FACULTADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Affective Narratives for Engagement in Digital Games

Matilde Freilão

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Rui Rodrigues

Second Supervisor: Filipe Rodrigues

July 22, 2020

Affective Narratives for Engagement in Digital Games

Matilde Freilão

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Prof. António Coelho
External Examiner: Prof. Nélson Zagalo
Supervisor: Prof. Rui Rodrigues
Second Supervisor: Prof. Filipe Rodrigues

July 22, 2020

Abstract

Stories can be told in many forms, but one that is especially powerful is video games. Nowadays, video game developers, when looking at ways to make a game more realistic and immersive, tend to turn to the interactive storytelling component of games. Most games have linear storylines, while some offer branching storylines, where the player can choose what to do next and be more involved in the story. However, these game stories, with or without branching paths, are often not enough to promote engagement and self-reflection (the activity of thinking about your own feelings and behavior, and the reasons that may lie behind them) in players throughout the game because the main traces of the story are still the same for every player, regardless of their different states of mind, emotions while playing and personality.

The objective of this thesis is to take an extra step in interactive storytelling and promote the development of tailor-made narratives, so that players can have a more engaging experience that enables self-reflection. A possible solution to achieve this is to collect the player's emotional data and use that data to adapt a game narrative to the player's emotions. It is expected that the resulting game story will be more engaging, promoting self-reflection, as opposed to other game stories.

This thesis presents the conceptualization, implementation and evaluation of a framework that can adapt game narratives to the player's emotions, by using an annotated narrative structure created by the game designer and the affective profile (the emotions) provided by the player. The framework is then able to create adapted narratives by choosing the narrative elements that fit the current player. These narrative elements can be for example the setting in which the story takes place, the dialogue of the characters, the gameplay mechanics and the order of the story events.

To learn whether games built with this framework would be more engaging and better game experiences than a game with a non-adapted narrative, user tests were performed. These tests consisted of the users playing a version of the game with a default narrative and a version with an adapted narrative, in a random order. The results obtained from this experiment show that there was an improvement in engagement and game experience, from the default version to the adapted version. Additionally, the majority of the players stated that they preferred the adapted version and would like to play a game like this again.

Keywords: video games, interactive narratives, affective narratives, self-reflection, engagement

Acknowledgements

I would like to start by thanking my mom Fernanda, my dad Mário, my stepmom Cristina and the rest of my family for loving and supporting me unconditionally.

I also wish to thank my supervisors, Rui Rodrigues and Filipe Rodrigues, for the helpful advice, constructive criticism and guidance given to me during this process.

To Catarina and Tiago, and to every friend and colleague with whom I shared this experience in FEUP, thank you, for your friendship and for helping me overcome many hardships.

To my childhood friends, Cláudia, Catarina, Mariana, Iris, Gabriela and Francisca, thank you for always making me smile whenever I felt troubled.

And to Inês and Zé, thank you for all the movie nights and junk food dinners where I could forget my troubles.

Last, but not least, to my boyfriend, Francisco, whom, without his support, the completion of this thesis would not have been possible. Thank you for always staying by my side.

Matilde Freilão

"Despite everything, it's still you."

in *Undertale* by Toby Fox

Contents

List of Figures

List of Tables

Abbreviations

NPC	Non Playable Character
PC	Playable Character
RPG	Role Playing Game
PANAS	Positive and Negative Affect Schedule
POMS	Profile of Mood States
DEQ	Discrete Emotions Questionnaire
JSON	JavaScript Object Notation
DFS	Depth-First Search
GEQ	Game Experience Questionnaire

Chapter 1

Introduction

Humans have been telling stories since the times they could speak, probably even before that. We tell stories to each other by talking but also by painting, by singing, by writing novels, short-stories and plays, by filming, and by using social-networks [Khan Academy, 2017]. Fictional stories allow us to escape reality and be able to experience different feelings and explore other worlds [Rabin, 2013], but stories are also important because they are the way we make sense of the world [Peng et al., 2018] and convey information to others [Kaptein and Broekens, 2015].

Additionally, stories are deeply connected with our emotions: our emotional systems are crucial for the creation and understanding of stories but stories also help us understand and develop our own emotional lives [Hogan, 2011]. By engaging in affective narratives (stories that express emotions) that mirror our own situation or current emotions, we often find ourselves meditating upon our own feelings and actions and the reasons that lie before them. This is called self-reflection [Cambridge Dictionary, 2020] and can be highly beneficial since it motivates us to change our behaviours for our own well-being or of our social, biological environment [Peng et al., 2018].

1.1 Context, Motivation and Objective

The same story has many ways of being told, but one medium that is especially powerful is video games. Thanks to their interactive component, players are able to experience stories in a different perspective: stories about "someone else" become stories about "us". Game stories used to be predominantly linear, meaning that every player would experience the same story, independently from their playthrough. However, more and more games with branching storylines are being made, where players are able to choose which path they want to take in the story. This appeals to players because of the high degree of customization of their game experience but comes with some shortcomings. Branching stories are harder and more costly to make, since all options may lead to different scenarios, which usually means more script and more content must be created. Rabin even says that branching storylines are a waste of resources, since most players will not even get to enjoy the game elements that fall outside of their path [Rabin, 2013]. Other ways of implementing branching exist, such as using *parallel paths*, which are branching plots that eventually join back to the main branch of the story. These implementations are less costly than *pure branching*, but players end up complaining about the lack of impact of their choices. This lack of branching in the game story also makes it so that game experiences, although fun, do not engage players in self-reflection, since the main traces of the story are the same for every player, not adapting to their different states of minds or emotions.

To respond to these problems it might be beneficial to search for other means of making a game story more personal. To this end, adapting a game's story to the player's state of mind and emotions might be a good solution to create experiences that are more personal and consequently induce self-reflection in players. Self-reflection not only has the ability of inducing positive behavior change [Peng et al., 2018] but, according to a study conducted by Oliver et al. [Oliver et al., 2016], experiencing introspection and self-reflection while playing video games helps in creating more enriching game experiences, making players recall those games as meaningful, which is something that gamers appreciate [Oliver et al., 2016]. Interestingly, the data gathered in this study also shows that these feelings of introspection and self-reflection were mostly strongly associated with the narrative of the game.

In conclusion, the objective of this thesis is to research and build a framework that is capable of helping the creation of an adapted story for a video game, based on the player's emotions and feelings, so that players can engage in self-reflection and have a more meaningful and engaging experience.

1.2 Research Questions

To guide the process of researching for the creation of this proposed framework, the following research questions were identified:

1. How do affective narratives work in games?

2. How to adapt a game's narrative to its player?

3. What criteria should be used to adapt the narrative?

4. Do personalized affective game narratives really induce self-reflection, thus impacting the player's engagement in the game?

The first three questions are the "kick-starters" of the research needed for the creation of the proposed framework: what to adapt, how to adapt and how should it be adapted. The fourth question is what this thesis intends to answer after user-tests are done using a game that was built with the help of the framework.

1.3 Document Structure

This document is divided in four chapters. Chapter 1, which this section belongs to, is the introductory chapter, where the context, motivation and objective are explained. The research questions are also outlined in this chapter. Chapter 2 is a review of the state of the art in the fields of game adaptivity, game narratives and player modeling. Chapter 3 details the conceptualization and the implementation of the aforementioned framework. Chapter 4 shows the user tests and the results and analysis that were done to test whether the framework could help in making games that are more engaging. Finally, chapter 5 contains the main conclusions of this work.

Chapter 2

Game Adaptivity through Narratives: a review

Typically, video games offer the same content, regardless of the player experiencing it. The game's content, rules, narratives and environments are usually static, while the player dynamically interacts with them. This kind of design is used because it allows a greater control, robustness and testability of the video game for the developers [Lopes and Bidarra, 2011]. At the same time, the more choices the game designer plans to give to the player, the more complex it is to create the game, inherently boosting production and time costs [Alves Nogueira, 2016].

Such rigidity in the game content might lead to predictability of the game outcomes by the player [Lopes and Bidarra, 2011], since the game's rules and gameplay do not adapt to the player's actions. For example, players can repeatedly exploit the same strategy to overcome an obstacle, which can make the game less interesting to them. Other common issue is that the game's "replayability value" (the game's content lifetime) is almost none: the player's are not encouraged to keep playing the game after finishing it for the first time, since its content remains the same through the different playthroughs [Alves Nogueira, 2016]. This is especially critical to story-driven games, as the player might lose interest in the game as soon as the story is fully revealed.

To counter these problems, developers turned to means of generating or altering game content according to the player's characteristics or actions. This is called game adaptivity.

2.1 Game Adaptivity

Game adaptivity's main objective is to make a game experience more appealing by making it more unpredictable, unique, harder or easier, depending on the player.

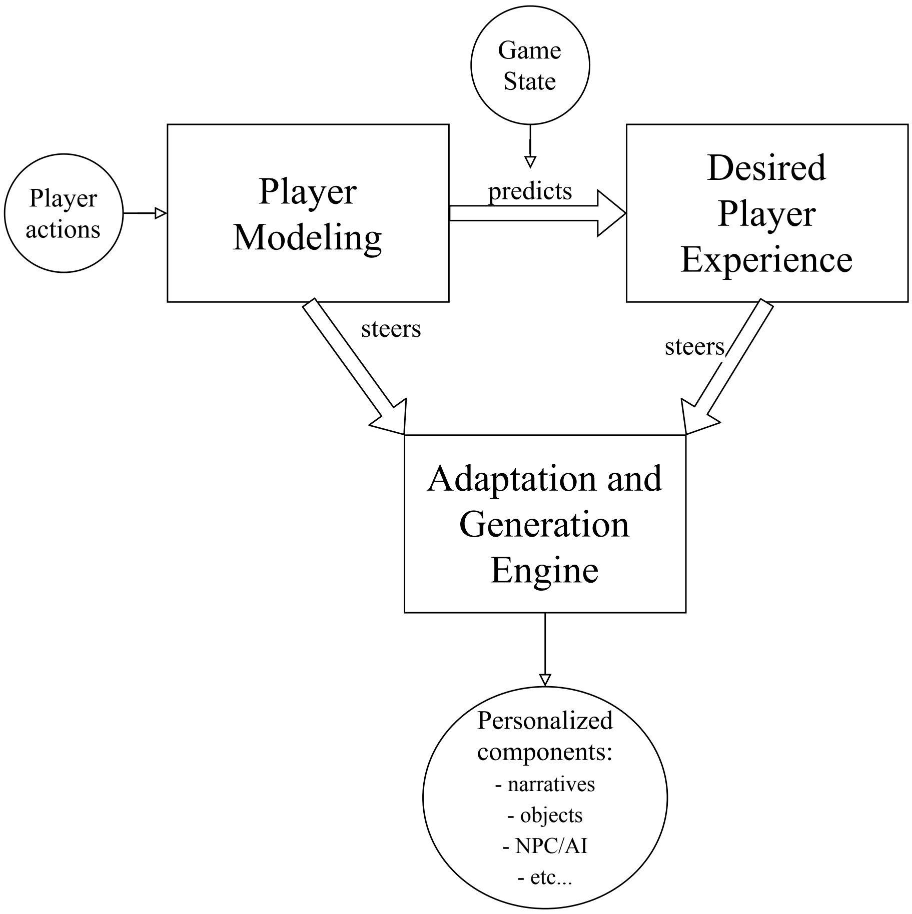


Figure 2.1: Overview of game adaptivity architectural principles as proposed by [Lopes and Bidarra, 2011]

Lopes and Bidarra [Lopes and Bidarra, 2011] describe the architectural principles of game adaptivity (figure 2.1) as follows: player actions, recorded by game logs or other means, are used to create a player model, which in turn, according to the current game state, is used to predict the desired experience of the player. Both the player model and the predicted desired player experience steer the adaptation and generation engine that is responsible for adapting and generating personalized video game elements.

Regarding the game components that can be adapted, these include but are not limited to:

- Game worlds
- Mechanics
- AI / NPC
- Narratives
- Scenarios / quests

Given the focus of this thesis on game narratives, this review of the state of the art will only target works done on the field of game adaptivity regarding narratives and scenarios / quests, which, in video games, are also part of how a story is told (see section 2.2.2). For more works that adapt other game components, please refer to the survey done by Lopes and Bidarra [Lopes and Bidarra, 2011].

The architecture proposed in figure 2.1 uses player actions as the input for the player modeling, which leads to an adapted game based only on the player's gameplay and decisions. Although some data about the player's emotions and state of mind can be derived from their gameplay and decisions (e.g. through approaches such as the one used by [Hernandez et al., 2015]), other kinds of input can be used to model the player's state of mind, for example, answering a questionnaire before being subjected to the experience (approach used by [Peng et al., 2018]). In section 2.3, a deeper analysis will be done regarding player modeling.

The sequence of actions in figure 2.1 can happen once or be the main body of a loop: this is called offline adaptivity and online adaptivity respectively. When using *offline adaptivity*, the adaptation and generation happens once, before initiating gameplay. One example is generating quests that compose the storyline of the game before being played, like the work of Li and Riedl [Li and Riedl, 2010]. *Online adaptivity* is when a game adapts its elements in real time, during the entire gameplay session. Hernandez et al. [Hernandez et al., 2015] use this approach to continuously adapt the plot to the players during the gameplay. The advantage of using online adaptivity is clearly the continuously adaptation of the game to the player, meaning that the game will always have the most recent model of the player, thus being able to better adapt to them in each circumstance. However, this continuous adaptivity can hurdle or even make impossible the creation and management of a continuously changing narrative that is still coherent. This results in the changes made to the narrative to be minimum, to minimize lack of coherency, which in turn leads to players not feeling the impact of their actions and choices, as reported by Hernandez et al. [Hernandez et al., 2015].

In conclusion, game adaptivity can be powerful enough to create unique and personal game experiences, by adapting game content to the players. The next step is figuring out how to adapt the content, more specifically, narratives, to each player. To that end, a greater knowledge about narratives is needed, in particular, about video game narratives.

2.2 Narratives and Video Games

Not all games tell stories (e.g. abstract games like *Tetris*) but nowadays it is more common to see games with intricate narratives than games without them. We now live in a world where games get scripts that could pass as blockbuster movie scripts and well-known Hollywood actors are cast as the main characters of the game, like *The Last of Us* [Naughty Dog, 2013] and the 2018 entry of *God of War* [Santa Monica Studio, 2018]. Good stories in games have been present since the beginning but the amount of good stories in video games is increasing and video games are rapidly turning into one of the favorite mediums of storytelling, next to books and movies. In fact, we can see more and more people being drawn into games just for their story as opposed to their gameplay, as well as games focused mainly in storytelling instead of gameplay like *Gone Home* [Fullbright, 2013] and *What Remains of Edith Finch* [Giant Sparrow, 2017].

This section presents and examines the main characteristics of game narratives and their role in video games as described by different authors through time.

2.2.1 Distinction between stories, plots and narratives

It is important to have a clear understanding of what is a *story*, a *plot* and a *narrative* before discussing further. Barry Ip [Ip, 2011] gathers from various authors the difference between stories, plots and narratives.

When telling someone "what happened today", we are telling them a **story**. A story is a sequence of events involving entities [Porter Abbott, 2014], that may be fictional or real. It has a beginning, a middle and an end.

The **plot** is the set of events of the story [Heath, 1996]. Additionally, according to Cobley [Cobley, 2001], it is also how these events are linked to each other. So, the plot is the set of events of the story and the causality between events. The plot holds what happens in each event and why it happens.

The **narrative** is the representation of the story [Porter Abbott, 2014]. Even though the time and the sequence of events is tightly bound to a story and the causation and links between events are encapsulated in the plot, the narrative influences decisions such as what weight and length each event should have and in which order should the events be presented [Ip, 2011]. Thus, a person can say that each telling of a story can be defined as a narrative and is arguably unique.

In conclusion, the story is the sequence of events, the plot is what happens in each event and what causes and links events together and the narrative is how the story is told and presented to the audience.

The defining characteristics of a narrative vary between different mediums - films, books, games, and so on - even though the story and plot remain the same. In the next section, we will be focusing on video game narratives and their characteristics.

2.2.2 Narratives in Video Games

In the past, game developers described game narratives as something non-interactive and merely there for exposition of the game story. Adams and Rollings write that narrative is the part where "you as the designer and author tell the player things without letting him do anything" [Adams and Rollings, 2006]. This view on narratives results on the belief that the more story-heavy a game is, the less the player is interacting and playing, and that means the game fails at its main role of being *playable*. This opinion might have been born due to some game designers considering that video game stories are only presented in non-interactive forms such as cut-scenes, pre-recorded dialogues and monologues, text boxes, captions, and so on [Wei, 2011].

This opinion was not shared by all authors at the time. Jenkins takes into consideration both interactive and non-interactive elements of narrative and divides the narrative content in two types: embedded narratives and emergent narratives [Jenkins, 2004]. Embedded narrative is the pre-generated content by the developers or writers, while emergent narrative is the content generated on the fly by the players. For example, the story of *Skyrim* [Bethesda, 2011] in a nutshell is about the main character, who is a Dragonborn, and their quest to defeat the dragon Alduin the World-Eater; this is the game's embedded narrative. Being an open world role-playing game (RPG), the main character's adventure is highly customizable, where the player controls the order of the adventures and quests, the time they spend on each city and even the main character looks and love interests, for example; this is the game's emergent narrative, created by the player. Although at a first glance embedded narratives seem to be the only ones which the creator of the game can control, emergent narratives, although not as easy, are also controllable, to an extent. Another *Skyrim* example, the cities and locations are all placed strategically in the map, meaning that locations which have easier quests and monsters will be closer to the player's starting point of the game. The first time the player steps out into the world, they are presented with a linear path that leads them to the first city. This path is a suggestion, but most players end up visiting that city first anyway, which contains important introductory lore to the game. This shows that game developers can still influence and guide to some extent the emergent narrative, even with a high degree of freedom of the player. Unfortunately, this has some setbacks, such as not being feasible for all game scenarios and that it can be very hard to write stories that take into account this amount of freedom. Since the game's emergent narrative is so hard to control and depends heavily on the actions of the player, it will not be considered as a possible subject of adaptation in this thesis.

This classification proposed by Jenkins is still a bit simple towards embedded narratives. We are left to think that the only way the game designer has of conveying their embedded narrative is by letting the player sit through non-interactive sequences. That is clearly not true, and several games already exist where the whole story is told through interactive sequences. For example, *Untitled Goose Game* [House House, 2019] never stops being interactive and the player is still able to learn the main character motivations, by seeing the main character's actions and the setting of the game.

Wei then proposes a new definition for embedded narrative in a game. By taking into consideration that games are visual interactive narratives, where the story is not only "told" but also "shown" and "interacted with", he writes that embedded narratives are not only sequences of shown events, but also game artifacts, that are able to tell mini-stories or back-stories [Wei, 2011]. Game artifacts can range from game objects and their disposition in order to tell a story, to game characters, that through dialogue, are able to narrate stories to the player. Note that the player can interact with these game artifacts, be it pick a game object or change the camera position to take a better look at it, or choose which NPC they want to talk to. With this new definition, we can (and should) take into consideration the possible actions of the player as part of the embedded narrative [Wei, 2011].

From this, it can be concluded that game narratives are composed of both interactive and non-interactive parts. In addition, the game designer is able to tell the story (embedded narrative) through exposition with game narrative elements such as cut-scenes, text boxes, pre-recorded dialogues [Besmond, 2019] and make use of the game space and artifacts, letting the player see and interact with the story [Wei, 2011].

2.2.3 Adaptive Narratives in Related Works

Although not as popular as difficulty adjustment adaptivity or game world generation [Lopes and Bidarra, 2011], there has been a considerable amount of works in the field of game adaptivity related to narratives. This section will review and summarize some of the approaches taken by different authors to adapt narratives, mainly what components of the narrative they decided to adapt and how they managed the narrative elements in the frameworks developed. A comparison table containing the focused characteristics of the frameworks or algorithms analysed in this section can be seen in table 2.1.

In 2007, Thue et al. created PaSSAGE, a system that automatically creates the player model based on his preferred playstyle and uses it to adapt the content of the story to the player [Thue et al., 2007]. PaSSAGE chooses from a library of events the encounters to present to the player. The events contained in the library were previously annotated with which player types they should be presented to, by the author of the story. So the encounters that the player experiences have coherency with each other, all the possible encounters are grouped into the different phases of a common used narrative structure for video games, the Monomyth from Joseph Campbell [Campbell, 1949]. This way, PaSSAGE can offer the player different encounters to choose from, still forming a believable and coherent story.

Other works in the following years adapt the same narrative elements (the plot of the story) and in a manner inspired by Thue et al. For example, PACE, the framework developed by Hernandez et al. [Hernandez et al., 2015], also has a library of story events annotated with player preferences according to their player model.

Back in 2007, Barber and Kudenko took a different approach. They also create a system capable of generating stories, but based on dilemmas to create dramatic tension [Barber and Kudenko, 2007]. The system is composed of a knowledge base and a narrative generator, that, according to the knowledge base and a user model, chooses various substories for the player to experience. The narrative generator is responsible of creating an interactive drama, created through a series of substories, each culminating in a dilemma. Each dilemma needs preconditions met; the satisfaction of those preconditions and the final dilemma become then a substory. So, instead of choosing different pre-made events, this system creates the events based on a knowledge base, creating several substories that culminate in dilemmas and are linked to each other. The story designer only has to feed the knowledge base the characters, their relations, locations and actions. One of the drawbacks of this system is that it will only work well in generating stories of genres that emphasize dilemmas between characters, with stereotypes and cliché [Barber and Kudenko, 2007].

Related Works	**Adapted Plot**	**Adapted other Narrative Elements**
PaSSAGE [Thue et al., 2007]	Yes	No
PACE [Hernandez et al., 2015]	Yes	No
Dynamic Generation of Dilemma-based Interactive Narratives [Barber and Kudenko, 2007]	Yes	No

Table 2.1: Related works in game narrative adaptivity

2.2.4 Conclusions

Narratives in video games are composed of several elements, both non-interactive and interactive. Stories can be told using this media through exposition and non-interactive means but also using the game space and artifacts, in a more interactive way. Most works done in adaptive narratives only adapt the narrative by choosing which events and in which order they should be presented to the player, so they only alter the plot element of the narrative. Some authors chose to generate substories linked to each other to create an entire story [Barber and Kudenko, 2007], while others preferred creating frameworks that choose which events or quests the player will experience [Thue et al., 2007, Hernandez et al., 2015]. From the works reviewed, none decided to adapt other narrative elements, like the spaces where the game takes place, or the setting, or the general mood of the game.

2.3 Player Modeling

Most work done in game adaptivity starts with defining a player model (figure 2.1). Houlette defines a player model as "essentially, a collection of numeric attributes, or *traits*, that describe the playing style of an individual player" [Houlette, 2003]. These traits represent the player's behavior in the game. For example, if we are adapting to each player the amount of weapon consumables that the game should drop throughout gameplay, the traits captured of the player could be "NumberOfGrenadesUsed", "NumberOfArrowsUsed" and so on. With these traits, the game adaptation engine could either try to favor the player by giving them more of their most used consumable or challenge them, by limiting the drops of that same consumable. So, a good player model should capture the main traits of the player that are most important to the type of "adaptation" that we need for the game.

Sharma et al. [Sharma et al., 2010] state that there are two type of player modeling approaches:

- approaches that take direct measurements of physiological, psychological or behavioral data, which they call *Direct measurements*
- approaches that infer information about the current player based on a previous collection of player features or player actions, which they call *Indirect measurements*. In this thesis, these kind of measurements will be referred to as *Inferred measurements*

Direct-measurement approaches for physiological data require that the player is connected to bio-sensors. An example of *direct-measurement* approach is that of Caminha [Caminha, 2017], where he uses biofeedback, such as measures of arousal and valence of the player, to control enemy spawns and guarantee that the player would not show signs of boredom. It is possible to recognize emotion using bio-sensors, such as the work done by Haag et al. [Haag et al., 2004], but it is out of the scope of this thesis.

Since the player model used depends on what is being adapted in the game, there are no "off the shelf" models; most models used in game adaptivity are custom made. As a result, this section will mostly be gathering works done in game adaptivity related to game narratives, some already seen in section 2.2.3, and analyzing what kind of player models were used. It is also important to remember that the objective of this work is to adapt the narrative to the emotions and feelings of the player, so the player model that will be built needs to focus on those emotive traits. Models that do so will be highlighted in the following section. A comparison table containing the summary of the focused characteristics of each model can be found in table 2.2.

2.3.1 Player Modeling in Related Works

This section starts with PaSSAGE, the system created by Thue et al [Thue et al., 2007]. The main objective of this system was to create game stories that had a greater entertainment value to players, hence the choice of preferred playstyle to model the players. As the basis for their model, they used the player types from Robins Laws' rules [Laws, 2002] which include Fighters, Power Gamers, Tacticians, Storytellers and Method Actors. The model was implemented as weights for the five player types, in which the higher the weight the higher the belief of the model that the player belonged to that type. The model is updated throughout gameplay, based on the player actions and choices. The model used for this work seems to be simple to implement and produced good results, showing that players who played the adaptative version found the game more fun [Thue et al., 2007]. However, this model does not contain the affective makeup of the player, meaning that the player's emotions were not considered in the narrative's adaptation.

In 2010, Sharma et al. created a Case-based Drama manaGer (C-DraGer) [Sharma et al., 2010], that similar to PaSSAGE, adapts the game story in order to improve the player's experience, by picking story events that are likely to be more interesting to the player. To that end, they created a player modeling module (PMM) that constantly builds and maintains a player model for the current player. The player model contains the player's characteristics, extracted by his actions during gameplay. The different actions that the player could take were categorized to facilitate comparisons between players. Then, the PMM calculates the "numeric interestingness" of the next possible story event to the player, given their player characteristics. This "interestingness" is calculated based on other player's opinions, so the PMM works on the assumption that similar players will have similar interests. As with PaSSAGE, this manager showed good results, as the tests done indicated that it improved the player's experience, especially for inexperienced players. The downside is that C-DraGer was used in a text-based interactive game, where the player actions were limited, and thus, easier to group and compare between players. This approach is harder to do with other kinds of games. Also, this model does not contain any information regarding the player's emotions.

