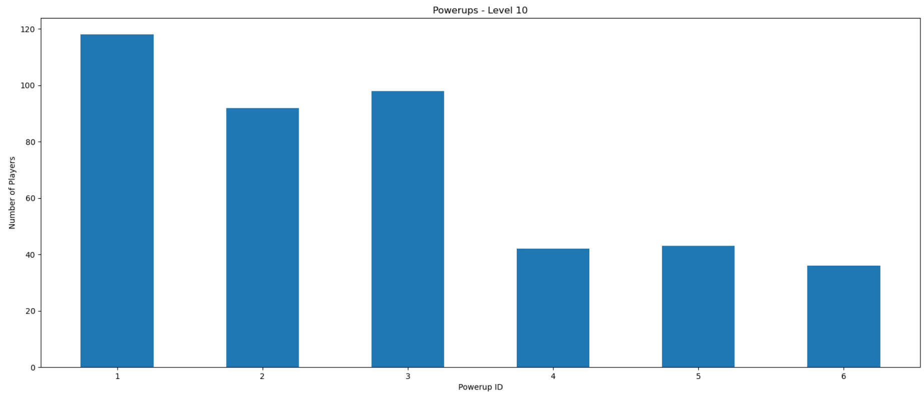

Figure B.33: Level 10 - Powerup collection by ID.

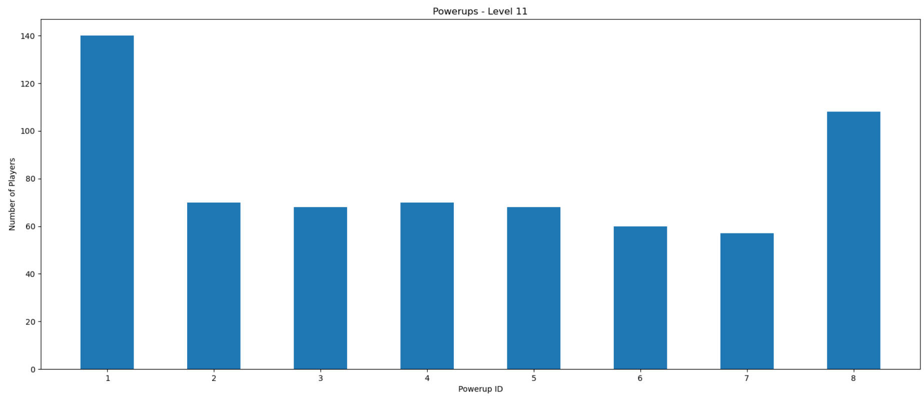

Figure B.34: Level 11 - Powerup collection by ID.