Also in 2010, Li and Riedl presented an offline algorithm to plan the story of role-playing games [Li and Riedl, 2010], by choosing which quests should the user see before they play the game. The objective was the same as both works above, that is, to improve players' experience. The approach used to model the player was the same as the one used by Sharma et al. but with two additions: similarity metrics between quests were used to calculate the "numeric interestingness" of quests that were not yet experienced by users and a novelty metric was used to calculate which quests would be appropriately novel to the player (so the player would not experience boredom or unpleasant surprises). Thus, a weighted sum of interestingness and novelty is used to choose which quests should show up in the player's adventure. Again, this approach also showed good results but the authors do not describe in great detail how they built the player model, leaving doubts regarding how they captured the player's characteristics before playing the game.

In 2015, Hernandez, Bulitko and Spetch implemented an "AI experience manager" called PACE [Hernandez et al., 2015] and tested it using the game "iGiselle". PACE predicts the emotional response of a player to certain events and uses these predictions to keep the player on a "target emotional curve" [Hernandez et al., 2015]. Contrary to the three works presented above, the main objective of PACE is not to improve the players' experience, although the results show that it is one of the consequences of the adaptation. The user model used by PACE is similar to the one used by PaSSAGE, in that it has information of the inclinations of the player towards certain playstyles. For example, the playstyles used for the "iGiselle" were *storytelling*, *showing off* and *being modest*. Each possible player action in the story changes these values. How it affects them is chosen by the author of the story. The model is then used to predict which story end-goal would be more desirable to the player, based on probabilities provided by the author. These desirability values are used to predict the emotional response of the player for the next possible story events. Finally, PACE chooses the event in which the player will have a predicted emotional response in agreement with the emotional response that the author wants the player to have in that moment of the story. Although showing inconclusive results, this is a clever implementation using a user model that was initially thought out with the objective of increasing the player's experience; its implementation does not rely on a lot of external data given by the player. Consequently, it relies heavily on author assumptions of the desired goals of each player and the probability of reaching that goal. These manual calibrations of the system are not properly validated and may be one of the reasons why, according to the authors, PACE showed inconclusive results. Yet, this implementation shows promise as one that tries to adapt the story taking in consideration the player's emotions.

In 2018, de Lima, Feijó and Furtado tried to implement a better player model for adaptation of video game narratives, since the ones implemented for previous works were based on simplified models of player archetypes and failed to represent "blended behaviours" [de Lima et al., 2018]. Their objective was to build a player model that could handle the complex and volatile nature of human behavior. To that end, the player model implemented was constantly updated using *time* *windows* during gameplay and it was based on the Five Factor Model (also known as "Big Five") [Goldberg, 1990], which models human personality. To extract behaviours from the personality model, an Artificial Neural Network was used, trained with statistics extracted from gameplay. To determine the player's personality traits, the authors developed a "Big Five Game Inventory", which comprises of 10 different questions related to the setting of the game, in which the decisions made by the player are equivalent of answering BFI-10 questions [Rammstedt and John, 2007]. With this method, the player's Big Five factors are determined through 10 questions that are fast to answer and are related to the game. According to the authors, the resultant player model, which models behaviours and not personality, helps game developers to prepare a game that dynamically adapts to the behavior of the player throughout the game. Although the player model implemented does not cover player's emotions, the "Big Five Game Inventory" could be used as inspiration for the method of retrieving player's emotions before gameplay in the context of this thesis' work.

All the methods reviewed above were *inferred-measurement* approaches, where the player information was inferred from player actions. From this review, there is little to no work done on video game narrative's adaptation using *direct-measurement* approaches to collect psychological or behavioral data, without the use of bio-sensors. There are, however, works where *direct-measurement* is applied, albeit not being in the video game field. The work of Peng et al. [Peng et al., 2018] adapts an animation to the viewer and is worth mentioning, since it uses *direct-measurement* approaches to adapt the narrative to the emotions and mood of the player, with the intuit of creating personalized emotional narratives that induce self-reflection. Also, animation is one of the core components of video games and some of its rules and fundamentals are interchangeable. The way the authors of the paper adapted the narrative of the animation was by collecting mood and behavior data for one week for each user. The collected data is directly mapped into character animations and perceived emotions, lighting, environment colors and overall mood. It also affects the order of the story events and the ending of the story. The results showed high engagement and self-reflection behaviours by most viewers, which are also the desired results of this thesis. The kind of data gathered from the users and the way it was shown in the animation can be of use to this thesis main objective. But, since this was done in the non-interactive narrative field, it might be harder to apply the same direct mapping technique between player's mood and narrative elements in an interactive narrative, due to the unpredictability that naturally arises from user interaction. The users still have to feel that they have freedom to explore by themselves, or else it will feel more like a movie than a game.

2.3.2 Conclusions

From what was gathered, most works done in game narrative's adaptation are done using *inferred-measurement* approaches. One of the reasons might be because these approaches are able to extract player features more naturally, that is, the only thing the player has to do is play and their characteristics are extracted through their actions in the game. *Direct-measurement* approaches can be seen as more intrusive and feel less natural to the player. The player has to answer questionnaires or use sensors **besides** playing the game, which can break immersion. On the other hand, most

Related Works	**Adapted interactive narratives**	**Type of measurement**	**Adaption based on player's...**	**Result was an affective narrative**
PaSSAGE [Thue et al., 2007]	Yes	Inferred	Preferred playstyle	No
C-DraGer [Sharma et al., 2010]	Yes	Inferred	Characteristics and estimated interest in the story events	No
Offline planning approach to game plotline adaptation [Li and Riedl, 2010]	Yes	Inferred	Characteristics and estimated interest in the story events	No
PACE [Hernandez et al., 2015]	Yes	Inferred	Preferred playstyle and desired emotional response	Inconclusive results
Player behavior and personality modeling for interactive storytelling in games [de Lima et al., 2018]	Yes	Inferred	Personality (modeled after the Big Five Model)	No
A trip to the moon: Personalized animated movies for self-reflection [Peng et al., 2018]	No	Direct	Mood and Behavior Data	Yes

Table 2.2: Related works on player models for narrative adaptivity

inferred-measurement approaches are only used to adapt the narrative to the personality [Thue et al., 2007, Sharma et al., 2010, Li and Riedl, 2010] or behavior [de Lima et al., 2018] of the player, with the main objective of increasing the player's experience and fun. Little to no research was found in adapting a game's narrative to the player's emotions and moods with the objective of inducing self-reflection behaviours in the player, like the work of Peng et al. [Peng et al., 2018].

Two possible approaches to reach this thesis goal can be taken from this review:

- use an *inferred-measurement* approach and instead of modeling the player according to player types or according to the "Big Five" model, research into affective models [Hudlicka, 2008] and try to determine if, from player actions, a correct affective model can be obtained. This approach has the drawback of not knowing if player actions can accurately infer player emotions

- use a *direct-measurement* approach similar to the one of Peng et al. and, through the results of a questionnaire, directly map those results to narrative elements or to story events. For example, have a set of key story events, and each player would see the events that better reflect their emotional profile. This has the drawback of only being tested in non-interactive narratives, so these key events would have to take in account possible user interaction

2.4 Self-reported Emotion Questionnaires

To adapt the narrative to the players' emotions, some means of acquiring their emotions had to be found. To this end, some research was done on self-reported emotion questionnaires.

Self-reported emotion questionnaires are usually given out by psychologists to their patients, as a means of assessing the patient's emotions [Riopel, 2020]. Three questionnaires were considered: the Positive and Negative Affect Schedule (PANAS)[Riopel, 2020], the Profile of Mood States (POMS) [Searight and Montone, 2017] and the Discrete Emotions Questionnaire (DEQ) [Harmon-Jones et al., 2016]. The three questionnaires are similar in practice, differing only in what they measure. They consist of different words, each corresponding to an emotion. The patient has to rate each word using a Likert scale, according to how they feel. Then, the score of each measure is calculated by the sum of the rates given to the words that correspond to that measure. This means that each questionnaire returns numerical measures, which is ideal as an input for the framework.

As it is a questionnaire based on words, the native language of the patient and the language in which the questionnaire is written can influence the results, as a person that is not so proficient in a certain language will probably misunderstand the meaning of certain words. Another weakness of these questionnaires is that it uses self-reported values, which are subjective and can be over or under estimated.

What each questionnaire measures, as well as their individual advantages and drawbacks can be seen in table 2.3.

Out of the three questionnaires, the DEQ is the one that seems the most useful for the objective of this thesis, as it measures the value of 8 emotions and not only the value of positive and negative affect. With the granularity of 8 emotions, a better narrative can be crafted for each player, as opposed to only using the values of the player's "good" emotions and "bad" emotions.

2.5 Summary

This research process showed that game adaptivity techniques could be used in order to adapt game narratives, which several works have already done. Game adaptivity often starts by modeling the player and then, according to that model, adapting the contents to the player. Most game elements can be adapted, one of them being the game's narrative. In order to adapt the game narrative, one needs to know more about how it works and its unique properties. Game narratives share some core elements with narratives in other media, but are specially unique thanks to their balance of interactive and non-interactive components. In fact, developers can tell stories in games, not only through direct exposition, like films and animation, but through the game's space and artifacts. However, the works reviewed only adapt the plot of the story, without taking into consideration the game's spaces, setting or even its general mood, which are also part of a game's narrative. In addition, the player models used for game narrative adaptation only take into consideration the player preferences according to their type or the player's behavior, extracted through player

Questionnaire	Measures	Advantages	Drawbacks
PANAS	Positive and Negative affect	- Widely used, having been translated to most languages (including Portuguese) - Has a shorter more concise version (PANAS-SF)	- Only returns two measures: the sum of positive and the sum of negative emotions
POMS	Positive and Negative Mood	- Also widely used, has Portuguese translation - Has a version for adults (over 18) and another for adolescents (13 to 17 years) - Also has a shorter version	- Same problem as PANAS, only returns two measures: the sum of positive and the sum of negative emotions
DEQ	Anger Disgust Fear Sadness Happiness Anxiety Desire Relaxation	- Measures each emotion individually, with a range of 4 (not feeling that emotion at all) to 28 (feeling that emotion in an extreme amount) - It is possible to use only part of the questionnaire, depending on which emotions to measure	- Quite recent and not as widely used as the other two, only exists in English - Fairly long, has the participant rate 32 words

Table 2.3: Self-reported emotion questionnaires

actions. Almost no works try to adapt the game's narrative to the player's emotion and state of mind, and even less have the objective of inducing self-reflection behaviors in the players. This is one of the reasons why this thesis' results can pave the way for exploring new game design techniques based on reflective designs, with the addition of new forms of generating tailor-made experiences in interactive media.

Therefore, the main objective of this work is to build a framework capable of adapting game narratives to the emotional state and behaviors of the player, so that players can feel reflected by the game and have more engaging experiences. The next chapter presents the proposed architecture for such a framework and details its implementation.

Chapter 3

Proposed Solution for the Framework and its components

Adapting a video game's narrative to the player's emotions can lead the player to feel like the game reflects their emotional state, which can make playing that video game a more meaningful and engaging experience. In this chapter, we are going to propose a framework and supporting architecture, and describe its implementation. The framework's objective is to generate narratives based on emotional profiles, to make gaming experiences more enjoyable and engaging.

Figure 3.1 illustrates a simplified overview of a system where the framework would be used. To be able to adapt the contents, the framework needs two inputs: the player's characteristics and the narrative as a whole, annotated with adaptivity information. The output of the framework should be the components of the narrative that are chosen for that player to see. These chosen components are then input to the game that will present the adapted narrative.

This chapter starts by describing which decisions were made in the process of building this system (section 3.1). Then, it presents the framework's conceptual architecture and its components in section 3.2. After, it specifies how the narrative is represented in the framework (section 3.3). After a brief overview of tools and technologies chosen (section 3.4), the implementation of each component of the framework is seen in sections 3.5, 3.6 and 3.7.

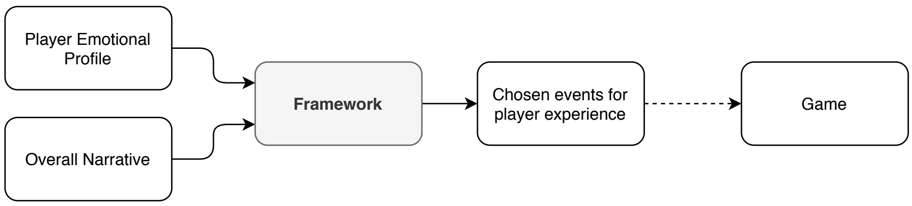


Figure 3.1: System overview

3.1 First Steps towards Game Narrative Adaptation

Before building the framework, some decisions had to be made. This section presents such decisions in an ordered view, regarding genre and style (3.1.1), elements to adapt (3.1.2), and the adaptation criteria (3.1.3).

3.1.1 Video game genre and narrative style

As defined in section 2.2.2, the concept of story is not the same as the concept of narrative: the same story can be told in different ways, thus having different possible narratives. This is no different in the world of video games, and, even in games inside the same genre, the ways of telling their stories might vary significantly from each other. For example, Dark Souls [From Software, 2011] and Devil May Cry [Capcom, 2001], two game series that fall in the action genre, have very different ways of communicating their stories. One gives very little information about the world and its characters directly to the player, choosing to tell its story gradually mostly through environmental storytelling. The other has a mission system, where every mission has a non-interactive section where it tells the story (cutscene) and the rest of the mission is mostly just focused on the gameplay.

Although it would be great to build a framework that could work with any video game genre and their specificities, it would be impossible to finish such a task in the given time for this thesis. So, this leads to the first question in developing the framework: which video game genre and narrative style should the framework be built for? The chosen genre was RPG with a narrative style similar to the story-driven games made in the RPG Maker engine [Enterbrain, 2015]. Some notable games made with this narrative style include Off [Mortis Ghost, 2008], Ib [Kouri, 2012] and Pocket Mirror [AstralShift, 2016]. These games are characterized for being 2D games where the main goal of the game is to unravel the story. The gameplay tends to only include interaction between items and characters, but can also include escape sequences, puzzles, mini-games and battles. Additionally, most of these games' stories have multiple routes and endings. As I was most familiarized with this game genre and it seemed the most appropriate for the thesis' objective, this was the first decision taken for building the framework.

Having the genre picked, the next step is to choose which narrative elements should the framework adapt to the player.

3.1.2 Narrative elements to adapt

As concluded in section 2.2.4, the narrative elements in a video game can be many, both non-interactive and interactive. The elements to be adapted by the framework were chosen with two constraints in mind: they would have to be characteristic narrative elements from the chosen video game genre (section 3.1.1), and the number and complexity of these elements needed to take into account the time given for implementation. The chosen narrative elements to adapt were:

- Visual elements such as:

  - Scenarios/maps
  - NPCs, PCs (playable characters) and items disposition in the map, as well as their appearance

- Textual elements such as:
  - Dialogues between characters
  - Items descriptions
  - Text in cutscenes

- The story and plot in general:
  - Which events should the player see
  - The sequence of those events
  - The type of the events - cutscene or gameplay

- In cutscenes, the behaviour (actions and positions) of the characters

- The gameplay mechanics:
  - Which NPCs and items the player could interact at the moment
  - Which enemies could be encountered by the player
  - What the player should do in order to progress in the game

These were the narrative elements chosen to be adapted to the player. The next decision to be made is how to adapt those elements to the player, by choosing which player characteristics should be the adaptation's criteria.

3.1.3 Adaptation criteria and the Player's Affective Profile

To adapt a game to its player, there needs to be some knowledge of the player's characteristics, defined in section 2.3 as player modeling. Also mentioned in that section, the player characteristics chosen for that player model must correspond to the type of adaptation being done. Since the objective of this thesis is to adapt the narrative to the player's emotions, it follows that the chosen player's characteristics were the player's emotions. But the "player's emotions" is a very broad notion, and defining it was one of the first challenges in building the framework; more specifically, which emotions, how to acquire the information about those emotions and how to quantify them in a way that could be used by the framework.

The end of section 2.3.2 contains the two possible approaches to modeling the player that were considered: an *inferred-measurement* approach, where the emotions of the player would be inferred from the player's actions in the game or a *direct-measurement* approach, where the emotions would be directly collected from the player, e.g. through a questionnaire before the game experience. The chosen approach was the *direct-measurement* approach, as it seemed to be the most feasible one, and, as opposed to the *inferred-measurement* approach, was used at least once by Peng et al.[Peng et al., 2018] for the same effect (emotional adaptation), in all of the reviewed projects in section 2.3.1.

To implement this *direct-measurement* approach, research was done on self reported emotion questionnaires, seen in section 2.4. This research on how the player's emotions could be acquired and measured led to a better understanding of which emotions could be used to model the player. So, the emotions chosen to model the player and to which the narrative is going to be adapted are the emotions measured by the Discreet Emotions Questionnaire (DEQ): **anger**, **disgust**, **fear**, **sadness**, **happiness**, **anxiety**, **desire** and **relaxation**. Throughout the rest of this thesis, this set of emotions and their values will be called the **player's affective profile**.

3.2 Framework's conceptual architecture

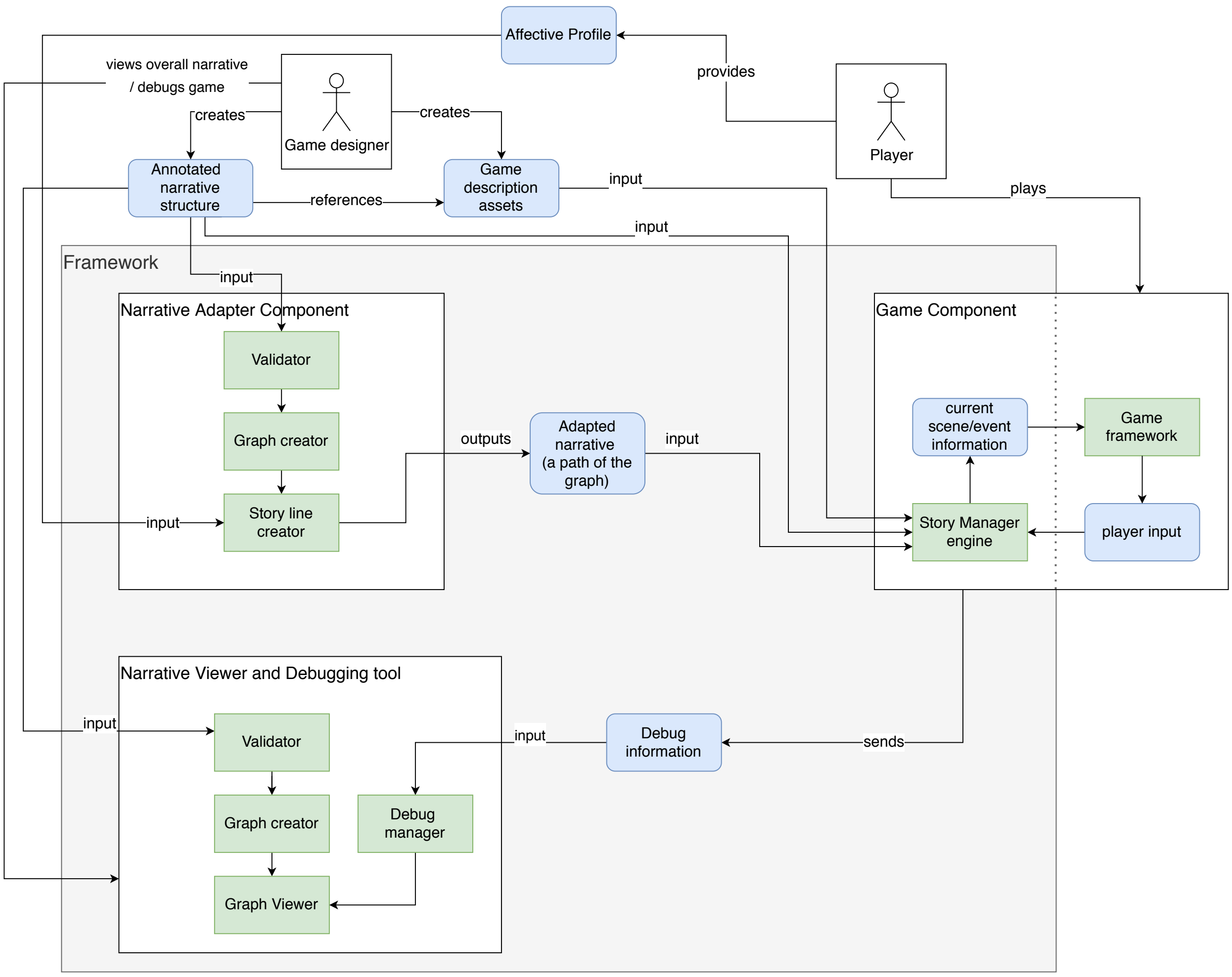


Figure 3.2: Conceptual Framework and System Architecture

With the video game genre, narrative elements to adapt, and the adaptation criteria chosen, it is possible to start designing and building all the needed components for the framework. Figure 3.2 shows a block diagram of the different components of the framework and the system where it would be inserted, as well as its users.

The framework is composed of three components: the **Narrative Adapter component**, the **Game component** and the **Narrative Viewer and Debugging tool**.

The **Narrative Adapter component** is responsible for creating an adapted narrative to the player's affective profile. As an input, it receives a narrative structure created by the game designer, which contains all the possible narratives that players can see. This narrative structure is annotated with information about which parts of the narrative can be seen by which players. This component also receives as input the affective profile provided by the player. It outputs the narrative adapted to the player. This component's workflow is composed of three steps. First, it validates the input given by the game designer (the annotated narrative structure). Then, it creates a graph from the narrative and from that graph chooses the path that is better adapted to the player. Its implementation can be seen in section 3.5.

The **Game component** is responsible for instantiating the adapted narrative and showing it to the player. It receives as input the adapted narrative created by the Narrative Adapter component and the narrative structure and game description assets created by the game designer. This component is composed of two components, the Story Manager engine and the Game framework. The Story Manager engine reads the annotated narrative structure and the game description assets created by the game designer. Then, according to the adapted narrative created by the Narrative Adapter component, it sends the information of which assets should the game framework instantiate and show to the player, depending on which point of the story the player is. The specification of this component can be seen in section 3.6.

The **Narrative Viewer and Debugging tool** is a support tool for the game designer. It allows the game designer to view the narrative that they are writing in its graph form and to debug the narrative while the game is running. This tool is detailed in section 3.7.

3.3 Annotated Narrative's Structure Specification

The game designer needs to create all the possible narratives that a player can experience. So that the framework can "understand" the narratives, a language protocol to write the narratives had to be defined. To help, a "toy example" was built, which can be partially seen in figure 3.3 (the complete "toy example" is available in appendix B).

The toy example as shown in fig 3.3 represents a game narrative that is composed of:

- Locations - the environment or the map where the events take place (e.g. a mansion, a kitchen, a forest)
- Events:

[1] Story based on Tchaikovsky's "The Nutcracker" ballet

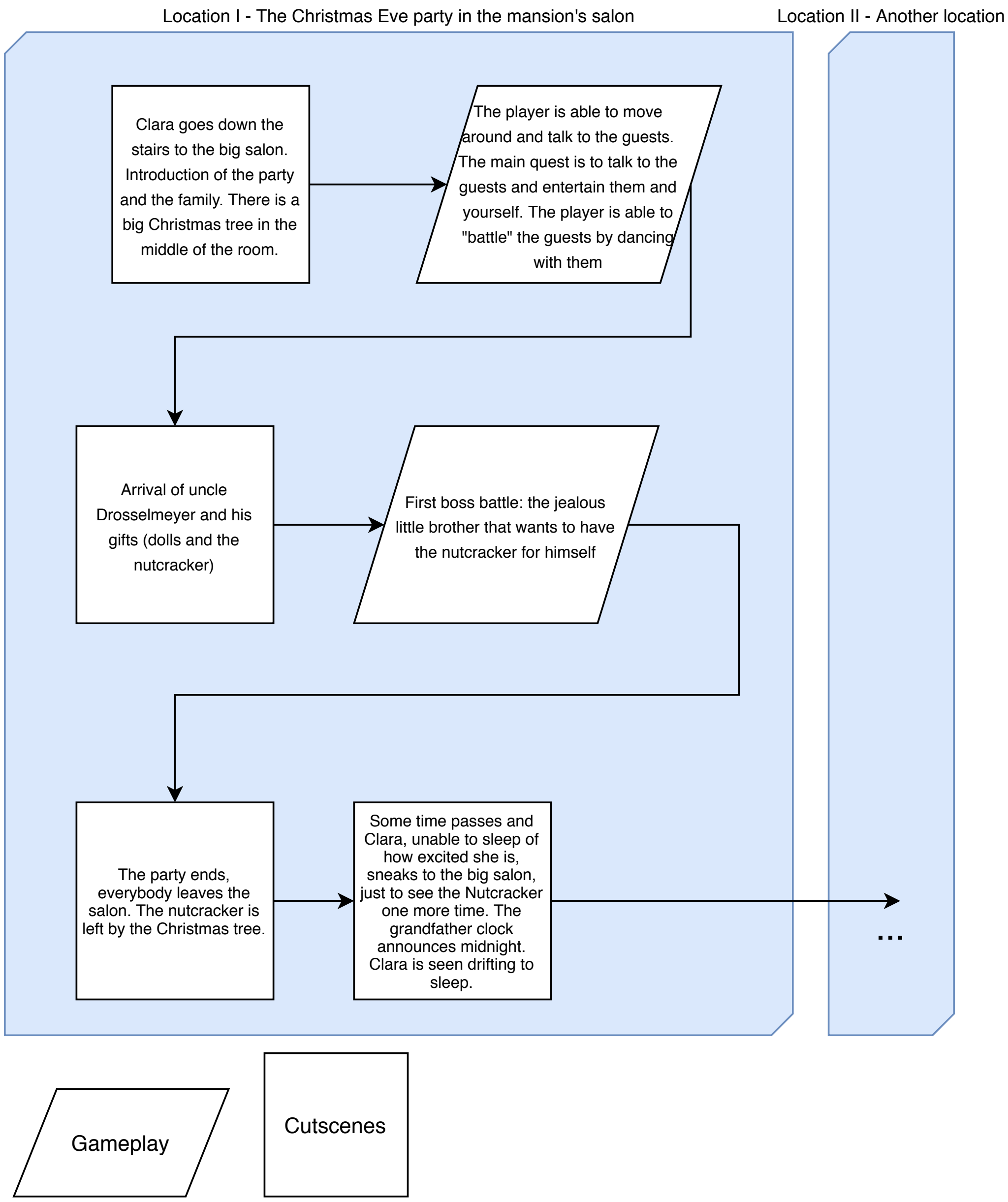


Figure 3.3: Toy example [1]

- – Cutscenes - a non-interactive sequence of events which can show conversations between characters and character actions (e.g. walking and talking)
- – Gameplay - an interactive sequence of events where the player can interact with objects or other characters, engage in battle with enemies and explore the environment

- Actors - all the characters present in the event that can act (NPCs, PCs and enemies)
- Transitions - by the end of the last event, the player transitions to another location

This example shows a linear game narrative, which means the narrative would be the same for every player. To support adapting the narrative to the player, some kind of branching needs to be introduced in the narrative (e.g. happy players will see a branch of the narrative which is different from the branch presented to sad players). The term "branching", when used in the context of video game narratives, means the game has moments where the course of the game can change, usually due to some explicit action of the player. In our context, what determines the different courses (branches) that are presented to the player is their affective profile. In other words, the game narratives created by this framework are linear in the player's point of view, but have different branches in the game designer's point of view.