Appendix C

Useful scripts for data collection, storing and analysis

C.1 Collect and Store data - Google App Script

```
// adapted from: http://mashe.hawksey.info/2014/07/goog  le-sheets-as-a-database-insert-with-apps-script-using-postget-methods-with-ajax-example/
// original gist: https://gist.github.com/willpatera/ee41ae374d3c9839c2d6

function doGet(e){
  return handleResponse(e.parameter["Sheet"], e);
}

function doPost(e){
  var parameters = JSON.parse(e.postData.contents);
  return handlePostResponse(parameters["Sheet"], parameters);
}

var SCRIPT_PROP = PropertiesService.getScriptProperties(); // new property service

function handleResponse(SHEET_NAME, e) {
  // shortly after my original solution Google announced the LockService[1]
  // this prevents concurrent access overwritting data
  // [1] http://googleappsdeveloper.blogspot.co.uk/2011/10/concurrency-and-google-apps-script.html
  // we want a public lock, one that locks for all invocations
  var lock = LockService.getPublicLock();
  lock.waitLock(30000);  // wait 30 seconds before conceding defeat.

  try {
    // next set where we write the data - you could write to multiple/alternate destinations
    var doc = SpreadsheetApp.openById(SCRIPT_PROP.getProperty("key"));
```

```
    var sheet = doc.getSheetByName(SHEET_NAME);

    // we'll assume header is in row 1 but you can override with header_row in GET/POST data
    var headRow = e.parameter.header_row || 1;
    var headers = sheet.getRange(1, 1, 1, sheet.getLastColumn()).getValues()[0];
    var nextRow = sheet.getLastRow()+1; // get next row
    var row = [];
    // loop through the header columns
    for (i in headers){
      if (headers[i] == "Timestamp"){ // special case if you include a 'Timestamp' column
        row.push(new Date());
      } else { // else use header name to get data
        row.push(e.parameter[headers[i]]);
      }
    }
    // more efficient to set values as [][] array than individually
    sheet.getRange(nextRow, 1, 1, row.length).setValues([row]);
    // return json success results
    return ContentService
          .createTextOutput(JSON.stringify({"result":"success", "row": nextRow}))
          .setMimeType(ContentService.MimeType.JSON);
  } catch(e){
    // if error return this
    return ContentService
          .createTextOutput(JSON.stringify({"result":"error", "error": e}))
          .setMimeType(ContentService.MimeType.JSON);
  } finally { //release lock
    lock.releaseLock();
  }
}

function handlePostResponse(SHEET_NAME, e) {
  // shortly after my original solution Google announced the LockService[1]
  // this prevents concurrent access overwritting data
  // [1] http://googleappsdeveloper.blogspot.co.uk/2011/10/concurrency-and-google-apps-script.html
  // we want a public lock, one that locks for all invocations
  var lock = LockService.getPublicLock();
  lock.waitLock(30000);  // wait 30 seconds before conceding defeat.

  try {
    // next set where we write the data - you could write to multiple/alternate destinations
    var doc = SpreadsheetApp.openById(SCRIPT_PROP.getProperty("key"));
    var sheet = doc.getSheetByName(SHEET_NAME);

    var headers = sheet.getRange(1, 1, 1, sheet.getLastColumn()).getValues()[0];
```

```
    var nextRow = sheet.getLastRow()+1; // get next row
    var row = [];
    // loop through the header columns
    for (i in headers){
      if (headers[i] == "Timestamp"){ // special case if you include a 'Timestamp' column
        row.push(new Date());
      } else { // else use header name to get data
        row.push(e[headers[i]]);
      }
    }
    // more efficient to set values as [][] array than individually
    sheet.getRange(nextRow, 1, 1, row.length).setValues([row]);
    // return json success results
    return ContentService
          .createTextOutput(JSON.stringify({"result":"success", "row": nextRow}))
          .setMimeType(ContentService.MimeType.JSON);
  } catch(e){
    // if error return this
    return ContentService
          .createTextOutput(JSON.stringify({"result":"error", "error": e}))
          .setMimeType(ContentService.MimeType.JSON);
  } finally { //release lock
    lock.releaseLock();
  }
}

function setup() {
    var doc = SpreadsheetApp.getActiveSpreadsheet();
    SCRIPT_PROP.setProperty("key", doc.getId());
}
```

Listing C.1: Collect and Store data into a Google Spreadsheet

C.2 First Study Data Analysis - Python

```
class Personality:
    def __init__(self, extraversion, aggreableness, conscientiousness, neuroticism, openness):
        self.extraversion = extraversion
        self.aggreableness = aggreableness
        self.conscientiousness = conscientiousness
        self.neuroticism = neuroticism
        self.openness = openness
```