Two different kinds of branching were considered: branching locations (figure 3.4) and branching events (figure 3.5). Note that, due to keeping the figures readable, the two kinds of branching are shown in separate figures, but they can be used simultaneously in the game narrative.

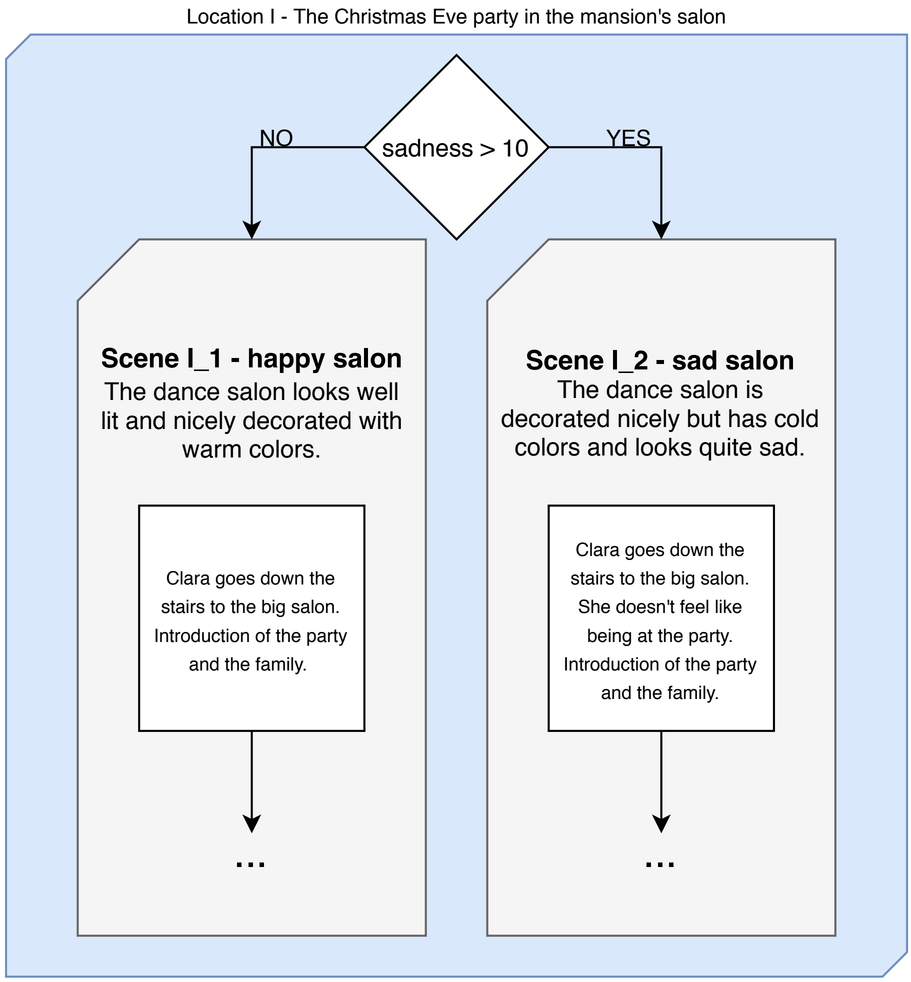


Figure 3.4: Toy example with branching locations

Figure 3.4 introduces a new concept: **scenes**. Scenes are different possible implementations of a location. As seen in the example, the same location has two scenes: one for sad players (sadness > 10) and another for the rest of the players. Each of these scenes has its own collection of ordered events. In addition, the scenes are responsible for the imagery of each location, where scene I_1 implements a happier looking salon than scene I_2. This means each scene holds information on how the location looks.

An example of the set of events of a scene can be seen in figure 3.5. While the scene is responsible for the visual representation of where the events take place, each event holds information of what happens in it, as well as characters and (interactable) items. All events have one or more possible previous events and one or more possible next events (except the starting and ending events respectively), forming an ordered sequence of events. Two events cannot exist simultaneously, one must end before the other starts. A cutscene event ends when all its actions are finished (e.g. cutscene 1A and 1B end with Clara finishing her walk to the middle of the salon). A gameplay event ends when the player has met the conditions for it to end (e.g. gameplay event 2A ends when the player has finished talking to the Grandma NPC). Apart from their order, all events are designed to be independent from each other, meaning a player action taken in a gameplay event will not influence any event after it. For example, it is not possible to open a special chest at the end of the game with a key that was found in one of the first events. That key would have to be used in the same event that it was picked from. If the aforementioned example would be allowed, this framework would also have to deal with branching caused by player actions. This is one of the limitations of this design, but the events being self-contained and independent help in making branching game narratives more cohesive, still keeping a certain degree of freedom for "mix and matching" events. It also assures the designer that players with the same affective profile will always see the same branch.

Each scene and event has a condition that rules which players will see it. These conditions, as seen in figures 3.4 and 3.5, are composed of the emotions belonging to the affective profile set. A condition can be as restrictive as the game designer wants, by combining any number of emotions from the set. Throughout the rest of this thesis, these conditions will be referred to as the "affective conditions" of an event or scene.

3.4 Tools and Technologies

Before implementing the framework, the tools and technologies were chosen. Since this thesis was made during the COVID-19 pandemic, user testing could not be done in person. So, web technologies were researched to develop the game, so that it could be available for everyone with a computer and an internet connection.

Phaser[2] was chosen as the game framework. It is an HTML5 game framework for Canvas and WebGL powered browser games. It is free and easy to learn. The current version is Phaser 3, which was the one used to develop the game. It has support for the Typescript language, the chosen language to write the game. Phaser was chosen for being a web framework but also for supporting the creation of 2D sprite-based games, with an equivalent feel of the ones made in RPG Maker.

[2]Phaser main page. https://phaser.io/

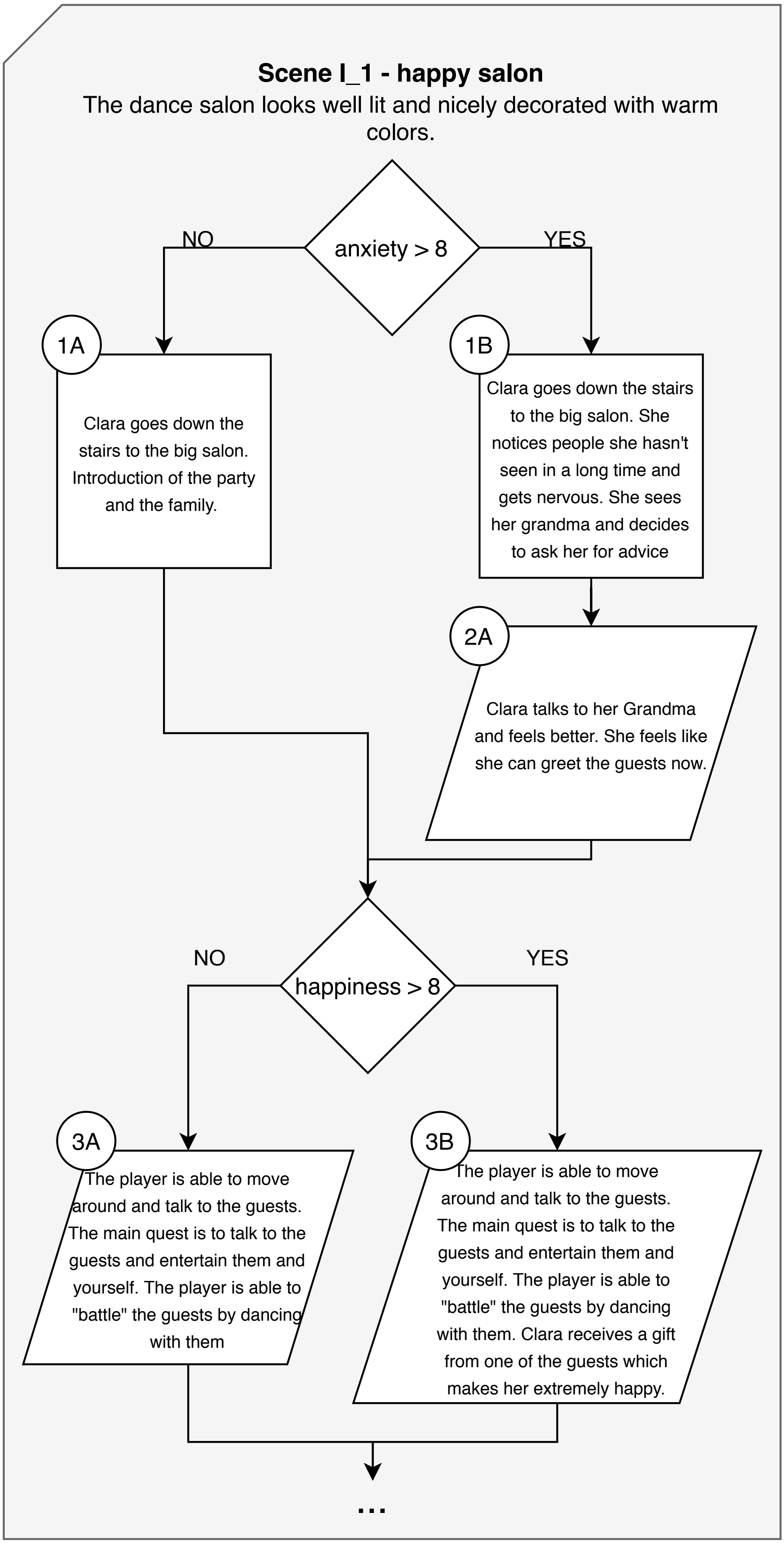


Figure 3.5: Toy example with branching events

The framework was written in Javascript. Both the game and the framework were built in a Node.js [3] runtime environment. NPM [4] (Node package manager) was used to manage application dependencies.

To bundle both game files and framework files with their respective dependencies, the Webpack[5] tool was used. Webpack is a module bundler, its main objective being bundling Javascript files for usage in a browser (usage case for the game). However, Webpack can also bundle files to create Javascript libraries (usage case for the framework). For the development of the game, webpack DevServer[6] was used to create a simple web server with live-reloading.

3.5 Narrative Adapter Component

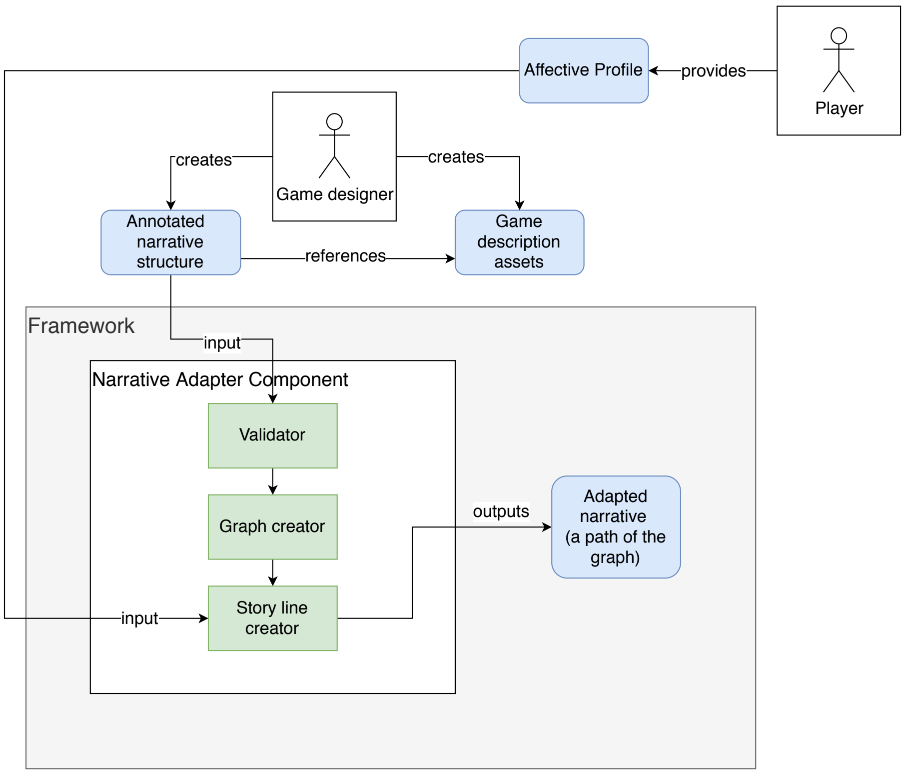


Figure 3.6: Excerpt of figure 3.2, containing only the Narrative Adapter Component

Since this component's objective is to adapt a game narrative to the player it needs two inputs: what to adapt annotated with how it is going to be adapted (the annotated narrative structure) and what it will adapt to (the player's affective profile). Figure 3.6 shows the component and both these inputs.

Both inputs are fed to the framework in the form of JavaScript Object Notation (JSON) documents since it is a notation that is easy for both humans to read and write, and for machines to

[3]Node.js main page. https://nodejs.org/en/
[4]NPM main page. https://www.npmjs.com/
[5]Webpack main page. https://webpack.js.org/
[6]Webpack DevServer main page. https://webpack.js.org/configuration/dev-server/

parse [Crockford, 2018]. And, by using JSON schema[7] to describe the format of the documents, the documents can be validated automatically before being used by the framework.

The affective profile document contains the 8 emotions that make up the player's affective profile (anger, disgust, fear, sadness, happiness, anxiety, desire and relaxation) and their values, which can range from 4 to 28. Its schema can be seen in figure A.1.

The overall narrative document contains the annotated narrative structure specified in section 3.3. The next section details the implementation of this document.

3.5.1 Overall Narrative Document

After specifying the annotated narrative structure, the **Overall Narrative** document's specification was created. The objective of this document is to only hold the information about the narrative as a whole. More specifically, it only holds the order of events and locations and the affective conditions for scenes and events. What happens in each event and how the scenes look are declared in different documents (seen in figure 3.2 as the game description assets), which are referenced by the Overall Narrative document. These assets are only parsed by the Game component, and will be explained in further detail in section 3.6.1. This kind of top-down approach to specifying the narrative helps its organization and makes it so that the framework only needs that one document as input to be able to produce which scenes and events should be shown to the player. This top-down approach can be seen in figure 3.7.

The Overall Narrative is then an object composed of a list of locations. Each location can be instanced by one scene, according to its affective conditions. Each scene contains a list of events, each event with affective conditions of their own. Each location, scene and event has a unique identifier.

The possible ordering of the scenes is also defined in this Overall Narrative document. The game designer defines which location will be the starting location for every player. The affective profile of the player decides how that location is going to be instanced, by choosing a scene (consequently, that scene becomes the first scene). Transitions between locations are defined in the scenes. This gives the freedom of writing scenarios where some transitions are only available under certain affective conditions. For example, in case the player is sad, they will be able to go from the salon to the kitchen to eat some sweets. In the happy branch, the player would not be prompted to go to the kitchen and would go to the garden instead. Additionally, each transition contains information on which events the transition becomes available.

As with scenes, events are also ordered in the Overall Narrative document. Each scene contains a list of the event ids that could be the first events to be shown to the player. Which event is the first is decided by its affective conditions. Then, every event has a list of the names of the events that could happen next. And, just like choosing the first event, the event that is next is defined by its affective conditions. An event is the last of its scene when it has an empty list of next events or when none of the next events' affective conditions match the player's affective profile.

[7] JSON Schema main page. `https://json-schema.org/`

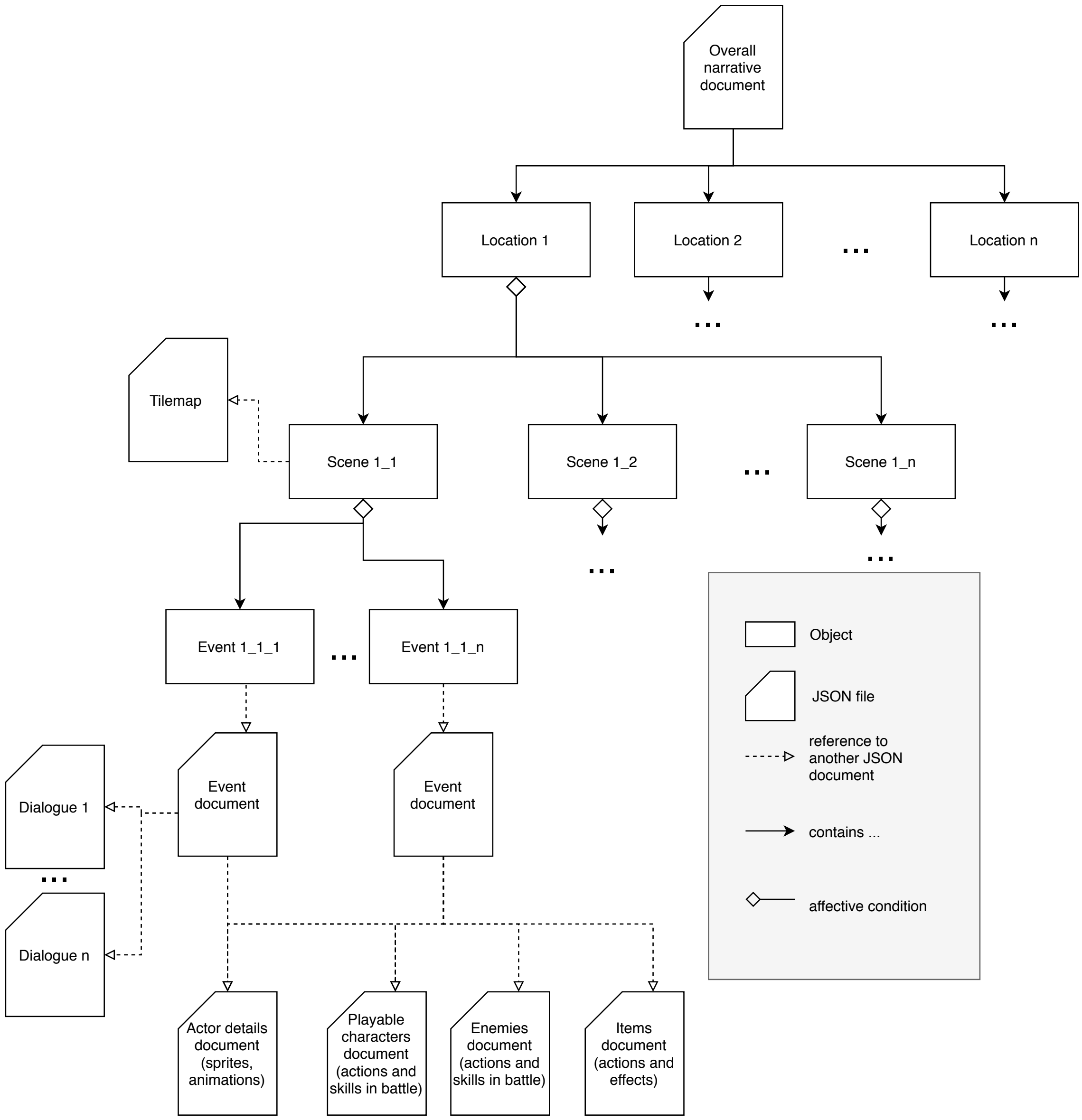


Figure 3.7: Top-down approach to the specification of the game narrative

Two or more scenes or two or more events may have affective conditions that match with the same affective profile. This can be a problem, for example:

1. Scene 1 needs the player to have sadness > 8
2. Scene 2 needs the player to have anxiety > 8
3. The affective profile of the player has sadness = 10 and anxiety = 12
4. Which scene is chosen for the player?

If the game designer wanted this player to see scene 1, this could be solved by restricting its condition by adding anxiety < 8. However, when adding more scenes, it could lead to overly complicated conditions. A more simple approach would be to add a priority to each scene and event. In the example above, scene 1 has priority 1 and scene 2 has priority 2. This also helps in building default scenes and events, for players that do not fit any of the conditions. The default scenes and events should always have the lowest priority, so that they do not override a branch with a more specific affective condition.

Summing up, the narrative is described in a top-down approach, using locations, scenes, transitions, events and actors. The **Overall Narrative** document contains this information and the relation between these elements and their affective conditions. For more details, the schema for the Overall Narrative document can be seen in figure A.2.

3.5.2 Narrative Adapter Component's Implementation

This component has two inputs: the player's affective profile and the Overall Narrative document. Its main objective is to adapt a narrative to the player. This adaptation is achieved by choosing an appropriate branch from the overall narrative for the player to see. To choose a branch, the framework goes through these steps (seen in figure 3.6):

1. Validator step: Validate the Overall Narrative document
2. Graph creator step: Parse the Overall Narrative document and create a directed graph of the story
3. Story line creator step: Navigate the directed graph and collect the scene nodes and event nodes that match the player's affective profile with their affective conditions (story line creator)
4. Output an adapted narrative graph: a chronologically ordered list of scenes, each scene containing a list of chronologically ordered events

The next subsections present the details of each step.

3.5.2.1 Validator

First, the Overall Narrative document is validated against its schema (appendix A) to detect syntax errors. This validation is done with AJV[8], a JSON schema validator for Node.js and browser. If the document has any syntax errors, the graph is not created and the game designer is notified.

3.5.2.2 Graph creator

After the validation, the document is parsed and its contents converted to a directed graph that contains the locations, the scenes, and the events. A visual representation of this graph can be seen in figure 3.8. By converting it to a directed graph it is easy to navigate the narrative and collect an ordered list of scenes and events that match the player's affective profile. The start of

[8] AJV main page. https://github.com/ajv-validator/ajv

Figure 3.8: A visual representation of the Overall Narrative graph (created by the visualization tool specified in section 3.7)

the narrative is marked with the 'Start' node. This node is connected to the first location. Each location is connected to its respective scenes, and each scene is connected to its respective possible first events. Each event is connected to its next events, and so on, thus representing the order of the story. Unfortunately, it was not possible to implement the connections between locations (transitions) due to lack of time.

The algorithm to create the directed graph is as follows:

1. Add the node 'Start' to the graph
2. Add the location nodes to the graph
3. Add an edge to connect the first location node to the 'Start' node
4. For each location node, add their scene nodes to the graph and connect them with edges to their respective location nodes
5. For each scene node, add their first event nodes to the graph and connect them with edges to their respective scene nodes
6. For each first event in the scene add all event nodes that come after and connect them, using a Depth-First Search (DFS) algorithm with the first event as the root

The Overall Narrative document is checked for semantic errors while its graph is being created. For example, if a list of next events contains an event id of a non-existing event, the algorithm skips that next event and throws an error. To detect infinite loops in the story, a DFS algorithm is run in each event node of the graph, to detect cycles. These errors are also an output of the Narrative Adapter component, alongside the adapted narrative graph.

3.5.2.3 Story Line Creator

The algorithm used to navigate the directed graph is as follows:

1. From the 'Start' node, move to the first location
2. Move to the scene that matches the player's affective profile and add it to the list
3. From that scene node, move to the first event that matches the player's affective profile and add it to the events list of that scene
4. From that event, choose the next event that matches the player's affective profile and add it to the events list of that scene. And from that event choose one of its next events, and so on, until the last event in the story is reached

Due to lack of time, the implemented algorithm does not deal with other locations other than the first location and their respective transitions, which means, the framework in its current state can only work for games that have only one location.

To choose the scene that matches the player's affective profile for a location, the scenes are first ordered in ascending order by their priority. Then, the first scene that matches the player's affective profile is chosen. If no scene matches the player's profile, no scene is added and that location will not be seen by the player. This is also done when choosing the next event from a list of next events. If no next events match the player's affective profile, then the story ends in that event for that player. Note that this can result in the player not having a game at all to play. To prevent that, default scenes and events should always be written (specified in section 3.5.1).

3.6 Game Component

The Game component is what instantiates the adapted narrative and the game itself, which the player interacts with. Figure 3.9 shows the conceptual architecture of this component.

The Game component is composed of two parts: the Story Manager engine and the Game framework. The Story Manager engine is part of the framework and delegates what parts of the narrative are being instantiated by the Game framework, which is not part of the developed framework. As stated in section 3.4, Phaser 3 was chosen as the Game framework.

As seen in section 3.5, the Narrative Adapter component only delegates the structure of the narrative, that is, which scenes and events are shown to the player, and the order of those scenes and events. What happens in each event and the appearance of each scene is only relevant to the Game

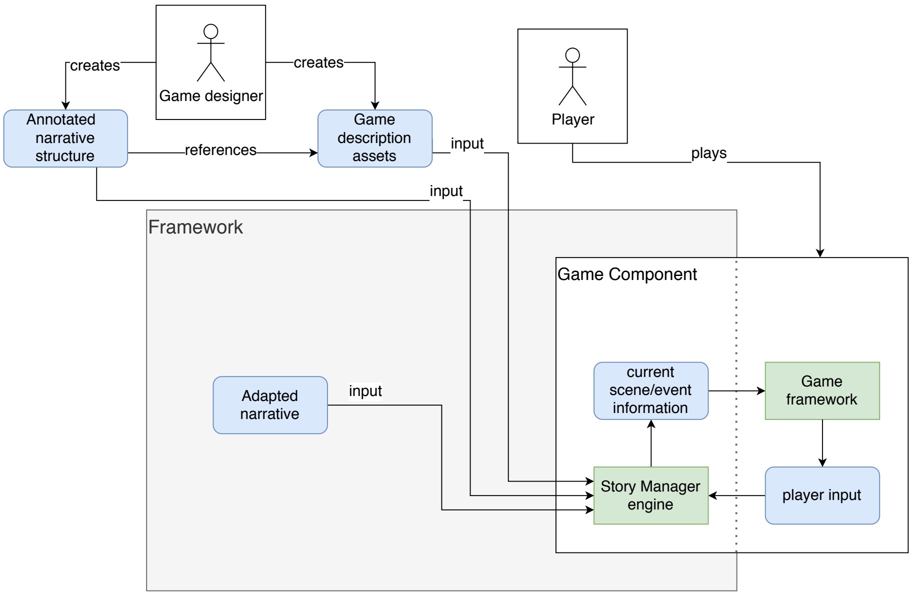


Figure 3.9: Excerpt of figure 3.2, containing only the Game Component

component, and is managed by the Story Manager engine. Figure 3.10 shows the other documents that are managed by the Story Manager engine and are important to the narrative (referenced in figure 3.9 as game description assets). An overview of those documents will be made in the next section.

3.6.1 Game description assets

These assets are represented as JSON documents, like the Overall Narrative document. They all serve the same purpose, which is to detail how the game will be representing the narrative. The game needs to know how it will be representing the scenes and what to do in each event.

3.6.1.1 Scene document - Tilemap

In the genre established in section 3.1.1, locations are represented in tilemaps. Tilemaps are a way of representing 2D game locations using small images with the same dimensions, called tiles [Itterheim and Löw, 2012]. By using different sets of tiles (tilesets) or by rearranging the tiles in the tilemap, several different scenes can be made to represent locations. An example of a tileset change can be seen in figure 3.11. The first salon looks happier, as the tileset used had more vibrant colors than the one used in the sadder looking salon. It is not represented in the figure, but the position of the items and the decorations can be changed as well, by changing the position of the tiles in the tilemap. It is possible to create completely different scenes for each location, by creating different tilemaps for each scene.

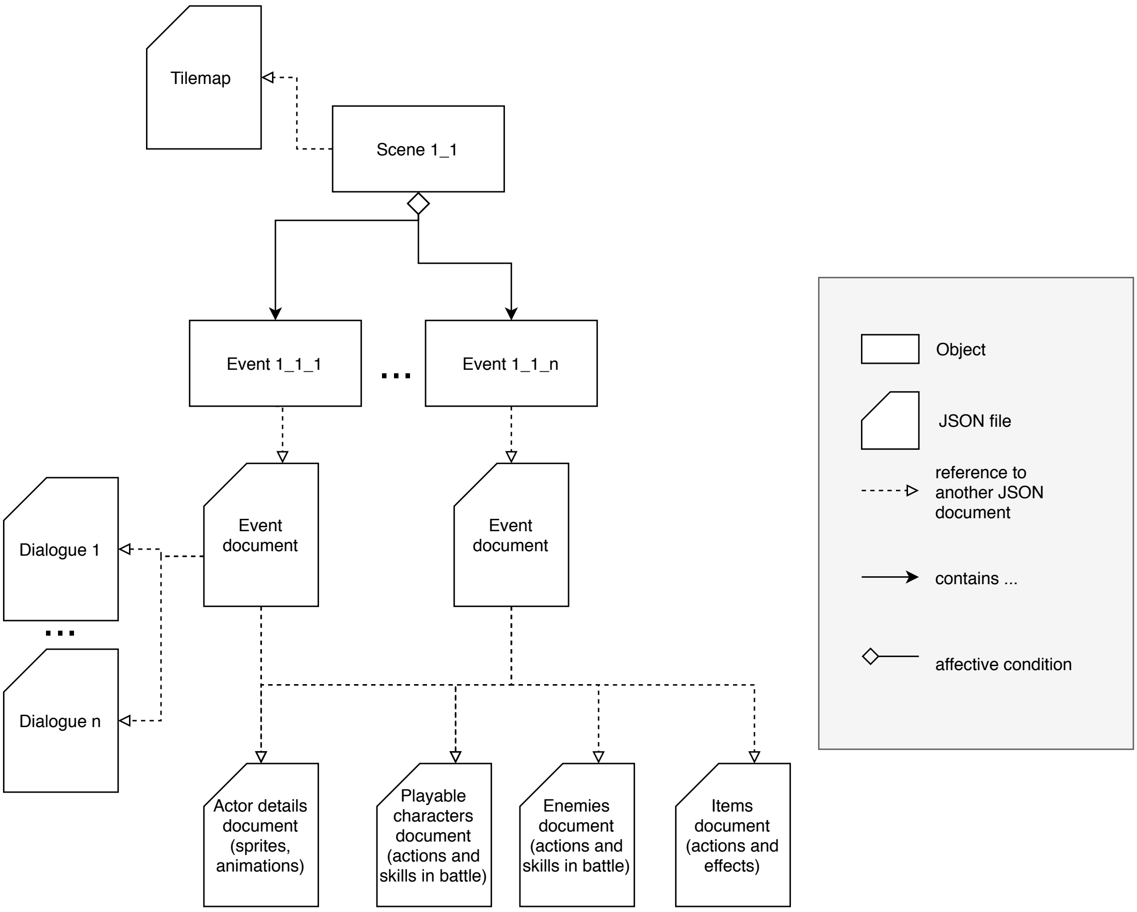


Figure 3.10: Excerpt of figure 3.7 with only the narrative documents parsed by the game

(a) Happy looking salon (b) Sad looking salon

Figure 3.11: Same location (salon), instantiated by different scenes

Tiled[9] is a popular software used for the creation of tilemaps. To organize the tiles in the map, it uses a layer system. The layers are mainly used to sort the depth of each tile (note in figure 3.11 how the tiles belonging to the decorations are on top of the floor tiles) but can also hold other properties. For example, in this game, the layers also declare if their contained tiles should collide

[9]Tiled main page. https://www.mapeditor.org/

with the player when the player walks up to them. Phaser 3 offers a parser for Tiled maps exported in JSON format, so it is easy to implement the maps in the game. In conclusion, by using Tiled, a scene can be fully described by only one tilemap JSON document.

3.6.1.2 Event documents

When designing the event documents, one needs to think of what usually happens in an event. As mentioned before, there are two types of events: cutscene events and gameplay events. For each type of event, a different document must be created.

A cutscene is a sequence of events where the player watches something unfold, usually without or with very little user interaction. They are frequently used for story exposition.

In this game, the events shown in cutscenes are composed of characters walking to a point in the map and characters talking. So, the Cutscene Event document is composed of two lists. The first contains all the actors that are participating in the cutscene. The second contains the actions taken by those actors, ordered chronologically. An action can be "walk" or "talk". Both action types have an associated actor, which will be the focus of the game camera, and specific arguments. The action "walk"'s arguments are how much the actor will be walking horizontally (x) and vertically (y). The action "talk"'s arguments are which actor is talking and what they say. Note that, for the action "walk" the game camera focuses on the actor that is walking, but for the action "talk" the game camera can focus on a different actor than the one that is talking. For more details on how each cutscene element is declared in the Cutscene Event document, its schema can be seen in figure A.3.

A gameplay event is when the player can move their character, explore the scene, interact with other characters and items, and engage in battle with enemies. As opposed to cutscene events, the sequence of the events that happen in a gameplay event are defined by the player. To determine when a gameplay event ends, it must be given an objective that can be achieved through one or more player actions.

The Gameplay Event document then contains the different elements that are part of the gameplay: the player, the enemies to be encountered, the NPCs and items that can be interacted with (interactables), and the end condition for the event. This condition is composed of flags, that can be changed whenever the player interacts with an interactable. These flags can be numerical or boolean values, and their initial values are declared in the document. Each declaration of an interactable contains which flags must be active with what values for that interaction to be possible, as well as which flags change and to what they change whenever the player interacts with them. The conditions and value changes of the flags can be complex expressions. To parse these expressions from the document, the *filtrex*[10] library is used. Consequently, the expressions must be valid according to filtrex's rules.

[10] filtrex main page. https://github.com/joewalnes/filtrex

An example of the flags system: in one gameplay event, the player must talk to three different guest NPCs and then talk to the Grandma NPC, in this order. The flags would be initialized like so:

$$numberOfGuestsTalked = 0$$
$$guest1Talked = false$$
$$guest2Talked = false$$
$$guest3Talked = false$$
$$grandmaTalked = false$$

And the end condition would be:

$$numberOfGuestsTalked == 3 \; and \; grandmaTalked == true$$

So that the Grandma NPC would have to be the last NPC to be talked to in the event, the condition for the Grandma NPC to be interactable would be:

$$numberOfGuestsTalked == 3$$

So that the player would not be able to talk to the guests repeatedly (and activate the flag change, adding 1 to the *numberOfGuestsTalked* variable), the condition for the guests to be interactable would be:

$$guest < n > Talked == false$$

Finally, every time the player would talk to a guest, the flags would change like so:

$$numberOfGuestsTalked = numberOfGuestsTalked + 1$$
$$guest < n > Talked = true$$

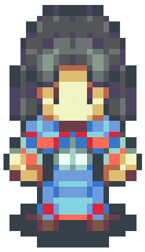
And once the player talked to the Grandma NPC, the flags would change like so:

$$grandmaTalked = true$$

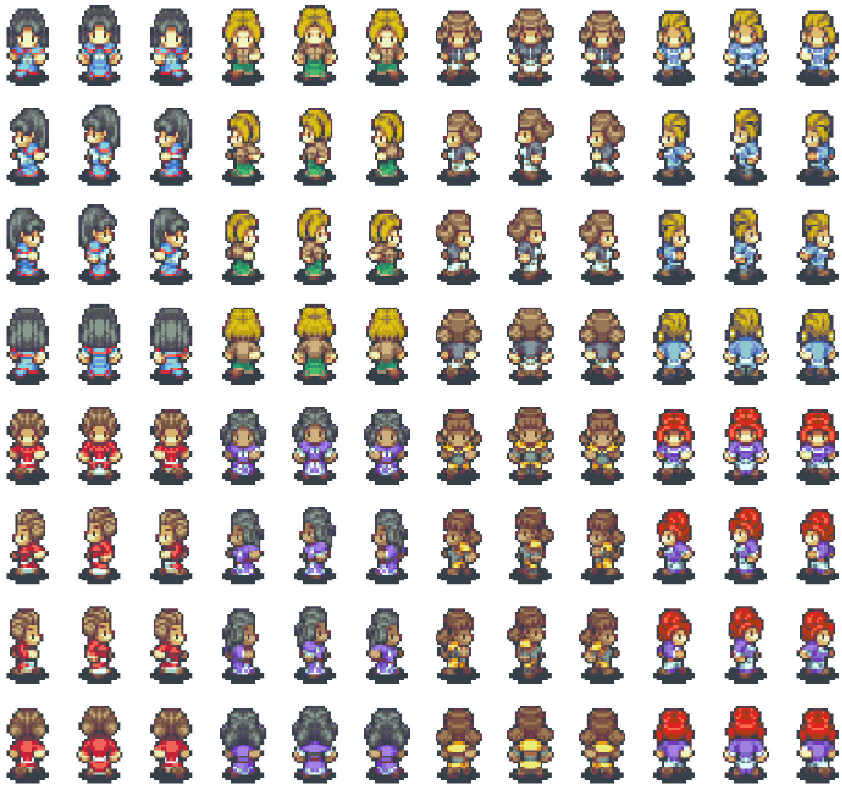
For more details on how each gameplay element is declared in the Gameplay Event document, its schema can be seen in appendix A, figures A.4 and A.5.

3.6.1.3 Other documents

To keep a certain degree of organization in the narrative documents and to take advantage of the reusability of certain narrative elements, the rest of the narrative elements are declared in different documents, which the Event documents reference (see figure 3.10).



(a) A sprite



(b) The tileset

Figure 3.12: A sprite and its tileset

Every NPC, PC, item, or enemy is represented in this game by a sprite. Sprites are two-dimensional images or animations that represent the non-static elements of a 2D game [Sobolev, 2019].

A game can use a huge variety of sprites. These sprites are often bundled into tilesets, which are images consisting of several sprites, of the same dimensions, organized in a matrix. An example can be seen in figure 3.12. Each sprite is then referenced by its index in the matrix. To declare the tilesets used in the game, the Tileset document was created. For more details on how each tileset is declared in the Tilesets document, its schema can be seen in figure A.9.

The Actors document declares the sprites, the sprite animations, and the dimensions of the physics body (used to determine collisions in the game) of each actor. Then an enemy, a PC, or an NPC can be instantiated as one of these actors and use its sprites. For more details on how each actor is declared in the Actors document, its schema can be seen in figure A.7.

The Items document is similar to the Actors document, as it declares the sprites for each item. Any item present in a gameplay event is instantiated as one of the items declared in the Items document. For more details on how each item is declared in the Items document, its schema can be seen in figure A.8.

Unfortunately, due to lack of time, battles were not implemented in the game. Still, a draft for the documents related to the battle system was created. This game's genre customary battle system is turn-based combat. To define the actions that the player and the enemies can take in battle, the Playable Characters document and the Enemies document was created. These documents contain information on battle stats such as health, mana, attack power, and the attacks or actions that they can take. Additionally, enemy actions (the enemy's "artificial intelligence") are also declared in the Enemies document. The drafted schema for these documents can be seen in figure A.10.

3.7 Story Viewer as a Narrative Visualization and Debugging tool

Writing a game narrative with several branches and keeping it cohesive can be complicated to manage. And, although JSON files are easily read and written by humans, they can get quite confusing when they become too complex. These conclusions were reached while designing the narrative documents. So that game designers could write annotated narrative structures more easily, a visualization tool for the narratives was created, called the **StoryViewer**. This tool's main objective is to show the contents of the Overall Narrative Document in a more "user-friendly" way.

3.7.1 StoryViewer Overview

The StoryViewer shows the contents of the document as a directed graph. This allows the user to better visualize the order of the events and locations and the possible branching at each step of the story.

Figure 3.13 shows the main screen of the StoryViewer. In (1) the user can upload which Overall Narrative document they want to see. The graph can be seen in the graph visualization window (2), where the user can move it around and zoom it. Hovering a node with the mouse highlights the node's name and shows the event's description (as seen in the figure, on node anxious_startingCutscene). The graph's visualization is detailed in the next paragraph. In the options window (3), the user can upload a player affective profile document to see the branch that the player will play highlighted in the graph (see figure 3.14). The user can also choose to see the default branch, which corresponds to the scenes and events that have the lowest priority and no affective conditions. In the errors window (4), the user can see if the document that they uploaded has any syntax or semantic errors. Additionally, the errors window also reports any infinite loops between events (identifiable by cycles in the graph).

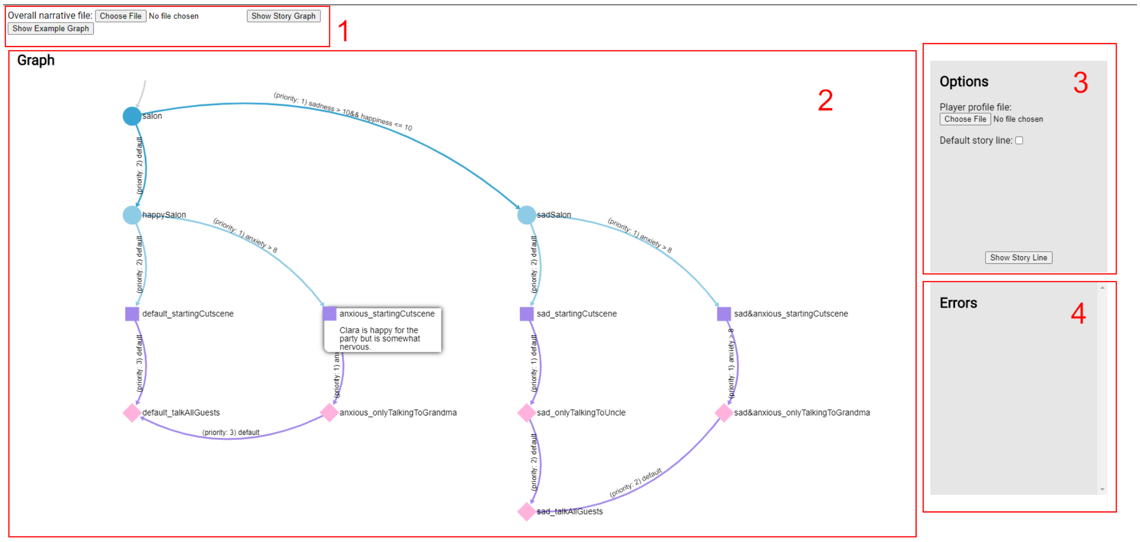


Figure 3.13: The main screen of the StoryViewer

Each element of the narrative is represented as a node, with a different symbol and color for better readability. The start of the narrative is marked with the 'Start' node (gray, star-shaped node). Dark blue circles are locations, light blue circles are scenes, purple squares are cutscene events and pink diamonds are gameplay events. Since it is a directed graph, the direction of the edges shows the order of the story. Each edge is also annotated with the affective conditions for each scene or event. This representation lets the story designer easily understand which branch of the story each player will see.

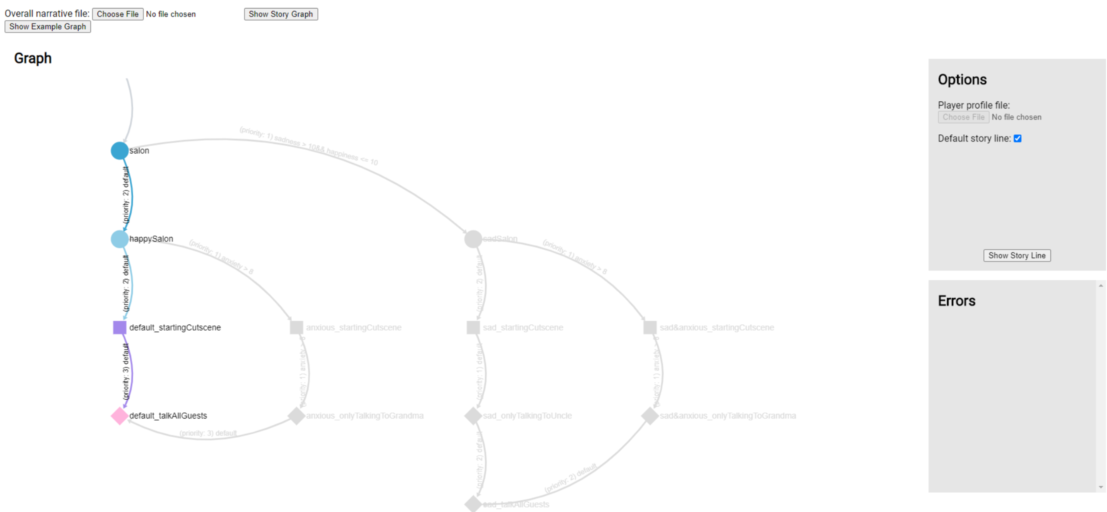


Figure 3.14: The default story branch highlighted in the StoryViewer

3.7.2 Narrative Debug Tool

To be able to debug the narrative more intuitively while creating the game, the StoryViewer tool was repurposed as a narrative debug tool. The debug tool uses the StoryViewer interface to show the user the entire narrative graph, highlighting the branch that the player will play and the current event that the player is seeing.

Figure 3.15 shows the main screen of the debug tool. The debug tool uses the same color scheme as the StoryViewer, except that it highlights the current event node by enlarging it and painting it orange. Additionally, the name of the current event is also shown on top of the graph. It also shows the player's affective profile and possible errors that the graph might have in a more user-friendly way. Previously, without the debug tool, the developer could only access those errors by opening the browser's console.

Figure 3.15: Screenshot of the debug view

3.7.3 StoryViewer implementation

The StoryViewer is a web application and can be accessed by a browser. Node.js was used to create the server side of the application. The client side is written in HTML and Javascript, with the help of the sigma.js[11] library to draw the graph.

When in narrative visualization mode, a user can upload an Overall Narrative document and see the narrative graph. First, the uploaded document is validated against its schema (appendix A) to detect syntax errors, using the AJV tool. If the document has any syntax errors, the graph is not created, and the user is notified on the errors window. Then, it creates the graph by using the same graph creator algorithm that is used by the Narrative Adapter component (section 3.5.2.2). Finally, it renders the graph using the sigma.js library.

[11] sigma.js main page. http://sigmajs.org/

To use the debug tool, both the StoryViewer's server and the development server of the game must be running. When starting, the game sends the entire narrative graph created by the Narrative Adapter component, the player's affective profile, the current event that is being seen by the player, and a unique id to identify the play session to the StoryViewer. The StoryViewer stores that information in local storage. Whenever the current event changes, the game notifies the StoryViewer. The graph's view is updated every second to reflect which event the player is seeing.

3.8 Summary

The resulting framework is composed of three components: the Narrative Adapter component, the Game component and a Narrative Viewer and Debugging tool. The Narrative Adapter component is responsible for creating the adapted narrative for the player. The Game component instantiates that adapted narrative. The Narrative Viewer and Debugging tool is an utility tool for the game designer, to view all the possible narratives and to debug the narrative while the user is playing the game.

This framework is then able to build game narratives that are adapted to the player's affective profile, by changing the game narrative elements described in section 3.1.2. To understand whether these adapted game narratives influence the player's engagement and game experience, user tests were done, which are described in the next chapter.

Chapter 4

Evaluation

The user experiments have the objective of answering the research questions raised in section 1.2. Specifically, they intend to examine whether the game created with the framework improved the players' experience and increased their engagement in the game.

To this end, an online platform was created where the users could partake in the experience. This chapter details the evaluation protocol in section 4.1 and the experience's structure and data to be gathered in section 4.2. Then it presents the gathered results and respective analysis in section 4.4. This chapter ends with a small analysis and conclusion in section 4.5.

4.1 Evaluation Protocol

The framework described in chapter 3 was developed with the objective of adapting the game narrative to the player's emotional state, so that the player can feel more engaged and overall have a more meaningful game experience. The evaluation protocol defined here intends to test whether an adapted game narrative can improve the game experience and engagement.

The full evaluation experience is divided in three steps. The first has the user start by answering some general questions, such as age, native language and gaming habits. Then the user answers the DEQ (discrete emotions questionnaire), to gather their affective profile.

After these initial questionnaires, the second step starts, where the user is asked to play a version of the game with a default narrative and a version of the game with an adapted narrative, in a random order. After each play session, the user fills a questionnaire to obtain their game experience. To compare the values of the game experience and engagement between versions and to learn whether there was an improvement, the difference between these results is used as a measure. So that the order of the versions does not influence the results, the distribution should be aimed at having half of the users play the default version first and half the users play the adapted version first.

The third and final step has the user answer some general questions regarding the experience as a whole.

4.2 Experience Structure and Gathered Data

The experience starts with a short introduction of the framework being tested and an explanation on what the user needs to do in each of the 3 steps that make up the experience. The user is informed that all gathered data is anonymous and that the data will only be used for academic purposes.

The user then starts the first step of the experience, divided in two pages. The first page consists of three questions: age, native language and gaming habits. If the user answers that they play games regularly (6 hours or more per week), they answer two more questions. The first asks what their favorite game genre is and the second asks the importance of the narrative in games to them, measured in a Likert scale.

The second page of the first step contains the DEQ. As mentioned in section 3.1.3, the DEQ was chosen as the questionnaire that was more adequate for this experience, as it measured 8 emotions as opposed to only positive and negative affect. One of the setbacks in using the DEQ was that it showed better results with people whose native language was the one that the DEQ was written on, in this case English. This is the reason why the participants are asked whether their native language is English in the first page. The DEQ consists of 32 words (table 4.2), each representing an emotion, which the user rates using a Likert scale, ranged from 1 to 7 (table 4.1).

1	2	3	4	5	6	7
Not at all	Slightly	Somewhat	Moderately	Quite a bit	Very much	An extreme amount

Table 4.1: DEQ scale

Although they are displayed in no particular order to the user, each of the 32 words belong to a group of 4, correspondent to one of the eight emotions. The affective profile of the user is calculated by the sum of the scores of the four words correspondent to each emotion. Each emotion in the affective profile can then have a value ranging from 4 (all 4 words scored at 1) to 28 (all 4 words scored at 7).

An introductory text in the DEQ page solicited the users to take some time and think about how they felt throughout the ongoing week up until the point of answering the questionnaire. The users should then rate each word with those feelings and emotions in mind.

After answering the DEQ, the user is directed to the second step of the experience, which starts with the first play session. The decision of which version the user should play first is made based on which version the last user that played before them played first. This way, a close to 50/50 division of the players can be achieved. The user does not know which version they are playing. A more detailed description of the game that was presented to the user, of the default and the adapted versions, can be seen in section 4.3. Game logs are saved when the game starts and when it ends,

Anger (Ag)	Scared (F)
Wanting (Dr)	Mad (Ag)
Dread (Ax)	Satisfaction (H)
Sad (S)	Sickened (Dg)
Easygoing (R)	Empty (S)
Grossed out (Dg)	Craving (Dr)
Happy (H)	Panic (F)
Terror (F)	Longing (Dr)
Rage (Ag)	Calm (R)
Grief (S)	Fear (F)
Nausea (Dg)	Relaxation (R)
Anxiety (Ax)	Revulsion (Dg)
Chilled out (R)	Worry (Ax)
Desire (Dr)	Enjoyment (H)
Nervous (Ax)	Pissed off (Ag)
Lonely (S)	Liking (H)

Ag = Anger items, Dg = Disgust items, F = Fear items, Ax = Anxiety items, S = Sadness items, Dr = Desire items, R = Relaxation items, H = Happiness items.