Listing C.2: Personality class

```
import pandas as pd
from matplotlib import pyplot as plt
import numpy as np
from ast import literal_eval
from scipy.stats import pearsonr, spearmanr, kendalltau
from personality import Personality

CoinsPerLevel = [77, 0, 51, 41, 70, 21, 54, 0, 0, 41, 52, 0]
PowerupsPerLevel = [1, 0, 5, 2, 6, 1, 0, 0, 0, 6, 8, 0]
EnemiesPerLevel = [17, 0, 6, 10, 14, 30, 27, 0, 13, 0, 0, 0]

def getPersonalityElementsFromDataFrame(df, elem_col):
    elements = []

    for elem in pd.DataFrame(df, columns=[elem_col]).values:
        elements.append(elem[0])

    return elements

def getElementsFromDataFrame(df, elem_col):
    elements = []

    for elem in pd.DataFrame(df, columns=[elem_col]).values:
        elements += literal_eval(elem[0])

    return elements

def plot(df, elem_col, num_elems, plot_title, x_label):
    elements = getElementsFromDataFrame(df, elem_col)
    plot_bins = []
    for i in range(1, num_elems + 2):
        plot_bins.append(i)

    fig, ax = plt.subplots(figsize =(20, 8))
    ax.hist(np.array(elements), bins = plot_bins, rwidth=0.5, align=’left’)
    ax.set_title(plot_title)
    ax.set_xlabel(x_label)
    ax.set_ylabel(’Number of Players’)

    fig.savefig(’./images/{}’.format(plot_title))

    plt.close(fig)

def createPersonality(df):
    extraversion = getPersonalityElementsFromDataFrame(df, ’Extraversion’)
    aggreableness = getPersonalityElementsFromDataFrame(df, ’Agreeableness’)
    conscientiousness = getPersonalityElementsFromDataFrame(df, ’Conscientiousness’)
    neuroticism = getPersonalityElementsFromDataFrame(df, ’Neuroticism’)
```

```
    openness = getPersonalityElementsFromDataFrame(df, 'Openness')

    return Personality(extraversion, aggreableness, conscientiousness, neuroticism,
         openness)

def pearsonrPersonality(personality, data):
    corrE_P, _ = pearsonr(personality.extraversion, data)
    corrA_P, _ = pearsonr(personality.aggreableness, data)
    corrC_P, _ = pearsonr(personality.conscientiousness, data)
    corrN_P, _ = pearsonr(personality.neuroticism, data)
    corrO_P, _ = pearsonr(personality.openness, data)

    return (corrE_P, corrA_P, corrC_P, corrN_P, corrO_P) if corrE_P >= 0.5 or
        corrA_P >= 0.5 or corrC_P >= 0.5 or corrN_P >= 0.5 or corrO_P >= 0.5 else (
        None, None, None, None, None)

def spearmanrPersonality(personality, data):
    corrE_P, _ = spearmanr(personality.extraversion, data)
    corrA_P, _ = spearmanr(personality.aggreableness, data)
    corrC_P, _ = spearmanr(personality.conscientiousness, data)
    corrN_P, _ = spearmanr(personality.neuroticism, data)
    corrO_P, _ = spearmanr(personality.openness, data)

    return (corrE_P, corrA_P, corrC_P, corrN_P, corrO_P) if corrE_P >= 0.5 or
        corrA_P >= 0.5 or corrC_P >= 0.5 or corrN_P >= 0.5 or corrO_P >= 0.5 else (
        None, None, None, None, None)

def kendalltauPersonality(personality, data):
    corrE_P, _ = kendalltau(personality.extraversion, data)
    corrA_P, _ = kendalltau(personality.aggreableness, data)
    corrC_P, _ = kendalltau(personality.conscientiousness, data)
    corrN_P, _ = kendalltau(personality.neuroticism, data)
    corrO_P, _ = kendalltau(personality.openness, data)

    return (corrE_P, corrA_P, corrC_P, corrN_P, corrO_P) if corrE_P >= 0.5 or
        corrA_P >= 0.5 or corrC_P >= 0.5 or corrN_P >= 0.5 or corrO_P >= 0.5 else (
        None, None, None, None, None)

def correlateElementsAndTraits(df, Level, elem_col, noElems, personality):
    elements = []
    for i in range(0, noElems):
        elements.append([])

    frame = pd.DataFrame(df, columns=[elem_col]).values

    # Array with percentage (%) of elements for each person
    people = []
    for elem in frame:
        elems = literal_eval(elem[0])
```

```
        for j in range(0, noElems):
            elements[j].append(1 if (j in elems) else 0)
        people.append(len(elems) / noElems)

    # Correlation between personalities and percentage (%) of elements
    peopleE_P, peopleA_P, peopleC_P, peopleN_P, peopleO_P = pearsonrPersonality(
        personality, people)
    if peopleE_P != None:
        print("PEARSONR: Level-{} {}: {}; {}; {}; {}; {}".format(Level, elem_col,
            peopleE_P, peopleA_P, peopleC_P, peopleN_P, peopleO_P))

    peopleE_S, peopleA_S, peopleC_S, peopleN_S, peopleO_S = spearmanrPersonality(
        personality, people)
    if peopleE_S != None:
        print("SPEARMANR: Level-{} {}: {}; {}; {}; {}; {}".format(Level, elem_col,
            peopleE_S, peopleA_S, peopleC_S, peopleN_S, peopleO_S))

    peopleE_K, peopleA_K, peopleC_K, peopleN_K, peopleO_K = kendalltauPersonality(