Table 4.2: The 32 words and their groups of the DEQ

whenever the player interacts with a NPC and when an event starts and ends. Each log contains a timestamp and the interaction logs additionally contain which flags changed in that interaction.

After playing the game, the user proceeds to answer some questions regarding their game experience. Firstly, they answer a set of questions taken from the Game Experience Questionnaire (GEQ) created by IJsselsteijn et al.[IJsselsteijn et al., 2013]. The questionnaire is divided in modules, the Core module, the Social Presence module and the Post-Game module. The module that seemed the most appropriate to use in this experiment was the Core module. The Core module has 33 questions, organized in seven groups, each correspondent to a measure or component of the game experience. The seven components are **Competence**, **Sensory and Imaginative Immersion**, **Flow**, **Tension/Annoyance**, **Challenge**, **Negative affect** and **Positive affect**. The Core module questions can be seen in table 4.3. Questions 2, 10, 15, 17 and 21 (Competence) and 11, 23, 26, 32 and 33 (Challenge) were omitted, as they were not appropriate for the kind of game that was being played and were not relevant to the objective of the framework.

The user answers each question with a Likert scale, ranging from 0 to 4 (table 4.4).

0	1	2	3	4
Not at all	Slightly	Moderately	Fairly	Extremely

Table 4.4: GEQ scale

The score of each component is calculated through the sum of the scores of the questions from that component. Each component of the GEQ has a different number of questions, which means the maximum score for each component varies, as shown in table 4.5.

Questions		Component
1	I felt content	Positive affect
2*	I felt skilful	Competence
3	I was interested in the game's story	Sensory and Imaginative Immersion
4	I thought it was fun	Positive affect
5	I was fully occupied with the game	Flow
6	I felt happy	Positive affect
7	It gave me a bad mood	Negative affect
8	I thought about other things	Negative affect
9	I found it tiresome	Negative affect
10*	I felt competent	Competence
11*	I thought it was hard	Challenge
12	It was aesthetically pleasing	Sensory and Imaginative Immersion
13	I forgot everything around me	Flow
14	I felt good	Positive Affect
15*	I was good at it	Competence
16	I felt bored	Negative Affect
17*	I felt successful	Competence
18	I felt imaginative	Sensory and Imaginative Immersion
19	I felt that I could explore things	Sensory and Imaginative Immersion
20	I enjoyed it	Positive affect
21*	I was fast at reaching the game's targets	Competence
22	I felt annoyed	Tension/Annoyance
23*	I felt pressured	Challenge
24	I felt irritable	Tension/Annoyance
25	I lost track of time	Flow
26*	I felt challenged	Challenge
27	I found it impressive	Sensory and Imaginative Immersion
28	I was deeply concentrated in the game	Flow
29	I felt frustrated	Tension/Annoyance
30	It felt like a rich experience	Sensory and Imaginative Immersion
31	I lost connection with the outside world	Flow
32*	I felt time pressure	Challenge
33*	I had to put a lot of effort into it	Challenge

Table 4.3: GEQ's Core module questions
The questions marked with * were omitted

GEQ Component	Max score
Sensory and Imaginative Immersion	24
Flow	20
Tension/Annoyance	12
Negative affect	16
Positive affect	20

Table 4.5: Minimum and maximum scores for each component of the GEQ

Secondly, they can answer some optional open answer questions about the game experience and the narrative. Those questions are:

- In your own words, how did you feel AFTER playing the game?
- What do you think the game reflected about you or your emotions?
- What did you think of the main character?

After answering these questions, the user is directed to play a second version of the game (adapted or default, depending on the version they played first). Again, the user does not know which version they are playing. The set of GEQ questions and optional open answer questions is repeated after the second play session.

When both play sessions and after play session questionnaires are completed, the user goes to step 3, the last step and page of the experience, where they answer some general questions about the experience as a whole. This page is composed of 3 questions, which are:

- Which version of the game did you like the most?
- Which version of the game did you think reflected you or your emotions the most?
- Would you like to play more games like this in the future?

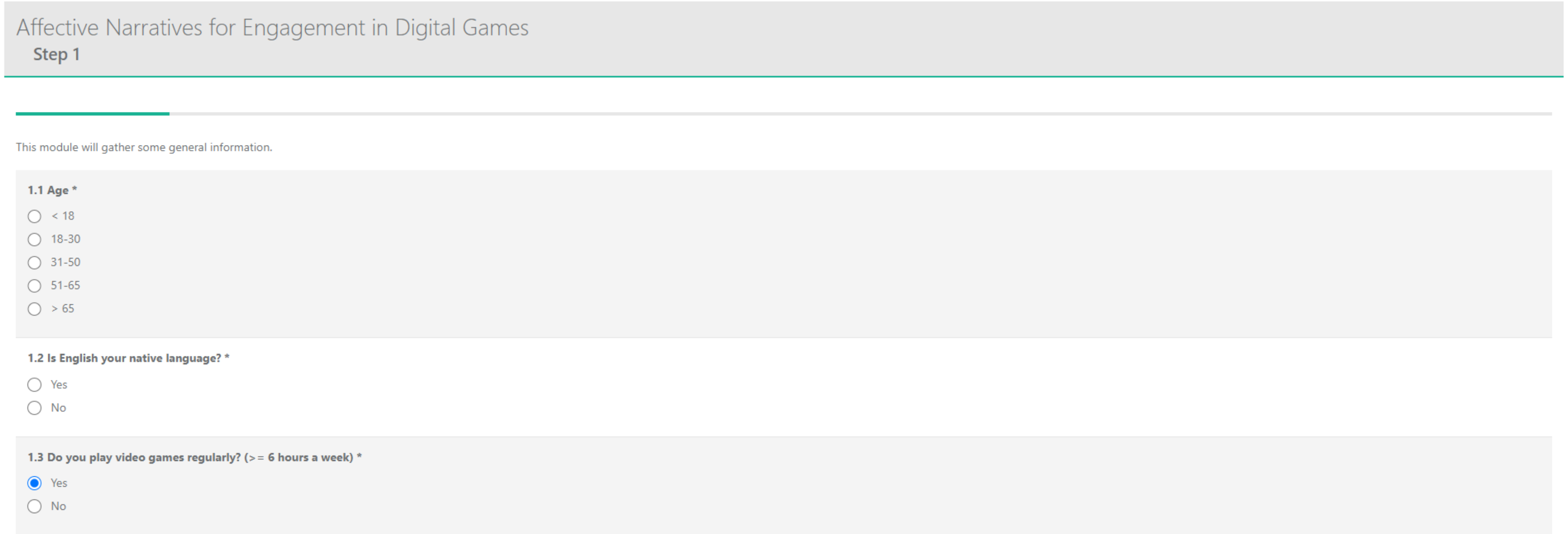
To the first two questions, the user could respond "the 1st version", "the 2nd version" and "none of the versions". For the third question, they could respond "yes" or "no". For all the questions, an optional text area was available for further explaining their answers.

Additionally, this last page also included an optional text area for suggestions and comments on the experience.

This experience was available in a custom online platform, hosted in the faculty's server. To create a pleasant user experience while filling the questionnaires, and for integrating the game sessions seamlessly in the flow, SurveyJS[1], a Javascript library to create custom surveys, was used. The data was saved in a SQLite database and PHP was used to create the server side of the platform and to handle communication with the database.

Figure 4.1 shows a screenshot of the first page of the questionnaire, created with the SurveyJS library. Appendix C can be consulted for screenshots of all the pages of the questionnaire.

[1]SurveyJS main page. https://surveyjs.io/



Affective Narratives for Engagement in Digital Games
Step 1

This module will gather some general information.

1.1 Age *

- ◯ < 18
- ◯ 18-30
- ◯ 31-50
- ◯ 51-65
- ◯ > 65

1.2 Is English your native language? *

- ◯ Yes
- ◯ No

1.3 Do you play video games regularly? (>= 6 hours a week) *

- ◉ Yes
- ◯ No

Figure 4.1: The first page of the questionnaire

4.3 The Game and its Default and Adapted Narratives

Each test participant played a default version of the game and an adapted version of the game, in a random order. The default version of the game had the same narrative for every participant. The adapted version had its narrative adapted to the participants' affective profile, extracted from the previously given DEQ.

The game's gameplay was fairly simple, it only involved walking around and talking to NPCs. Some screenshots can be seen in figure 4.2. To walk, the players would use the arrow keys. To interact with NPCs, the players would use the Z key or the space bar. These instructions were given to the player before each play session in the game window itself, but were also available in the game session page below the game window (as seen in figures C.6 and C.7). Along the game instructions, a small introduction to contextualize the setting of the game was also given (seen in figure C.6).

Figure 4.2: Some screenshots of the game

The game scenarios for each possible narrative were written so that they would be short, to not tire the players, since they would play two versions almost back to back. The game was inspired by the narrative that was written for the toy example (figure 3.3). The player controls Clara, a daughter of a rich family. It is Christmas and her parents are throwing a party. Clara should attend the party and greet every guest as is expected of her. How Clara handles this responsibility is decided by the players' emotions gathered with the DEQ.

The default version of the story starts with a cutscene showing Clara descending the stairs to the mansion's salon and expressing her enthusiasm for the party. The salon is well lit, colorful, full of Christmas decorations. Then, the player is prompted to greet every guest in the salon, which total 5 NPCs. The player would greet Clara's uncle, Clara's grandma, a group of two girls and a woman. The interactions consist of mostly generic, hopefully comedic, dialogue. Clara's uncle tells her he does not want to be there, but his sister, Clara's mother, made him attend, in the hopes of seeking a potential romantic partner. Clara's grandma is just glad to see her granddaughter. The group of two girls swoon over the cakes being served at the party and worry about their weight. The woman tells Clara how much she loves her parents' parties and that they are the best in town. Finishing this task, another cutscene starts, where Clara reflects on how the night went.

For the adapted version, different events and scenes were written for the emotions contained in the affective profile: anger, disgust, anxiety, sadness, desire and happiness. Although fear and relaxation are also a part of the affective profile, no changes to the default narrative were written based on these emotions. The changes to the default narrative for each emotion can be seen in table 4.6.

The full StoryViewer generated graph is hard to analyze, due to having a lot of nodes and edges. Figures 4.3 and 4.4 show a subset view of the graph containing the default storyline and an example of an adapted storyline, respectively. The full graph with all the possible narratives can be seen in the appendix in figure D.1.

Emotional condition	**Changes to the default narrative**
Sadness >10	The player sees a different salon, with the same decorations and items but faded colors Clara states that she would rather be somewhere else than the party, but her parents made her attend She does not feel like greeting the guests
Sadness >10 and Anxiety <= 8	She sees her uncle walking in the salon. It is unusual for him to attend so she decides to only greet him as of now After greeting her uncle, Clara feels better and decides to try and greet the guests
Sadness >10 and Anxiety >8	Clara feels anxious as she has not seen the guests in a very long time, and her sad mood is not helping She sees her grandma walking in the salon. She decides to seek her advice After talking to her grandma, Clara feels better and decides to try and greet the guests
Sadness <= 10 and Anxiety >8	Clara feels nervous as she has not seen the guests in a long time and is afraid that she will do something wrong She seeks her grandma for advice and after talking to her gains some courage to greet the guests
Sadness <= 10 and Happiness >8	Clara's grandmother gives her a gift, which raises Clara's happiness
Disgust >8	The woman guest has eaten something that makes her breath smell bad Clara is disturbed by it
Desire >8	After greeting all the guests, Clara gets a sudden desire to eat cake She goes on a quest to find the cake, and urges the player to remember who talked to her about the cake (the group of two girls) She talks to the girls and they give her one of their cakes, afraid to eat more and get out of shape
Anger >8	Just as she was finishing greeting the guests, a loud crash is heard Clara's brother shows up in the salon with a guilty look Clara confronts him and he confesses that, while playing and running in the hallways, he broke her favorite wall mirror Clara gets mad and her brother runs away

Table 4.6: The changes to the default narrative for each emotion of the affective profile (excluding fear and relaxation)

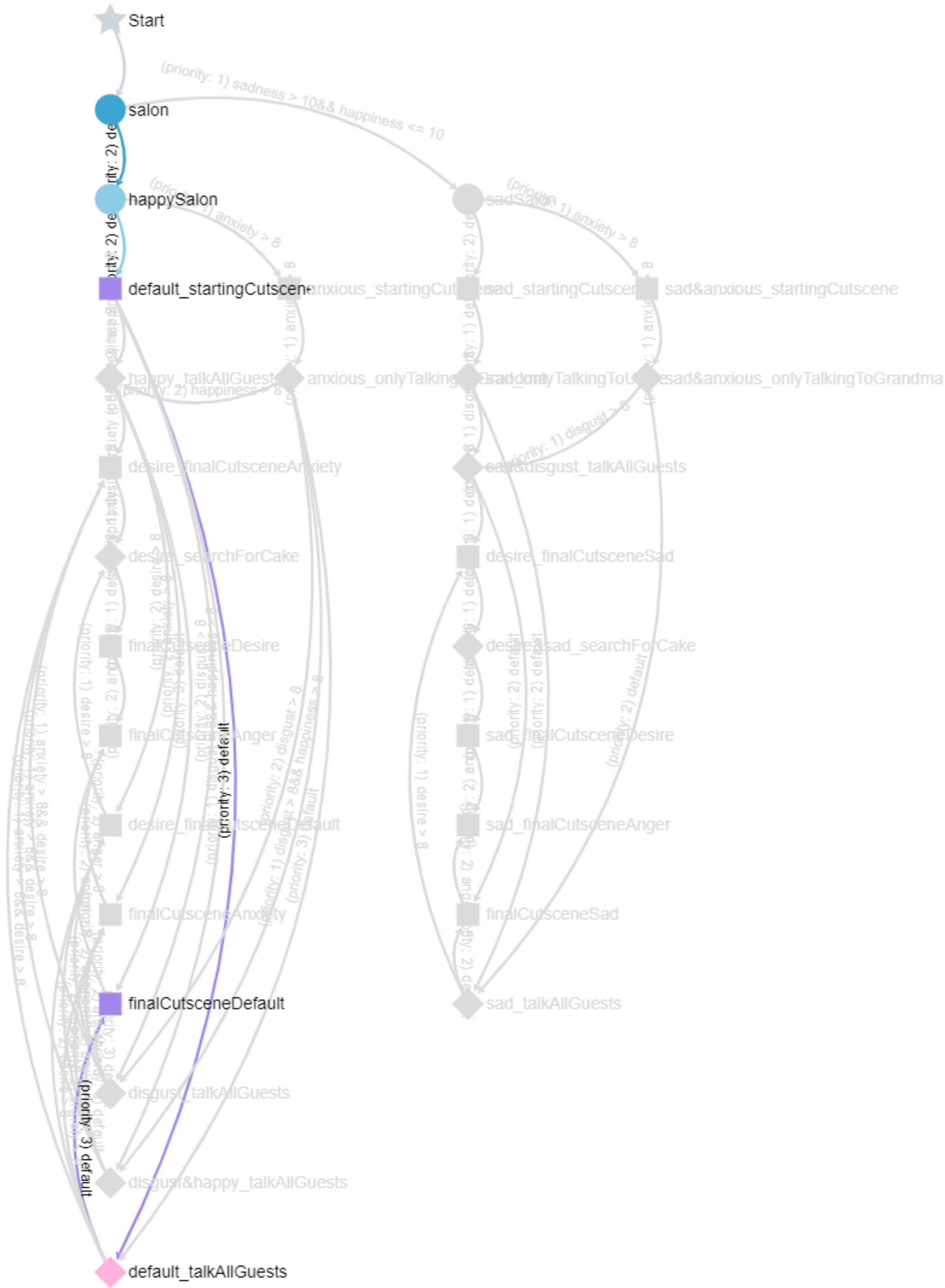


Figure 4.3: View of the default storyline

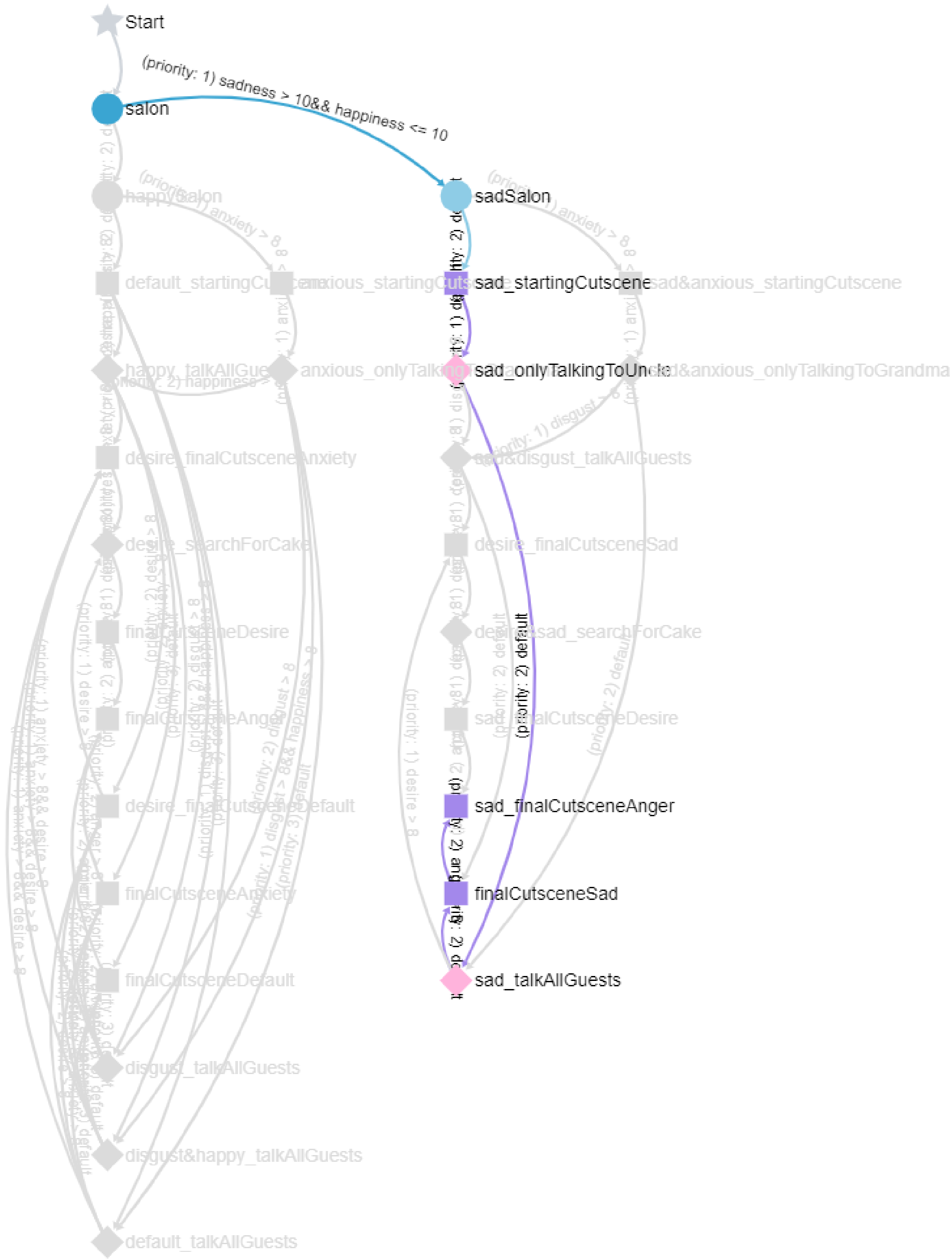


Figure 4.4: View of an adapted storyline
Affective profile:
anger: 10, disgust: 7, fear: 4, anxiety: 5, sadness: 14, desire: 7, relaxation: 4, happiness: 4

4.4 Results and Analysis

In total, 107 users participated in the experience. From the 107 participants' data, 106 were considered valid and 1 invalid, as they gave the same answer to all the questions with a Likert scale and the open answer questions were answered inappropriately. From the 106 participants, 56 played the default version first (52.83%), which means the ratio between users that played the default version first and users that played the adapted version first was very close to 50%, and hence balanced. This section will show the results and analysis of the 106 participants' data for each part of the experience described in the previous sections.

4.4.1 Part 1 - General Questions

The ages of the participants can be seen in table 4.7. The vast majority of the participants were aged between 18 to 30 years old. Only one participant was an English native speaker, as opposed to the other 105. Due to only having one English native speaker, no analysis could be done to assess if the language barrier affected filling out the DEQ and consequently the game experience.

Age	Participants
<18	2
18 - 30	92
31 - 50	12
51 - 65	0
>65	0

Table 4.7: Participants' ages

When asked whether they played games regularly (for 6 hours or more weekly) 66 participants (62.26%) answered yes. From those 66 participants, the answers to their favorite game genre and the importance of narrative in games can be seen in graph 4.5. The graph shows that the favorite genre amongst most of the regular players is RPG (22, 33.33%). Additionally, the narrative was considered important (scored 4 or more out of 5) by 56 (84.85%) of the regular players. In every genre, half or more of the players considered the narrative to be important, except for players that prefer the action genre.

4.4.2 Part 2 - Game Sessions and GEQ

The average game time for the two game versions can be seen in table 4.8. As expected, players spent less time on the default version than the adapted version, as the adapted version had more content. The adapted version's average time for players that played it in the 1st session and for players that played it in the 2nd session is similar. On the other hand, players spent more time in the default version when playing it in the 1st session as opposed to the players that played it in the 2nd session. This might be due to players using the first session to understand the controls and how the game works.

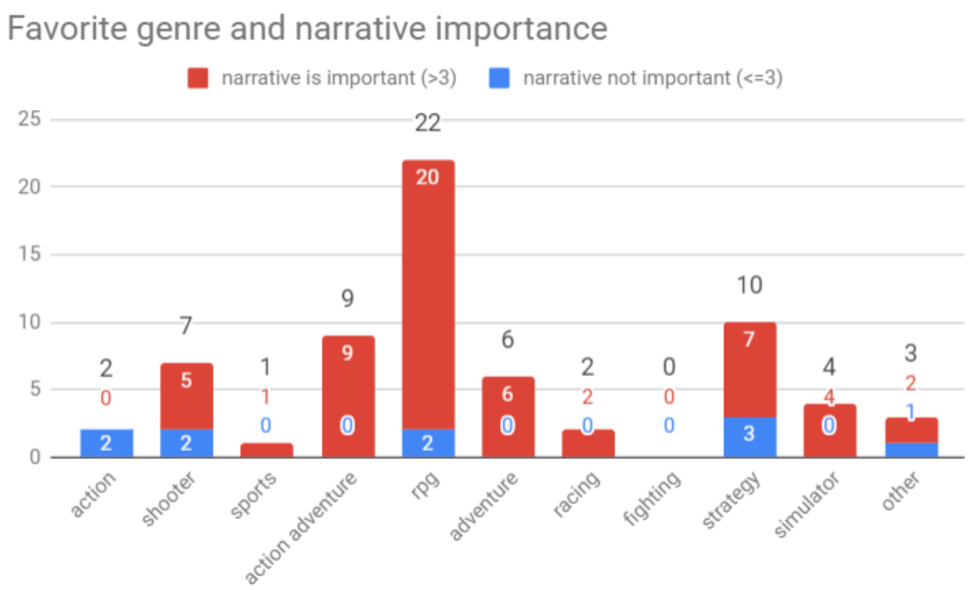


Figure 4.5: Favorite game genre and importance of narrative amongst regular players

To confirm if the adapted version of the game improved the experience and engagement of the players when compared to the default version, the difference between the scores of the adapted version and the default version for each category of the GEQ was calculated for each player. If the difference of a certain component is a positive number, it means that the score of that component increased from the default to the adapted versions. Likewise, if the difference is a negative number, it means that the score decreased. The average, median, minimum and maximum values of the difference of scores as well as the number of increases, decreases and indifferences (difference = 0) between versions for all the players can be seen in table 4.9.

To learn whether the order of the sessions influenced the results, the same calculations were done only for the players that played the default version first (default-first group) and only for the players that played the adapted version first (adapted-first group). The results can be seen in tables 4.10 and 4.11, respectively.

For an improved and more engaging game experience, it is generally expected that the components Sensory and Imaginative Immersion, Flow and Positive affect increase from the adapted to the default versions, while the Tension/Annoyance and Negative affect are expected to decrease.

Average game time (mins:secs.mills)	For players that played it in 1st session	For players that played it in 2nd session	For all players
Default version	02:07.761	01:26.800	01:48.440
Adapted version	04:43.217	04:59.965	04:52.065

Table 4.8: The average game time for the two game versions

Total: 106 players	**Sensory and Imaginative Immersion** (0-24)	**Flow** (0-20)	**Tension/ Annoyance** (0-12)	**Negative affect** (0-16)	**Positive affect** (0-20)
Average	1.623 *	1.208	-0.019 **	-0.500	1.113
Median	1	1	0	0	0
Max	17	13	6	9	12
Min	-11	-8	-9	-11	-10
Total Increases	67 (63.21%)	63 (59.43%)	27 (25.47%)	35 (33.02%)	48 (45.28%)
Total Decreases	23 (21.70%)	29 (27.36%)	27 (25.47%)	45 (42.45%)	34 (32.08%)
Total Indifferences	16 (15.09%)	14 (13.21%)	52 (49.06%)	26 (24.53%)	24 (22.64%)

Table 4.9: The difference between the score of the adapted version and the score of the default version for each category of the GEQ among **all players**

* A positive difference means the adapted version ranked more than the default version
** A negative difference means the adapted version ranked less than the default version

Total: 56 players	**Sensory and Imaginative Immersion** (0-24)	**Flow** (0-20)	**Tension/ Annoyance** (0-12)	**Negative affect** (0-16)	**Positive affect** (0-20)
Average	1.500	0.357	0.107	-0.768	0.679
Median	1	0	0	0	0
Max	17	13	5	5	10
Min	-6	-8	-5	-7	-10
Total Increases	37 (66.07%)	26 (46.43%)	16 (28.57%)	17 (30.36%)	21 (37.50%)
Total Decreases	13 (23.21%)	25 (44.64%)	15 (26.79%)	26 (46.43%)	22 (39.29%)
Total Indifferences	6 (10.71%)	5 (8.93%)	25 (44.64%)	13 (23.21%)	13 (23.21%)

Table 4.10: The difference between the score of the adapted version and the score of the default version for each category of the GEQ among the players of the **default-first group**

Total: 50 players	**Sensory and Imaginative Immersion** (0-24)	**Flow** (0-20)	**Tension/ Annoyance** (0-12)	**Negative affect** (0-16)	**Positive affect** (0-20)
Average	1.760	2.160	-0.160	-0.200	1.600
Median	2	2	0	0	1
Max	11	13	6	9	12
Min	-11	-8	-9	-11	-10
Total Increases	30 (53.57%)	37 (66.07%)	11 (19.64%)	18 (32.14%)	27 (48.21%)
Total Decreases	10 (17.86%)	4 (7.14%)	12 (21.43%)	19 (33.93%)	12 (21.43%)
Total Indifferences	10 (17.86%)	9 (16.07%)	27 (48.21%)	13 (23.21%)	11 (19.64%)

Table 4.11: The difference between the score of the adapted version and the score of the default version for each category of the GEQ among the players of the **adapted-first group**

The following subsections will focus on the analysis of such increases or decreases of the score for each component of the GEQ.

4.4.2.1 Sensory and Imaginative Immersion

For the Sensory and Imaginative Immersion component, it can be seen in table 4.9 that there was an average increase (difference of 1.623) for this component from the adapted version to the default version. Most users (63.21%) reported this component increasing in the adapted version compared to the default version, with a maximum difference of 17 (increase) and a minimum difference of -11 (decrease).

Comparing the results between the user groups (tables 4.10 and 4.11), the adapted-first group had a slightly greater average increase. But, the adapted-first group also reported the biggest decrease between versions (minimum difference of -11). Still, both groups had a similar percentage of users (66.07% and 53.57%) report that this component improved between versions. Additionally, the average increase was similar in both groups. It can be concluded that, for this component, the order of the sessions did not influence the Sensory and Imaginative Immersion component's results.

The Sensory and Imaginative Immersion component measures whether the players were interested and attracted to the game, in other words, if they liked the game. The results show that this component increased in score from the default version to the adapted version, meaning that, overall, the players preferred the adapted version to the default version.

4.4.2.2 Flow

For the Flow component, table 4.9 shows that there was an average increase of score (1.208) from the adapted version to the default version. Again, most users reported an increase in this component (59.43%), with a maximum difference of 13 (increase) and a minimum difference of -8 (decrease).

The default-first group (table 4.10) has an average difference of 0.357, which gives the impression that the Flow component did not change much from the default version to the adapted version. However, the total number of increases and total number of decreases show that this is not the case: 44.43% of the players of this group reported an increase in Flow, with a maximum increase of 13, while 44.64% reported a decrease in Flow, with a maximum decrease of 8. Only 8.93% of the players reported indifference in the Flow between versions. This, also supported by the median value (0), means that approximately half of this group reported an increase in Flow, while the other half reported otherwise.

The adapted-first group (table 4.10) has an average difference of 2.160, a substantially bigger increase compared to the default-first group. It can also be seen that the majority of the players of this group (66.07%) reported an increase in this component.

This difference of results from the two groups shows that the Flow was influenced by the order of the sessions. The Flow component measures whether the players were engaged and concentrated in the game. The default-first group were possibly as engaged in the default version of the game as they were engaged in the adapted version, since in the first session the game was new to them and they had to learn how to play. However, since there was not an overall decrease in Flow in the second session (adapted version), it could mean that, although not being a completely new game, the differences in the narrative were enough to keep the players engaged. On the other hand, the adapted-first group learned how to play and received a longer, possibly more interesting version of the game in the first session, possibly losing engagement in the default version of the game, as it had nothing "new" to show them.

4.4.2.3 Tension/Annoyance

Table 4.9 shows that, for most players (49.06%), there was no difference between the adapted version and the default version for this component. However, the average difference of -0.019 shows that tension and annoyance decreased for the average player from the adapted to the default versions, even if ever so slightly.

The results are similar for both the default-first and adapted-first groups (tables 4.10 and 4.11), with the exception that the default-first group reported an increase in tension and annoyance from default to adapted versions. This can be due to the fact that, since the adapted version has a longer game time, the players that did not enjoy the game felt annoyed that they had to play an even longer version of the game.

4.4.2.4 Negative affect and Positive affect

The Negative affect and the Positive affect components measure if the player felt unpleasant feelings or good feelings while playing the game, respectively. Negative affect can mean that the player was bored or was in a bad mood. Positive affect can mean that the game was fun to the player, but can also mean that the player was happy. These components are difficult to analyze in this context, for example, the result of a bad mood can mean that they felt empathy towards the main character, revealing engagement in the game and not dislike of it.

For the Negative affect component, the results of all players were similar to the results of their groups. The three tables, 4.9, 4.10 and 4.11, show that there was an average decrease for this component. Although the total number of decreases is bigger than the number of increases and indifferences, it is not the majority.

Similar to the Negative affect component, the Positive affect component did not have very different results when seeing the players as a whole or in their groups. This component had an average increase as seen in the three tables, 4.9, 4.10 and 4.11. And again, just like the Negative affect component, although the total number of increases is bigger than the number of decreases and indifferences, it is not the majority.

4.4.2.5 Summary of the analysis of each component

Overall, there was an increase in the Sensory and Imaginative Immersion, Flow and Positive affect components and a decrease in the Tension/Annoyance and Negative affect components, from the default version to the adapted version.

The default-first and adapted-first groups did not have any major differences between each other in their results for each component. Curiously, the players belonging to the adapted-first group had a slightly bigger difference in all of the components, except for the Negative Affect component, when compared to the default-first group. This might indicate that the adapted-first players felt that the default version of the game had nothing new to offer to them, creating a bigger difference between both versions.

4.4.3 Part 3 - Last questions

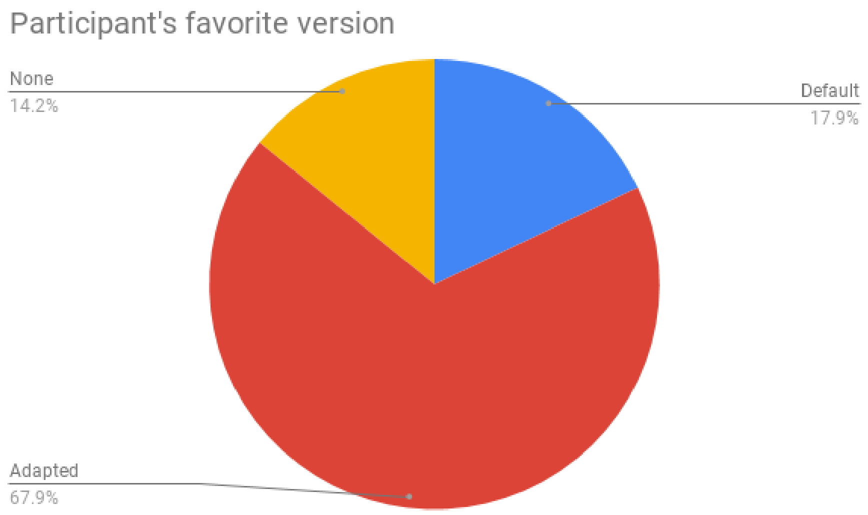
After completing part 2, the players were asked the following questions:

- Which version of the game did you like the most?
- Which version of the game did you think reflected you or your emotions the most?
- Would you like to play more games like this in the future?

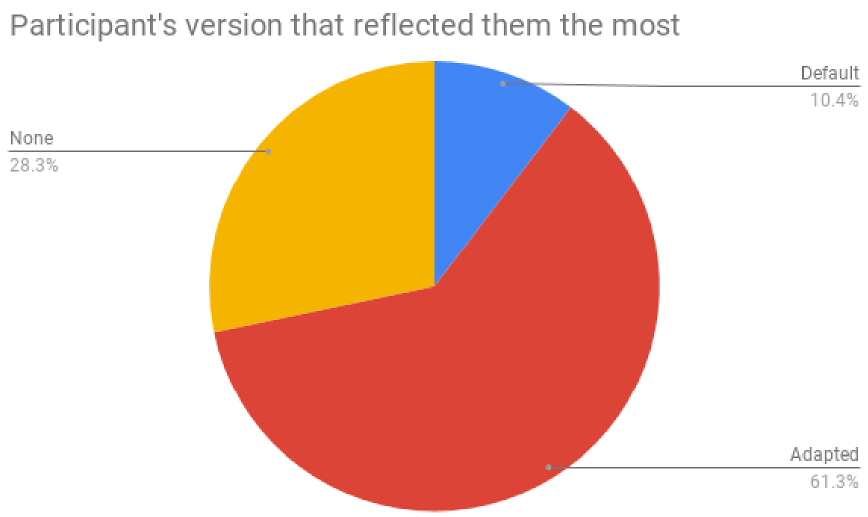
The answers to these questions can be seen in figure 4.6. A comparison between the participant's favorite version and the version that they think reflected them the most can be seen in 4.7.

The results seen in figure 4.6 are consistent with the ones taken from the GEQ (section 4.4.2). The majority of players preferred (67.9%) and thought they were more reflected (61.3%) by the adapted version. Additionally, the majority of players (55.7%) were interested enough that they would like to play a game like this again.

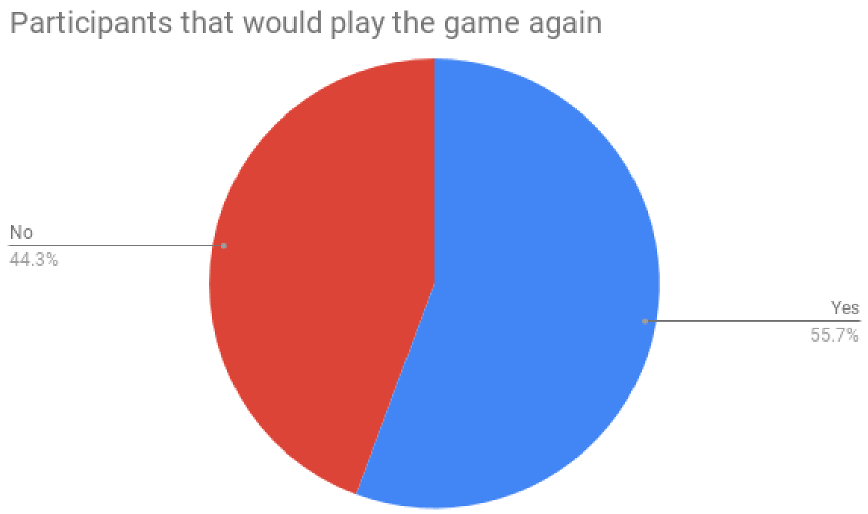
Figure 4.7 shows that the majority of the players that preferred one of the versions as opposed to none would like to play the game again. On the contrary, almost all of the players that preferred none of the versions would not play the game again. It also shows that the majority of the people that prefer the adapted version were the ones that felt more reflected by it. And although some participants felt they were not reflected by any of the versions, they preferred the adapted version (probably due to being longer and not as generic).



(a) Participants' favourite version of the game



(b) Participants' version that reflected them the most



(c) Participants' answer to the question "Would you play a game like this again?"

Figure 4.6: Participant's answers to the last questions

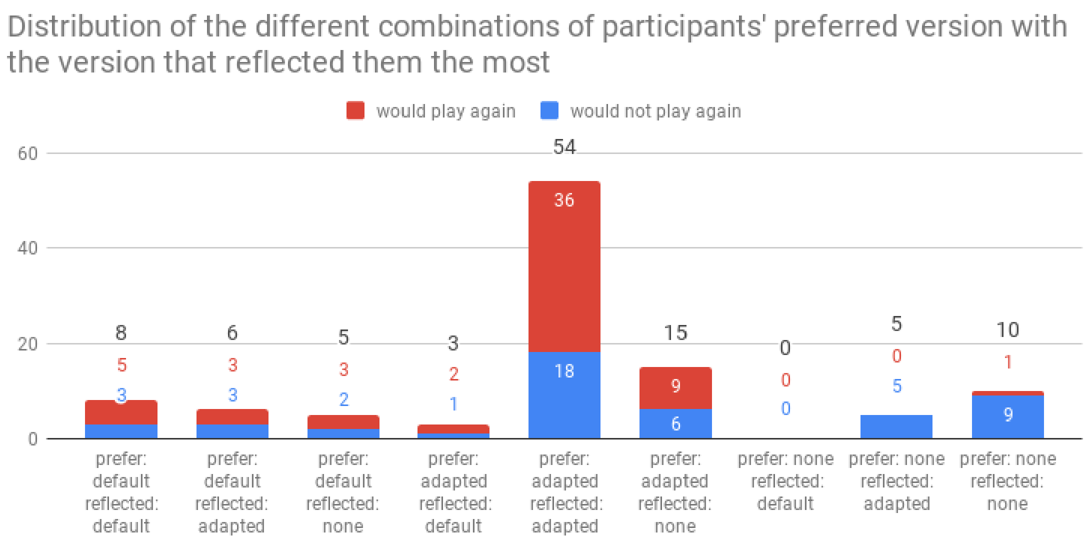


Figure 4.7: Distribution of the different combinations of participants' preferred version with the version that reflected them the most

4.5 Discussion

The user tests that were performed helped understand the impact of a game made with the framework on engagement and game experience.

The results show that none of the participants were indifferent to the adapted version. There was an overall improvement from the default version to the adapted version on the Sensory and Imaginative Immersion, Flow and Positive affect, components of the game experience that, when having a good score, generally mean that the player had a good game experience and felt engaged. On the other hand, the Tension/Annoyance and Negative affect components, generally associated with a bad game experience, had their scores decreased from the default version to the adapted.

The default-first and adapted-first groups did not have any major differences between each other in their results for each component. Curiously, the players belonging to the adapted-first group had a slightly bigger difference in all of the components, except for the Negative Affect component, when compared to the default-first group. This might indicate that the adapted-first players felt that the default version of the game had nothing new to offer to them, creating a bigger difference between both versions.

The majority of the players preferred the version that they felt reflected them the most. This could mean that feeling reflected by the game is an important factor in determining player's preference. This also shows that adapting a game to the player's emotions can lead to the players feeling more reflected by it thus improving their game experience and engagement.

Additionally, seeing that the majority of the participants would like to play a game like this again shows that the framework might successfully help in the creation of more engaging games.

Chapter 5

Conclusions

There are many ways that a story can be told, but one medium that is especially powerful is video games. The interactive component of this medium is very appealing, and lets players experience stories in a way that can not be replicated by any other form of media, as they let the player interact directly with the story and be a part of it. Some video games let the player choose what happens in certain parts of the story. These are very popular, because of the higher degree of customization of their game experience, but also because players often feel more engaged in a story that they influenced and that feels more personal to them. Knowing the importance of personalized narratives in video games, an attempt to adapt the game narrative to the player was done in this thesis. More specifically, a framework that can create adapted stories to the player's emotions was built. The player's emotions were chosen as the form of adaptation since stories that reflect the players' emotions can induce behaviors of self-reflection in the players, thus creating more engaging and meaningful game experiences.

To build this framework, research on previous works that involved game adaptation and player modelling was done (chapter 2), to know how the adaptation of a game usually happens. Video game narratives and their characteristics were also researched, since the subject of adaptation had to be well known. It was also seen that, although there are some works that adapt video game narratives to the player, none exist, as of now, that adapt the narratives to their emotions.

After this initial research, the conceptualization and implementation of the framework began (chapter 3). With an annotated narrative structure created by the game designer and the affective profile provided by the player, the framework is capable of choosing the elements of the narrative that better match the player's emotions, thus creating personalized game narratives.

To learn whether this framework helped in the creation of more engaging and meaningful game experiences, user tests were performed, using a game built with the help of the framework. The users would provide their affective profiles and then play two versions of the game, one with a default narrative and one with an adapted narrative. The results analyzed in chapter 4 show that game experience and engagement increased between the default version and the adapted version. Most players preferred the adapted version and would like to play a game like this again.

Thus, the objective of this thesis, that is to create more engaging and meaningful game experiences by adapting game narratives to the player's emotions, was carried out successfully with this work. However, there are components that could be improved in the future, which are detailed in the next section.

5.1 Future Work

Due to lack of time, there were narrative elements that are not known to the framework, thus limiting the kind of games that the framework can help in creating. For example, the framework does not deal with transitions between locations, meaning that the framework in its current state can only help the creation of games that have only one location, as seen in sections 3.5.2.2 and 3.5.2.3. Additionally, the game description assets described in section 3.6.1 do not cover every aspect of a game. An improvement of these documents would help in creating more varied and interesting games with the framework.

The StoryViewer tool, used to view the annotated narrative structure in graph form and to debug the narrative in game, can also be worked on to improve usability of the framework. For example, the tool could be upgraded from a viewing only tool, to a graphic interface where game designers could directly alter the game narrative by creating new nodes and connect them to the graph, without having to tamper directly with JSON files. Additionally, the StoryViewer would benefit in having an algorithm that could organize the graph nodes automatically for better visualization (see figure D.1 for a current view of how the StoryViewer handles "complex" narratives).

Appendix A

Narrative documents' schemas

This appendix contains a visual representation[1] of the schema of each narrative document that is used by the framework and parsed by the game.

[1] Visual representation of the json schemas achieved with Json Schema Viewer tool available at https://github.com/stoplightio/json-schema-viewer

Player Affective Profile schema

object	{8}	The affective profile of a player	optional+1
anger	integer	The amount of anger that the player is feeling.	optional
disgust	integer	The amount of disgust that the player is feeling.	optional
fear	integer	The amount of fear that the player is feeling.	optional
anxiety	integer	The amount of anxiety that the player is feeling.	optional
sadness	integer	The amount of sadness that the player is feeling.	optional
desire	integer	The amount of desire that the player is feeling.	optional
relaxation	integer	The amount of relaxation that the player is feeling.	optional
happiness	integer	The amount of happiness that the player is feeling.	optional

Figure A.1: Player Affective Profile schema

Overall Narrative schema

- object {2} The representation of the overall narrative of the game — optional
  - firstLocation string Which location should be used for first scene — required
  - scenes array[object] {9} An array containing all the possible scenes of the game. Each scene represents a different map in the game — required
    - locationId string The location to which this scene belongs to. — required
    - name string The name/id of the scene. Must be unique — required
    - description string A small textual description for this scene — required
    - map string The key to the map of this scene — required
    - priority integer In case there is a tie in the emotional requirements of the scenes, the framework chooses the one with the highest priority — required
    - emotionalRequirements array[object] {3} The emotional requirements for this scene to show to the player. This scene will only show to the player if the conditions stated in this array are ALL TRUE — required
      - parameter string The name of the parameter(emotion) to be evaluated. Can be "anger", "disgust", "fear", "anxiety", "sadness", "desire", "relaxation" or "happiness" — required
      - condition string Can be >, <, >=, <=, =, != — required
      - value integer The value to be compared to — required
    - transitions array[object] {3} List of transitions possible from this scene/map to other scenes and the necessary conditions for these transitions to be available — required
      - name string The id/name of this transiction. Must be unique — required
      - toLocation string The id of the location to transition to — required
      - unlockedOnEventsEnding array[string] Which events'endings trigger the unlocking of the transiction — optional
    - firstEvents array[string] A list of event ids (of this current scene) that can be the first event of the scene — required
    - events array[object] {6} The list of events to take place in the scene. Can be cutscenes or gameplay — required
      - name string The name/id of the event. Must be unique. Must match the name (minus the extension) of the file containing the event. — required
      - description string A small textual description of the event — required
      - type string Can be cutscene or gameplay — required+1
      - priority integer In case there is a tie in the emotional requirements of the scenes, the framework chooses the one with the highest priority — required
      - emotionalRequirements array[object] {3} The emotional requirements for this event to be available to the player. This event will only show to the player if the conditions stated in this array are ALL TRUE — required
        - parameter string The name of the parameter(emotion) to be evaluated. Can be "anger", "disgust", "fear", "anxiety", "sadness", "desire", "relaxation" or "happiness" — required
        - condition string Can be >, <, >=, <=, =, != — required
        - value integer The value to be compared to — required
      - nextEvents array[string] A list of event ids (of this current scene) that can be the next events occuring after this event — required

Figure A.2: Overall Narrative schema

Cutscene schema

object {3} The description of a game cutscene	optional
name string The name/id of the event	required
▾ actors array[object] {2} A list of the actors present in the event	required
actorId string The ID of the actor defined in the actors file	required
start array[integer] The starting position of the actor in the map	required
▾ actions array[object] {3} A list of the actions in this event	required
action string The type of action. For example, it can be "walk" or "talk"	required
arguments object {0} The arguments for the specific action	required
actorId string The actor that should perform the action	required

Figure A.3: Cutscene schema

Gameplay schema

Property	Type	Description	Requirement
	object {7}	The description of a game gameplay event	optional
name	string	The name/id of the gameplay event. Should be unique	required
▾ flags	array[object] {2}	The flags that will be used in this gameplay event and their initial values	required
name	string	The flag name. Must be unique from all the flags in this event.	required
value	number or boolean	The inital value of the flag. Can be a boolean or a number	required
endEventCondition	string	The condition for this event to end. Should be a valid expression according to the filtrex parser and containing flags that were previously declared for this event	required
▾ player	object {3}	The description of the player character	required
playerId	string	The Id for this player	required
actorId	string	The id of the actor (previously declared in actors.json) to be used for this player	required
startPosition	array or string	The start position of the player in the map. Can be the previous position where the player left off in the last event (keyword: current) or coordinates to a specific position	required+1
▾ enemies	array[object] {2}	The list of possible enemy encounters in this event	required
id	string	The id of the enemy (previously declared in the enemies.json file) to be encountered	required
encounterPercentage	integer	The probability of the player finding this enemy of all the enemies available. If there is only one enemy, then the percentage should be 100	required
▾ npcs	array[object] {5}	A list of the NPCs available in this event	required
actorId	string	The Id of the actor (previously declared in actors.json) to be used for this npc	required
position	array[integer]	The position of this NPC	required
dialogue	string	The file containing the dialogue for this NPC	required
isInteractableConditions	string	The condition for this npc to be interactable. Should be a valid expression according to the filtrex parser and can contain flags that were previously declared for this event	required
▾ flagChangesAfterInteraction	array[object] {2}	A list of the flag changes after this interaction	required
id	string	The name of the flag. Should be previously declared in this file	optional
value	string	The value that the flag should change to. Should be a valid expression according to the filtrex parser and can contain flags that were previously declared for this event	required

Figure A.4: Gameplay schema

▾ items array[object] {7} A list of the items available in this event		required
itemId string The id of the item (previously declared in items.json) to be used for this item		required
position array[integer] The position of this item in the map		required
dialogue string The dialogue file containing possible dialogue with this item. Can be null		required
▾ inventoryChange object {2} Description of a possible inventory change after interaction with the item		required
itemId string The item id (previously declared in inventory_items.json) that will be changed in inventory		required
action string The kind of change to do in the inventory. Can be "+n" or "-n", where n is the number of items added or subtracted		required
isInteractableConditions string The condition for this item to be interactable. Should be a valid expression according to the filtrex parser and can contain flags that were previously declared for this event		required
▾ flagChangesAfterInteraction array[object] {2} A list of the flag changes after this interaction		required
id string The name of the flag. Should be previously declared in this file		optional
value string The value that the flag should change to. Should be a valid expression according to the filtrex parser and can contain flags that were previously declared for this event		required
disappears boolean A boolean to know if the graphic of the item should disappear from the map if it was interacted with		required

Figure A.5: Gameplay schema (cont.)

Dialogue schema

object {1} The description of an in-game dialogue	optional
▾ lines array[object] {2} An array of all the dialogue lines, in order of appearance	required
author string The author that says the line	required
text string The text of the line	required

Figure A.6: Dialogue schema

Actors schema

	object	{1}	A list of all the actors in the game and their properties	optional
▼ actors	array[object]	{6}	A list of all the actors in the game (protagonists, NPCs, enemies)	required
id	string		The unique ID for this actor	required
name	string		The actor's name	required
tilesetId	string		The name/id of the tileset containing the sprites for this actor	required
defaultFrame	integer		The default/initial frame of this actor.	required
▼ animations	array[object]	{4}	A list of the animations for the actor. The keys "right", "left", "top" and "down" are reserved for walking animations.	required
key	string		The animation key	required
frames	array[integer]		The frames in the tileset required for the animation	required
frameRate	integer		The framerate of the animation	required
repeat	integer		Number of times the animation should repeat. -1 if animation should loop forever	required
▼ body	object	{3}	Description of the physics body of this actor	required
height	integer		The height of the physics body, as a percentage of the height of the sprite.	required
width	integer		The width of the physics body, as a percentage of the width of the sprite.	required
anchor	string		The anchor to which the sprite center will be calculated.	required+1

Figure A.7: Actors schema

Items schema

	object	{1}	A list of all game items that the player can pick and add to the inventory	optional
▼ items	array[object]	{4}	A list of all the items in this game	required
itemId	string		The id for this item. Should be unique	required
name	string		The name of this item as seen by the player	required
tileset	string		The id/name of tileset where the graphics for this item can be found	required
frame	integer		The specific frame in the tileset where the item is	required

Figure A.8: Items schema

Tilesets schema

array[object] {4} A list of all the tilesets used in the game	optional
id string The id of this tileset. Must be unique	required
filename string The filename where this tileset is	required
frameWidth integer The width of each frame	required
frameHeight integer The height of each frame	required

Figure A.9: Tilesets schema

Enemies schema

Field	Type	Count	Description	Status
	object	{1}	A list of all the enemies in the game and their properties	optional
▾ enemies	array[object]	{8}	An array containing all enemies of the game	required
id	string		The unique Id of the enemy	required
hp	integer		The total Health Points of the enemy	required
mp	integer		The total Magic Points of the enemy	required
attack	integer		The base attack of the enemy	required
defense	integer		The defense of the enemy	required
speed	integer		The speed of the enemy to decide its order in the battle turns. The higher speed, the sooner the enemy acts in the turn	required
▾ moveset	object	{2}	A list of all the moves available to this enemy	required
▾ attacks	array[object]	{5}	A list of all the attacks available to the enemy	required
name	string		The name of the attack. The name "attack" is reserved to the basic attack	required
power	integer		The power of the attack. The damage of this attack is basicAttack*power	required
mpCost	integer		The MP cost for this attack	required
accuracy	number		The accuracy of this attack in percentage. If 100% the attack always hits	required
target	string		The target of the attack. If "one" the attacks targets one party member. If "all" the attack targets everyone in the party	required
▾ buffs	array[object]	{5}	The list of possible buffs for this enemy to use in their party	required
name	string		The name of the buff	required
action	string		The kind of action this buff does	required
arguments	object	{0}	An object with the list of arguments needed for this action. Example: if "heal" arguments could be "percentage": 30, meaning the amount of health healed	required
mpCost	integer		The MP cost for this buff	required
target	string		The target of the buff. If "one" the attacks buffs one party member. If "all" the buff targets everyone in the party	required
▾ behaviors	array[object]	{3}	The "AI" of the enemy: a list of behaviors ordered by priority	required
▾ conditions	array[object]	{4}	The activation conditions for the enemy to conduct this behavior	required
field	string		The field that this behavior will watch. Example: if "hp" this behavior will activate when the "hp" reaches a certain value	required
target	string		The target that contains the field that is being watched. Can be "enemyOne", "enemyAll", "friendOne", "friendAll"	required
operator	string		The operator to be used in the comparison of the field being watched and the desired percentage value	required
percentage	string		The percentage value. Example: if the field being watched is "hp", the operator is "<=" and percentage is "30", this behavior will activate if the hp is less or equal than 30%	required
activation	integer		A percentage of activation for this behavior. If the behavior activation: 100 the behavior will always occur if the conditions allow it	required
action	string		The name of the action to take. Can be an attack or a buff	required

Figure A.10: Enemies schema

Appendix B

Complete toy example

This appendix contains the complete toy example, that corresponds to the first draft of the story that was later implemented in the game and presented to the players.