        personality, people)
    if peopleE_K != None:
        print("KENDALLTAU: Level-{} {}: {}; {}; {}; {}; {}".format(Level, elem_col,
             peopleE_K, peopleA_K, peopleC_K, peopleN_K, peopleO_K))

    # Correlate collection of each element with personalities
    for i in range(0, noElems):
        corrE_P, corrA_P, corrC_P, corrN_P, corrO_P = pearsonrPersonality(
            personality, elements[i])

        if corrE_P != None:
            print("PEARSONR: Level-{} {}-{}: {}; {}; {}; {}; {}".format(Level,
                elem_col, i, corrE_P, corrA_P, corrC_P, corrN_P, corrO_P))

        corrE_S, corrA_S, corrC_S, corrN_S, corrO_S = spearmanrPersonality(
            personality, elements[i])

        if corrE_S != None:
            print("SPEARMANR: Level-{} {}-{}: {}; {}; {}; {}; {}".format(Level,
                elem_col, i, corrE_S, corrA_S, corrC_S, corrN_S, corrO_S))

        corrE_K, corrA_K, corrC_K, corrN_K, corrO_K = kendalltauPersonality(
            personality, elements[i])

        if corrE_K != None:
            print("KENDALLTAU: Level-{} {}-{}: {}; {}; {}; {}; {}".format(Level,
                elem_col, i, corrE_K, corrA_K, corrC_K, corrN_K, corrO_K))

def correlateTimeLeftOnWinning(df, Level, p):
    causeOfDeath = pd.DataFrame(df, columns=['causeOfDeath']).values
    timeLeft = pd.DataFrame(df, columns=['timeLeft']).values
```

```python

    _extraversion = []
    _aggreableness = []
    _conscientiousness = []
    _neuroticism = []
    _openness = []
    _timeLeft = []

    for i in range(0, len(causeOfDeath)):
        if causeOfDeath[i] == -1:
            _extraversion.append(p.extraversion[i])
            _aggreableness.append(p.aggreableness[i])
            _conscientiousness.append(p.conscientiousness[i])
            _neuroticism.append(p.neuroticism[i])
            _openness.append(p.openness[i])
            _timeLeft.append(timeLeft[i][0])

    if len(_timeLeft) < 2:
        return

    new_p = Personality(_extraversion, _aggreableness, _conscientiousness, _neuroticism, _openness)
    _e, _a, _c, _n, _p = pearsonrPersonality(new_p, _timeLeft)
    if _e != None:
        print('PEARSONR Time Left, Level {}: {} {} {} {} {}'.format(Level, _e, _a, _c, _n, _p))

    _e, _a, _c, _n, _p = spearmanrPersonality(new_p, _timeLeft)
    if _e != None:
        print('SPEARMANR Time Left, Level {}: {} {} {} {} {}'.format(Level, _e, _a, _c, _n, _p))

    _e, _a, _c, _n, _p = kendalltauPersonality(new_p, _timeLeft)
    if _e != None:
        print('KENDALLTAU Time Left, Level {}: {} {} {} {} {}'.format(Level, _e, _a, _c, _n, _p))

def removeOutliers(level_data, hasCoins, hasPowerups, hasEnemies):
    if not hasCoins:
        level_data = level_data[level_data.Coins == "Not applicable to the previous level"]

    if not hasPowerups:
        level_data = level_data[level_data.Powerups == "Not applicable to the previous level"]

    if not hasEnemies:
        level_data = level_data[level_data.EnemiesNumber == "Not applicable to the previous level"]
```

```
        level_data = level_data[level_data.EnemiesTypes == "Not applicable to the previous level"]

    return level_data

demographics = pd.read_excel(r'First Prototype.xlsx', sheet_name='Demographics').drop_duplicates(['GUID'], keep='last')

for Level in range(1, 13):
    level_data = pd.read_excel(r'First Prototype.xlsx', sheet_name='Level_{}'.format(Level)).drop_duplicates(['GUID'], keep='last')

    level_data_without_outliers = removeOutliers(level_data, CoinsPerLevel[Level - 1] != 0, PowerupsPerLevel[Level - 1] != 0, EnemiesPerLevel[Level - 1] != 0)

    df = demographics.merge(level_data)
    df1 = demographics.merge(level_data_without_outliers)

    p = createPersonality(df) # Personality Traits

    if CoinsPerLevel[Level - 1] != 0:
        plot(df, 'collectedCoins', CoinsPerLevel[Level - 1], 'Coins - Level {}'.format(Level), 'Coin ID')
        correlateElementsAndTraits(df, Level, 'collectedCoins', CoinsPerLevel[Level - 1], p)

    if PowerupsPerLevel[Level - 1] != 0:
        plot(df, 'collectedPowerups', PowerupsPerLevel[Level - 1], 'Powerups - Level {}'.format(Level), 'Powerup ID')
        correlateElementsAndTraits(df, Level, 'collectedPowerups', PowerupsPerLevel[Level - 1], p)

    if EnemiesPerLevel[Level - 1] != 0:
        plot(df, 'killedEnemies', EnemiesPerLevel[Level - 1], 'Enemies - Level {}'.format(Level), 'Enemy ID')
        correlateElementsAndTraits(df, Level, 'killedEnemies', EnemiesPerLevel[Level - 1], p)

    correlateTimeLeftOnWinning(df1, Level, p)

# Show plot
# plt.show()
```

Listing C.3: Data analysis script