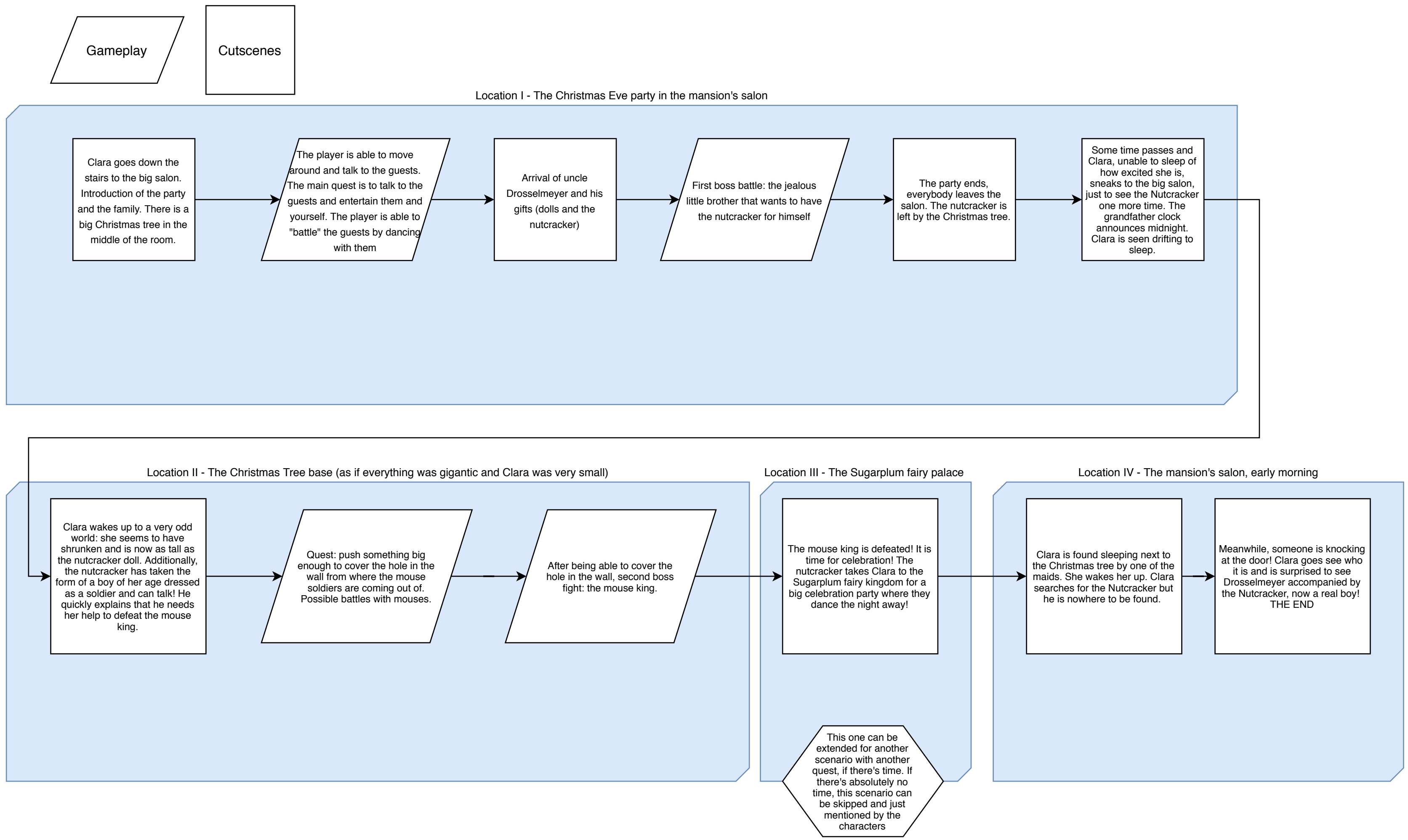


Figure B.1: Complete toy example

Appendix C

Questionnaire

This appendix contains screenshots of the questionnaires presented to the user.

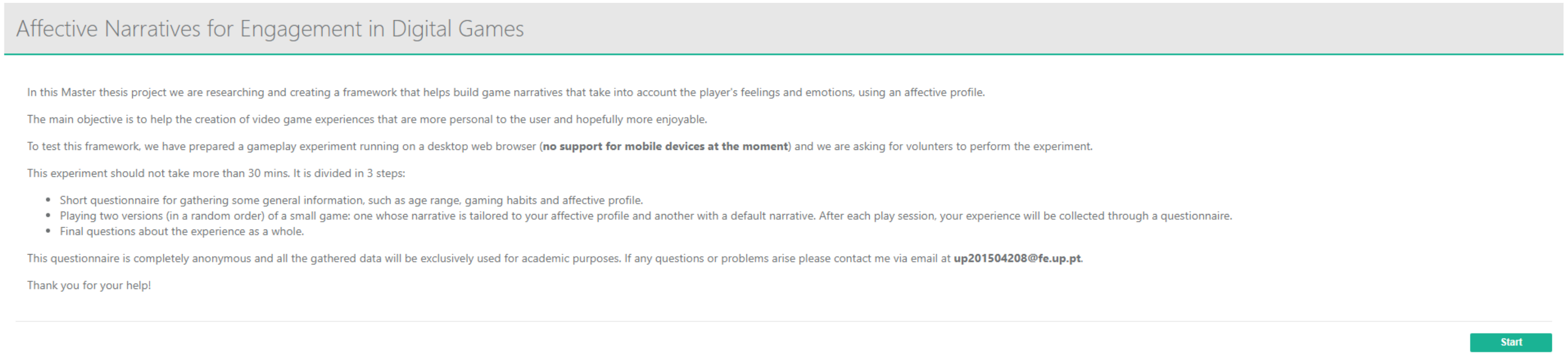
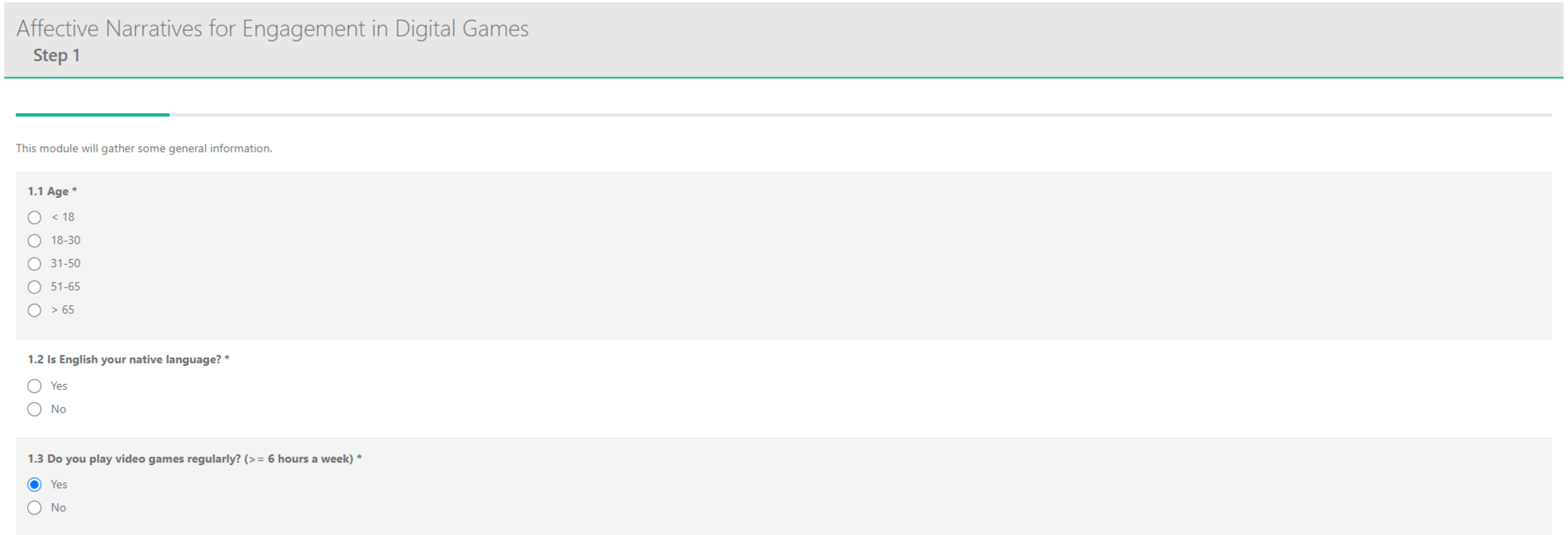


Figure C.1: Introductory page


Affective Narratives for Engagement in Digital Games
Step 1

This module will gather some general information.

1.1 Age *
- < 18
- 18-30
- 31-50
- 51-65
- > 65

1.2 Is English your native language? *
- Yes
- No

1.3 Do you play video games regularly? (>= 6 hours a week) *
- Yes
- No

Figure C.2: Page 1 of step 1 - General questions

If you answered yes to the previous question:

1.4 What is your favorite game genre? *

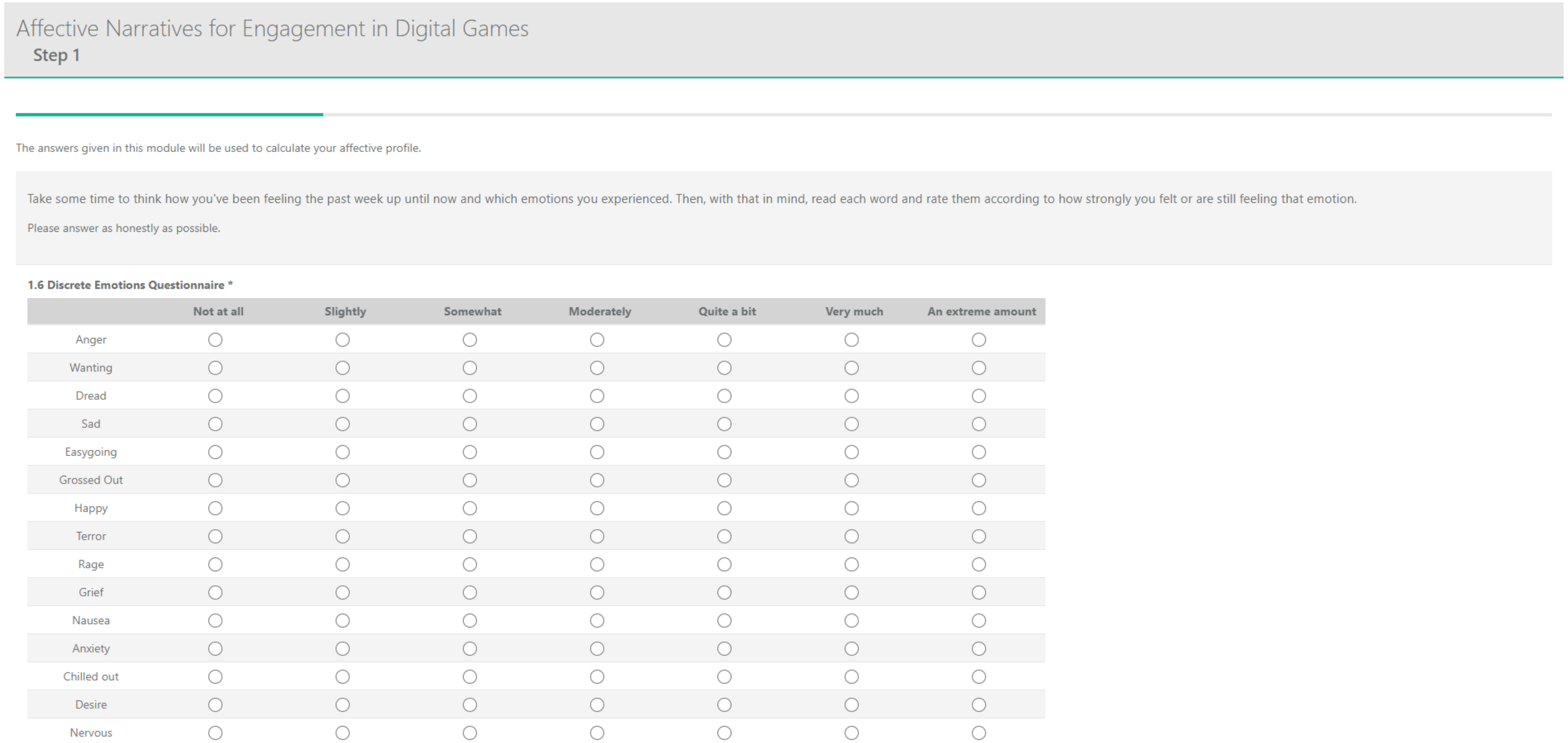
- ◯ Action games
- ◯ Shooter games
- ◯ Sports games
- ◯ Action Adventure games
- ◯ Role Playing games
- ◯ Adventure games
- ◯ Racing games
- ◯ Fighting games
- ◯ Strategy games
- ◯ Simulator games
- ◯ Other

1.5 Considering games with a substancial narrative, how important to your enjoyment of the game is its narrative to you? *

Not important at all | 1 | 2 | 3 | 4 | 5 | Extremely important

Next

Figure C.3: Page 1 of step 1 - General questions (cont.)



Affective Narratives for Engagement in Digital Games

Step 1

The answers given in this module will be used to calculate your affective profile.

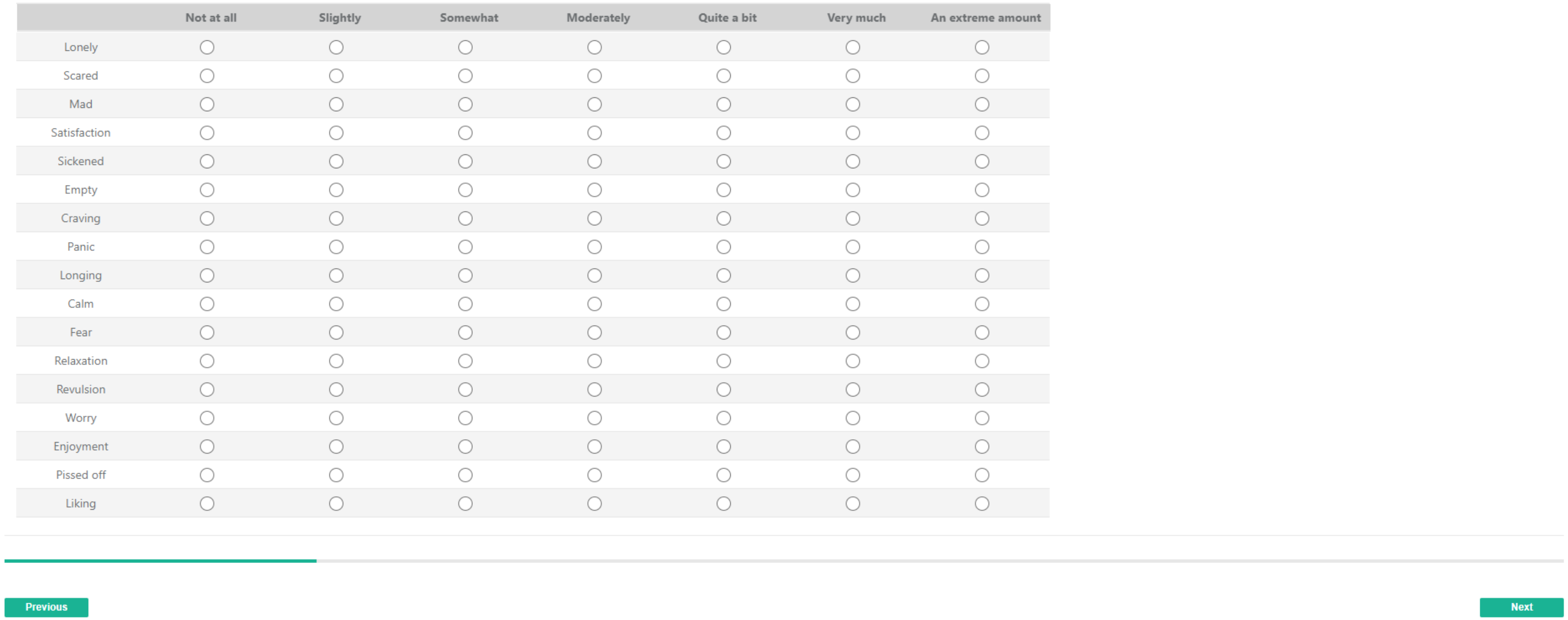
Take some time to think how you've been feeling the past week up until now and which emotions you experienced. Then, with that in mind, read each word and rate them according to how strongly you felt or are still feeling that emotion.

Please answer as honestly as possible.

1.6 Discrete Emotions Questionnaire *

	Not at all	Slightly	Somewhat	Moderately	Quite a bit	Very much	An extreme amount
Anger	○	○	○	○	○	○	○
Wanting	○	○	○	○	○	○	○
Dread	○	○	○	○	○	○	○
Sad	○	○	○	○	○	○	○
Easygoing	○	○	○	○	○	○	○
Grossed Out	○	○	○	○	○	○	○
Happy	○	○	○	○	○	○	○
Terror	○	○	○	○	○	○	○
Rage	○	○	○	○	○	○	○
Grief	○	○	○	○	○	○	○
Nausea	○	○	○	○	○	○	○
Anxiety	○	○	○	○	○	○	○
Chilled out	○	○	○	○	○	○	○
Desire	○	○	○	○	○	○	○
Nervous	○	○	○	○	○	○	○

Figure C.4: Page 2 of step 1 - DEQ



	Not at all	Slightly	Somewhat	Moderately	Quite a bit	Very much	An extreme amount
Lonely	○	○	○	○	○	○	○
Scared	○	○	○	○	○	○	○
Mad	○	○	○	○	○	○	○
Satisfaction	○	○	○	○	○	○	○
Sickened	○	○	○	○	○	○	○
Empty	○	○	○	○	○	○	○
Craving	○	○	○	○	○	○	○
Panic	○	○	○	○	○	○	○
Longing	○	○	○	○	○	○	○
Calm	○	○	○	○	○	○	○
Fear	○	○	○	○	○	○	○
Relaxation	○	○	○	○	○	○	○
Revulsion	○	○	○	○	○	○	○
Worry	○	○	○	○	○	○	○
Enjoyment	○	○	○	○	○	○	○
Pissed off	○	○	○	○	○	○	○
Liking	○	○	○	○	○	○	○

Previous

Next

Figure C.5: Page 2 of step 1 - DEQ (cont.)

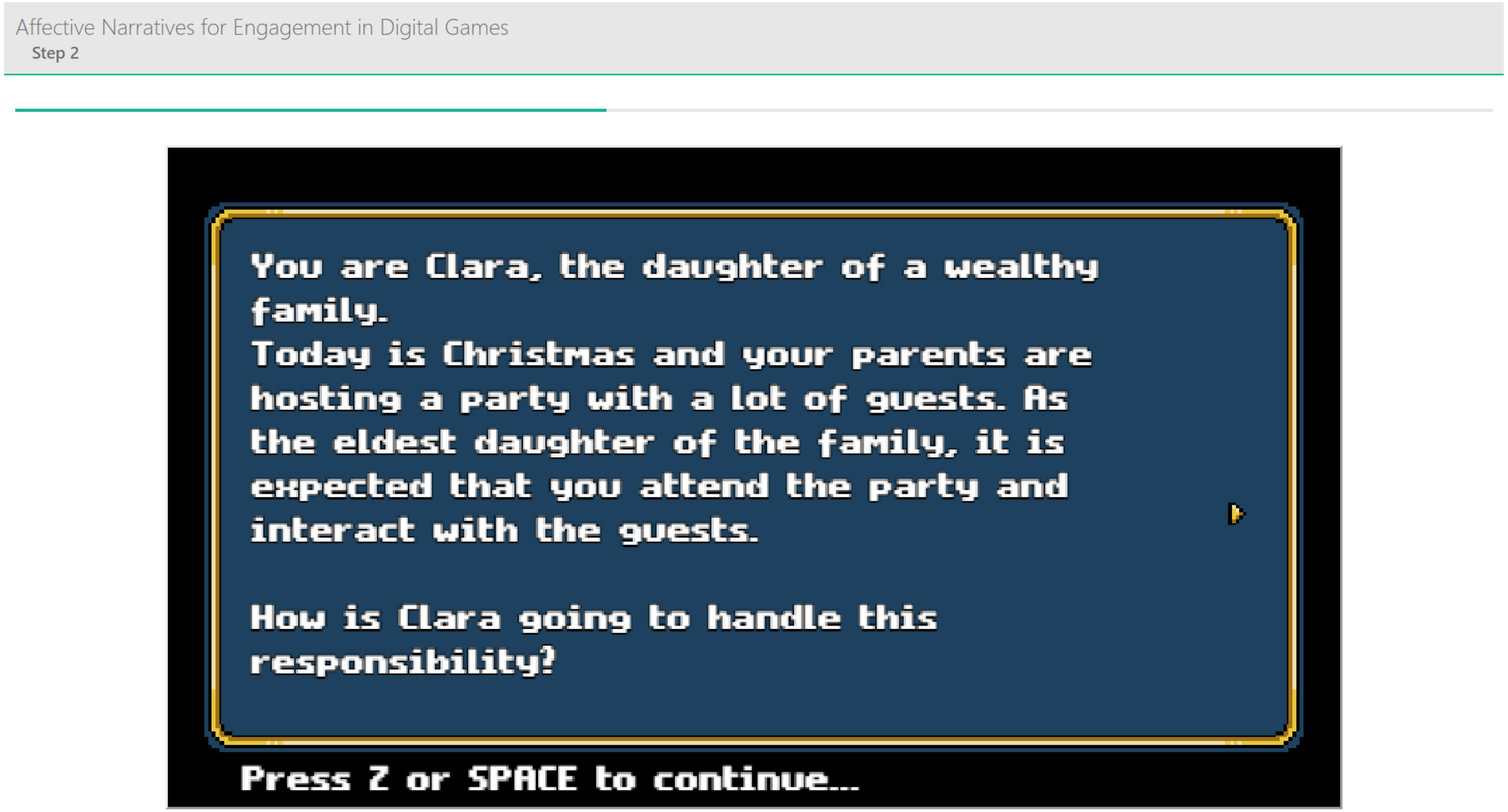


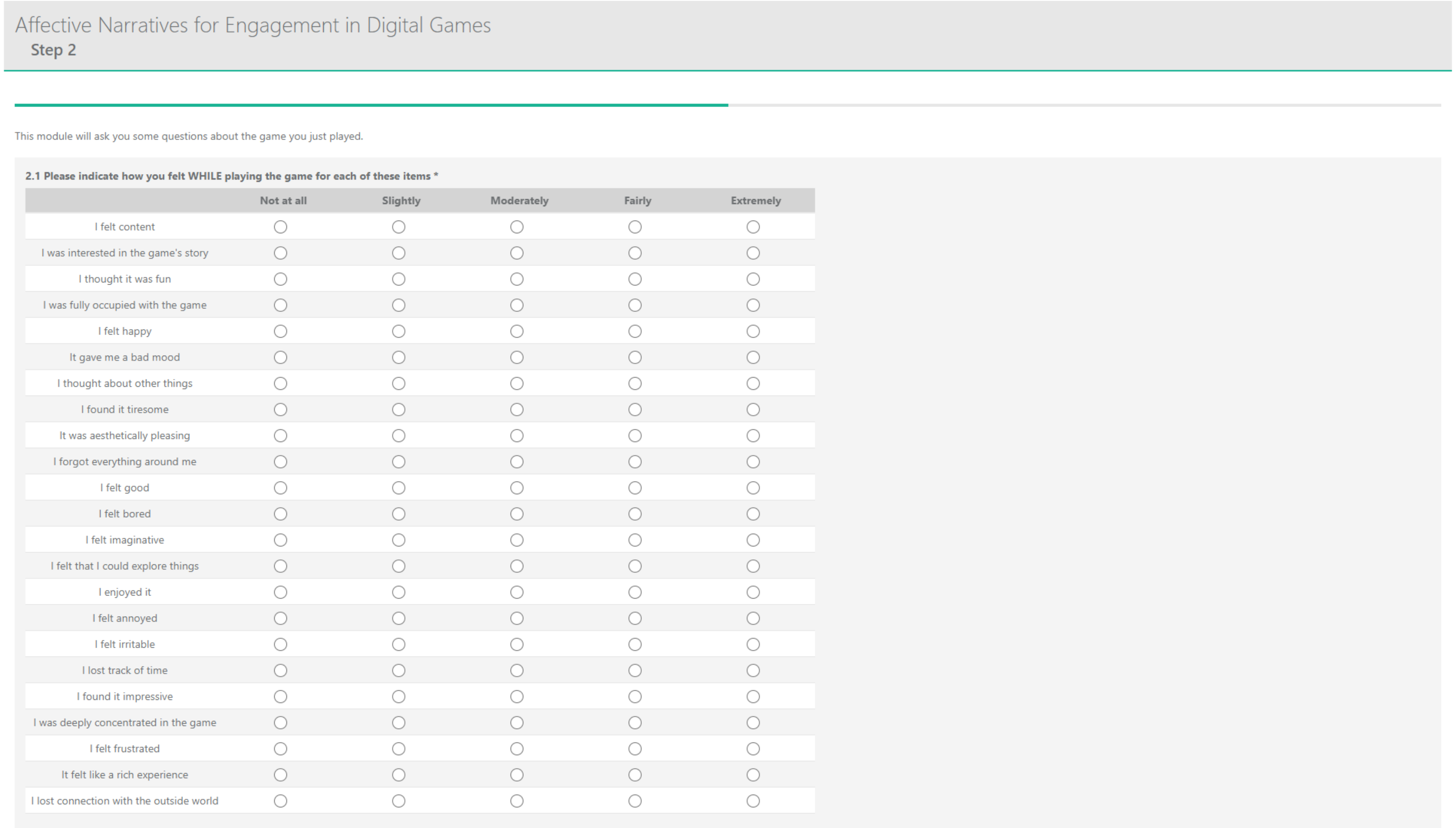
Figure C.6: Page 1 of step 2 - Game session

You are Clara, the daughter of a wealthy family. Today is Christmas and your parents are hosting a party with a lot of guests. As the eldest daughter of the family, it is expected that you attend the party and interact with the guests. How is Clara going to handle this responsibility?

Game controls

Arrows	Walking
Z	Interacting with characters whenever prompted Interacting with the dialogue box

Figure C.7: Page 1 of step 2 - Game session (cont.)



Affective Narratives for Engagement in Digital Games

Step 2

This module will ask you some questions about the game you just played.

2.1 Please indicate how you felt WHILE playing the game for each of these items *

	Not at all	Slightly	Moderately	Fairly	Extremely
I felt content	○	○	○	○	○
I was interested in the game's story	○	○	○	○	○
I thought it was fun	○	○	○	○	○
I was fully occupied with the game	○	○	○	○	○
I felt happy	○	○	○	○	○
It gave me a bad mood	○	○	○	○	○
I thought about other things	○	○	○	○	○
I found it tiresome	○	○	○	○	○
It was aesthetically pleasing	○	○	○	○	○
I forgot everything around me	○	○	○	○	○
I felt good	○	○	○	○	○
I felt bored	○	○	○	○	○
I felt imaginative	○	○	○	○	○
I felt that I could explore things	○	○	○	○	○
I enjoyed it	○	○	○	○	○
I felt annoyed	○	○	○	○	○
I felt irritable	○	○	○	○	○
I lost track of time	○	○	○	○	○
I found it impressive	○	○	○	○	○
I was deeply concentrated in the game	○	○	○	○	○
I felt frustrated	○	○	○	○	○
It felt like a rich experience	○	○	○	○	○
I lost connection with the outside world	○	○	○	○	○

Figure C.8: Page 2 of step 2 - Post game session questions (GEQ)

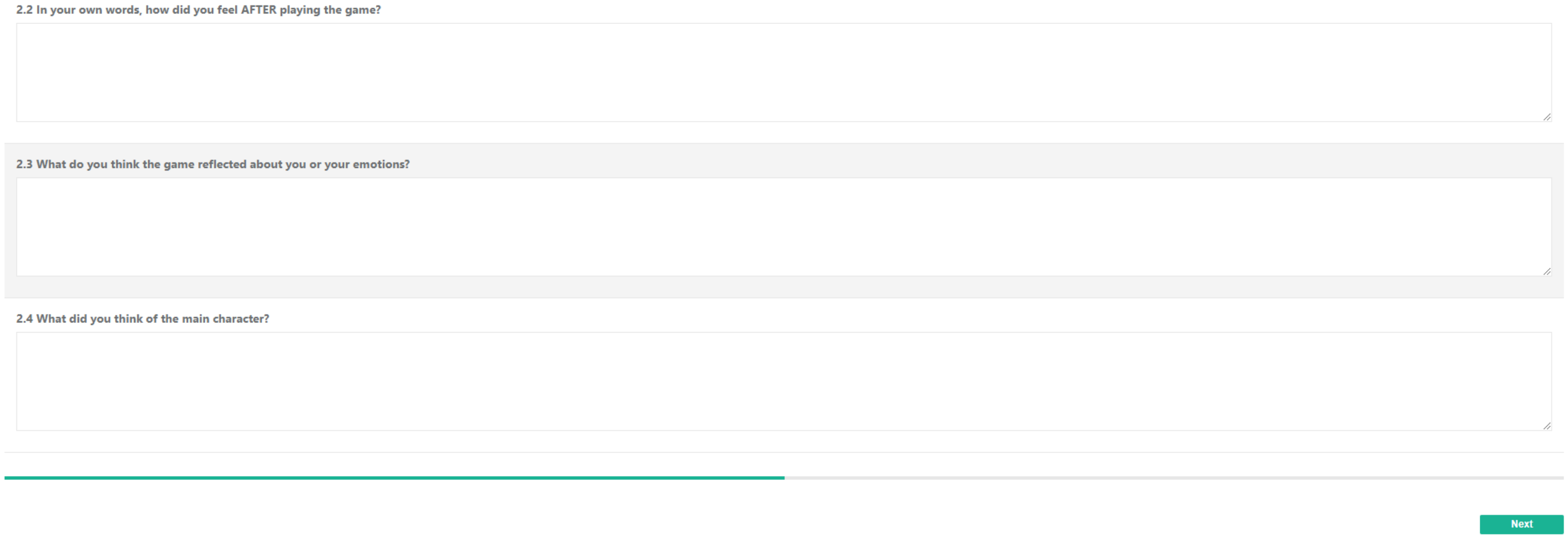
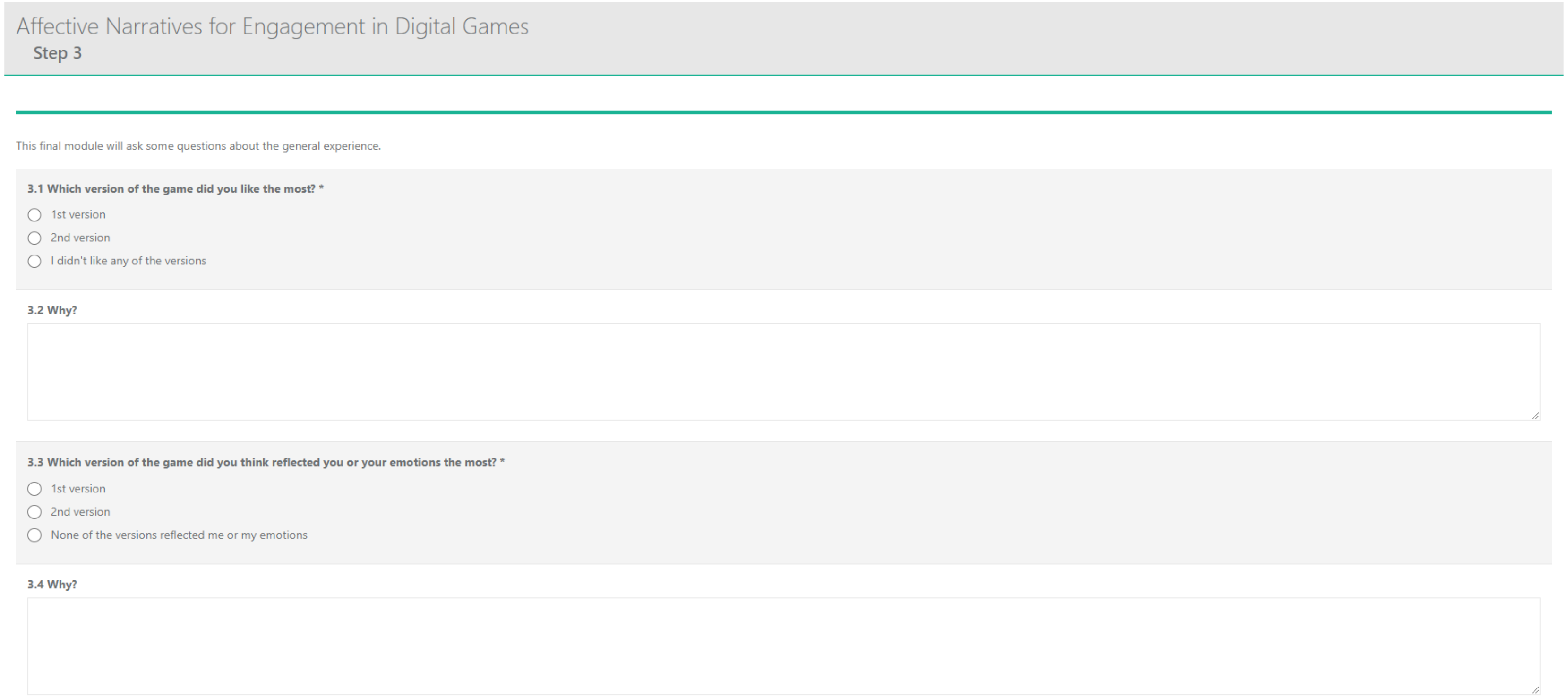


Figure C.9: Page 2 of step 2 - Post game session questions (GEQ) (cont.)



Affective Narratives for Engagement in Digital Games
Step 3

This final module will ask some questions about the general experience.

3.1 Which version of the game did you like the most? *

- ◯ 1st version
- ◯ 2nd version
- ◯ I didn't like any of the versions

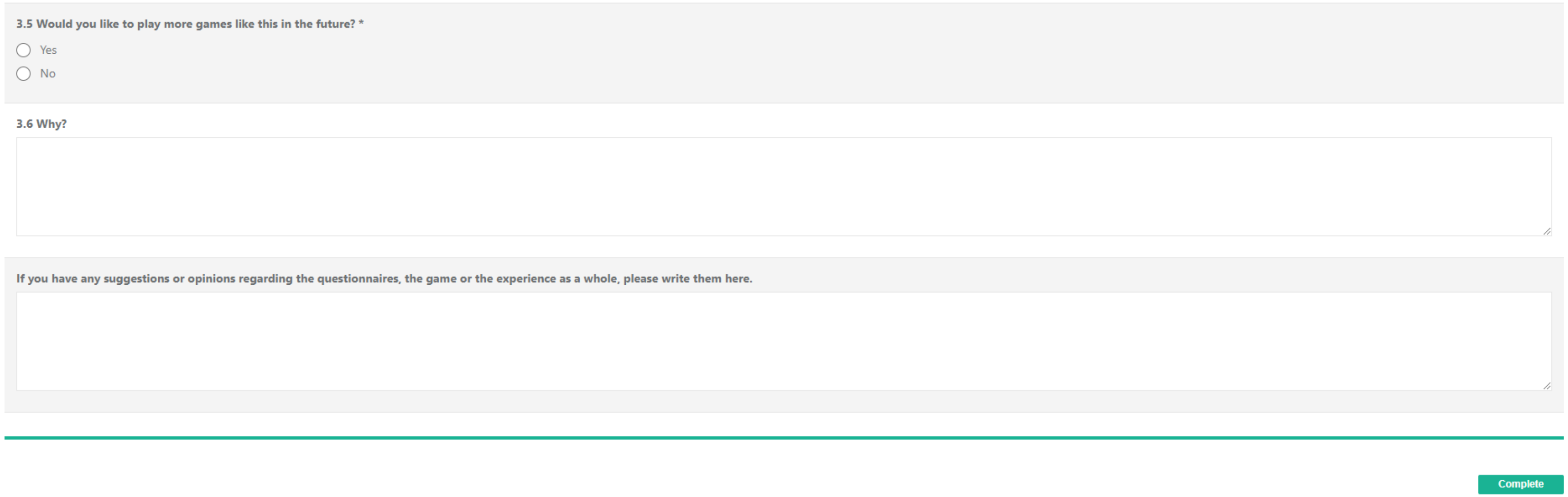
3.2 Why?

3.3 Which version of the game did you think reflected you or your emotions the most? *

- ◯ 1st version
- ◯ 2nd version
- ◯ None of the versions reflected me or my emotions

3.4 Why?

Figure C.10: Step 3 - General questions about the whole experience



3.5 Would you like to play more games like this in the future? *

- Yes
- No

3.6 Why?

If you have any suggestions or opinions regarding the questionnaires, the game or the experience as a whole, please write them here.

Complete

Figure C.11: Step 3 - General questions about the whole experience (cont.)

Appendix D

Game Narratives

This appendix contains the graph view of all the possible narratives that could be presented to a player.

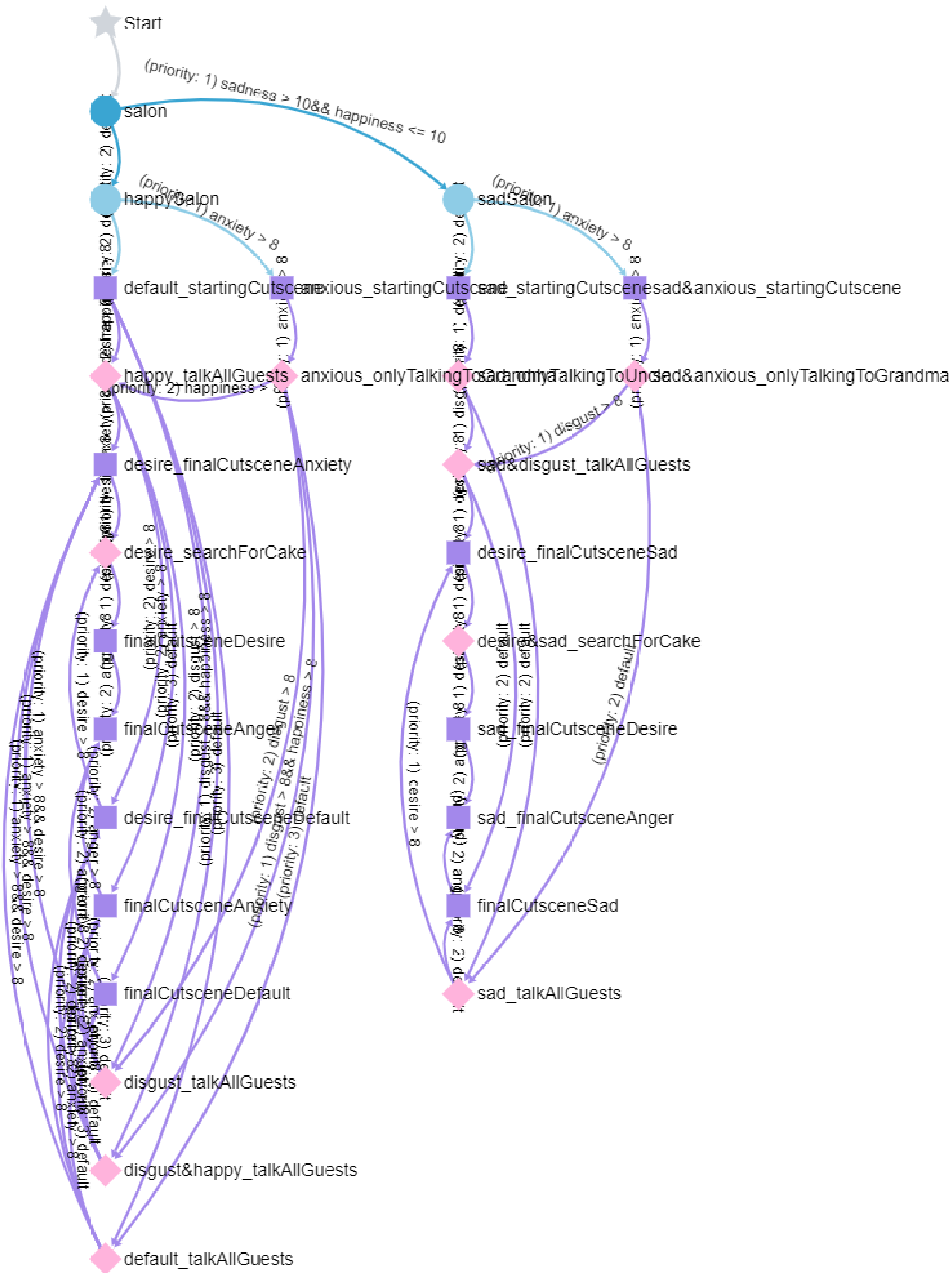


Figure D.1: View of all possible narrative branches

References

[Adams and Rollings, 2006] Adams, E. and Rollings, A. (2006). *Fundamentals of Game Design (Game Design and Development Series)*. Prentice-Hall, Inc., USA.

[Alves Nogueira, 2016] Alves Nogueira, P. (2016). *Emotional State Regulation in Interactive Environments: A Psychophysiological Adaptive Approach for Affect Inductive Experiences*. Ph.d. thesis, Universidade do Porto.

[AstralShift, 2016] AstralShift (2016). Pocket Mirror. https://astralshiftpro.itch.io/pocket-mirror. Accessed: 2020-06-11.

[Barber and Kudenko, 2007] Barber, H. and Kudenko, D. (2007). Dynamic generation of dilemma-based interactive narratives. In *Proceedings of the 3rd Artificial Intelligence and Interactive Digital Entertainment Conference, AIIDE 2007*, pages 2–7.

[Besmond, 2019] Besmond, S. (2019). *Narrative Structure of Videogames*. Bachelor's thesis, Jyväskylä University of Applied Sciences.

[Bethesda, 2011] Bethesda (2011). The Elder Scrolls | Skyrim. https://elderscrolls.bethesda.net/en/skyrim. Accessed: 2020-05-08.

[Cambridge Dictionary, 2020] Cambridge Dictionary (2020). SELF-REFLECTION | meaning in the Cambridge English Dictionary. https://dictionary.cambridge.org/dictionary/english/self-reflection. Accessed: 2020-03-10.

[Caminha, 2017] Caminha, D. C. C. (2017). *Development of Emotional Game Mechanics through the use of Biometric Sensors*. Dissertation, Universidade do Porto.

[Campbell, 1949] Campbell, J. (1949). *The Hero with a Thousand Faces*. Princeton University Press.

[Capcom, 2001] Capcom (2001). Devil May Cry series. http://www.devilmaycry.com/. Accessed: 2020-06-19.

[Cobley, 2001] Cobley, P. (2001). *Narrative. The New Critical Idiom*. Routledge, New York.

[Crockford, 2018] Crockford, D. (2018). Introducing JSON. https://www.json.org/json-en.html. Accessed: 2020-06-22.

[de Lima et al., 2018] de Lima, E. S., Feijó, B., and Furtado, A. L. (2018). Player behavior and personality modeling for interactive storytelling in games. *Entertainment Computing*, 28(March):32–48.

[Enterbrain, 2015] Enterbrain (2015). RPG Maker. https://www.rpgmakerweb.com/. Accessed: 2020-06-11.

[From Software, 2011] From Software (2011). Dark Souls series. https://en.bandainamcoent.eu/dark-souls. Accessed: 2020-06-19.

[Fullbright, 2013] Fullbright (2013). Gone Home. https://gonehome.game/. Accessed: 2020-03-10.

[Giant Sparrow, 2017] Giant Sparrow (2017). What Remains of Edith Finch. https://store.steampowered.com/app/501300/What_Remains_of_Edith_Finch/. Accessed: 2020-03-10.

[Goldberg, 1990] Goldberg, L. R. (1990). An Alternative "Description of Personality": The Big-Five Factor Structure. *Journal of Personality and Social Psychology*, 59(6):1216–1229.

[Haag et al., 2004] Haag, A., Goronzy, S., Schaich, P., and Williams, J. (2004). Emotion recognition using bio-sensors: First steps towards an automatic system. In *Affective Dialogue Systems*, pages 36–48, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Harmon-Jones et al., 2016] Harmon-Jones, C., Bastian, B., and Harmon-Jones, E. (2016). The discrete emotions questionnaire: A new tool for measuring state self-reported emotions. *PLoS ONE*, 11(8):1–25.

[Heath, 1996] Heath, M. (1996). *Aristotle poetics*. Penguin Books, London.

[Hernandez et al., 2015] Hernandez, S. P., Bulitko, V., and Spetch, M. (2015). Keeping the player on an emotional trajectory in interactive storytelling. In *Proceedings of the 11th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE 2015*, volume 2015-Novem, pages 65–71.

[Hogan, 2011] Hogan, P. C. (2011). *Affective Narratology*. University of Nebraska Press.

[Houlette, 2003] Houlette, R. (2003). Player Modeling for Adaptive Games. In *AI Games Programming Wisdom 2*, pages 557–566.

[House House, 2019] House House (2019). Untitled Goose Game. https://goose.game/. Accessed: 2020-03-10.

[Hudlicka, 2008] Hudlicka, E. (2008). Affective computing for game design. In *4th International North-American Conference on Intelligent Games and Simulation, Game-On 'NA 2008*, pages 5–12.

[IJsselsteijn et al., 2013] IJsselsteijn, W., de Kort, Y., and Poels, K. (2013). *The Game Experience Questionnaire*. Technische Universiteit Eindhoven.

[Ip, 2011] Ip, B. (2011). Narrative structures in computer and video games: Part 1: Context, definitions, and initial findings. *Games and Culture*, 6(2):103–134.

[Itterheim and Löw, 2012] Itterheim, S. and Löw, A. (2012). *Working with Tilemaps*, pages 265–290. Apress, Berkeley, CA.

[Jenkins, 2004] Jenkins, H. (2004). Game Design as Narrative Architecture. In Wardrip-Fruin, N. and Harrigan, P., editors, *First person: new media as story, performance, and game*, pages 118–130. MIT Press, Cambridge, Mass.

[Kaptein and Broekens, 2015] Kaptein, F. and Broekens, J. (2015). The affective storyteller: Using character emotion to influence narrative generation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9238, pages 352–355. Springer Verlag.

[Khan Academy, 2017] Khan Academy (2017). Introduction to storytelling (video). https://www.khanacademy.org/humanities/hass-storytelling/storytelling-pixar-in-a-box/ah-piab-we-are-all-storytellers/v/storytelling-introb. Accessed: 2020-02-08.

[Kouri, 2012] Kouri (2012). Ib. https://vgperson.com/games/ib.htm. Accessed: 2020-06-11.

[Laws, 2002] Laws, R. D. (2002). *Robin's Laws of Good Game Mastering*. Steve Jackson Games.

[Li and Riedl, 2010] Li, B. and Riedl, M. O. (2010). An offline planning approach to game plotline adaptation. *Proceedings of the 6th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE 2010*, (January):45–50.

[Lopes and Bidarra, 2011] Lopes, R. and Bidarra, R. (2011). Adaptivity challenges in games and simulations: A survey. *IEEE Transactions on Computational Intelligence and AI in Games*, 3(2):85–99.

[Mortis Ghost, 2008] Mortis Ghost (2008). Off. https://off.fandom.com/wiki/Downloads. Accessed: 2020-06-11.

[Naughty Dog, 2013] Naughty Dog (2013). The Last of Us. https://www.playstation.com/en-us/games/the-last-of-us-ps3/. Accessed: 2020-03-10.

[Oliver et al., 2016] Oliver, M. B., Bowman, N. D., Woolley, J. K., Rogers, R., Sherrick, B. I., and Chung, M.-Y. (2016). Video games as meaningful entertainment experiences. *Psychology of Popular Media Culture*, 5(4):390–405.

[Peng et al., 2018] Peng, F., Labelle, V. C., Picard, R. W., and Yue, E. C. (2018). A trip to the moon: Personalized animated movies for self-reflection. In *Conference on Human Factors in Computing Systems - Proceedings*, volume 2018-April. Association for Computing Machinery.

[Porter Abbott, 2014] Porter Abbott, H. (2014). *The Cambridge introduction to narrative, second edition*. Cambridge University Press, Cambridge.

[Rabin, 2013] Rabin, S. (2013). *Introduction to Game Development*. Course Technology Cengage Learning.

[Rammstedt and John, 2007] Rammstedt, B. and John, O. P. (2007). 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(1):203–212.

[Riopel, 2020] Riopel, L. (2020). What is the Positive and Negative Affect Schedule? (PANAS). https://positivepsychology.com/positive-and-negative-affect-schedule-panas/. Accessed: 2020-06-19.

[Santa Monica Studio, 2018] Santa Monica Studio (2018). God of War. https://godofwar.playstation.com/. Accessed: 2020-03-10.

[Searight and Montone, 2017] Searight, H. R. and Montone, K. (2017). Profile of Mood States. In Zeigler-Hill, V. and Shackelford, T. K., editors, *Encyclopedia of Personality and Individual Differences*, volume 2012, pages 1–6. Springer International Publishing, Cham.

[Sharma et al., 2010] Sharma, M., Ontañón, S., Mehta, M., and Ram, A. (2010). DRAMA MANAGEMENT AND PLAYER MODELING FOR INTERACTIVE FICTION GAMES. *Computational Intelligence*, 26(2):183–211.

[Sobolev, 2019] Sobolev, J. (2019). What Are Sprites And How They Work In Games? https://gamingshift.com/sprites-in-games/. Accessed: 2020-06-29.

[Thue et al., 2007] Thue, D., Bulitko, V., Spetch, M., and Eric, W. (2007). Interactive storytelling: A player modelling approach. In *Proceedings of the 3rd Artificial Intelligence and Interactive Digital Entertainment Conference, AIIDE 2007*, number July, pages 43–48.

[Wei, 2011] Wei, H. (2011). *Analyzing the game narrative: Structure and technique*. Dissertation, Simon Fraser University.