A method for procedurally generated narratives in video games

Lars Hagen, Medialogy Master Thesis.

Aalborg University Copenhagen

Semester: 10th

Title: A method for procedurally generated narratives in video games

Project Period: Spring 2014

Semester Theme:

Aalborg University Copenhagen
A.C. Meyers Vænge
2450 København SV, Denmark

Semester Coordinator:

Secretary:

Supervisor(s): Georgios Triantafyllidis

Project group no.:

Members:

Lars Strange Hagen

Abstract:

This project focus on procedurally generated narratives for video games. The area shows a lot of potential for game developers as it can decrease production cost, decrease production time and increase the replayability of the game.

In this project I focus specifically on procedurally generating narratives for roleplaying games. The product consists of two parts. There is the storyengine that will generate the stories and then there is the game itself that will execute the story. The storyengine generates the stories based on the hero's journey.

In this report I will investigate related works. I will analyse theory relating to the project and I will describe the design and implementation of the storyengine and game.

The product was tested on 29 test participants and evaluated based on the results. The storyengine in its current state does not create stories that are long and deep enough for the players to really get involved. My test results indicates that procedurally generated narratives like the ones produced by the storyengine can in fact improve replayability, but nothing can be concluded.

Copies: 3

Pages: 46

Finished: 28th of may 2014

CONTENTS

1 INTRODUCTION

In the recent years the subject of interactive narratives has been getting notable attention form games theorists and industry professionals (Ip, 2011). Video games today almost always include some sort of story, it might range from the very simple such as in Super Mario where Mario has to save the princess, to the hugely complex stories in games such as Bioshock, Mass Effect and The Walking Dead.

There is a reason for this, stories can add entertainment values to help keep the players interested in the game for longer time and it helps selling the game (Adams, 2010). Many players want stories in their games and many genres require one, including genres such as Role Playing Games, Action-Adventure and Adventure Games (Adams, 2010). It can provide a greater emotional satisfaction when the player progresses towards a dramatically meaningful goal rather than an abstract one (Adams, 2010). Even games with low emphasis on story such as Space Invaders, Super Mario, Sonic the Hedgehog and Street Fighter all contain simple and appealing stories used to attract the attention of the players (Ip, 2011).

Stories can also provide emotional richness. A game in its bare format can only provide a few basic emotions such as success, frustration, determination and perhaps an 'aha' moment, but deeper emotions can only come when the player can identify with characters and their problems, this can only happen in a well written story. If the game wants to invoke a greater variety of emotions than the basics, a story is needed (Adams, 2010).

Role playing games are content heavy games. They require a large world, sophisticated storylines and numerous side-quests. Due to their nature the replay value drops off significantly after the gameplay affordances of the story and world are exhausted. (Hartsook, Zook, Das, & Riedl, 2011). Having played one story the player may demand a new one, therefore the ability to generate customized plotlines may enhance replayability and improve player experience. Plotlines generated from scratch has long been the goal of story generation. (Boyang & Riedl, 2010)

The term Procedural Content Generation (PCG) refers to the creation of game content automatically through algorithmic means. Included in this definition are assets such as terrain, maps, levels, stories, dialogue, quests, characters, rulesets, camera viewpoint, and weapons. (Togelius, Yannakakis, Stanley, & Browne, 2012) Procedural content generation may make the creation of content-heavy games such as roleplaying games cheaper by semi-automating the construction of some game content. (Hartsook, Zook, Das, & Riedl, 2011)

Game developers can have an interest in procedural content generation for multiple reasons. Game production usually require a significant amount of time and money for content creation and in comparison the consumption of said content is usually much faster (Boyang & Riedl, 2010). Procedural content generation can help avoid the huge expenses associated with manually creating game content (Togelius, Yannakakis, Stanley, & Browne, 2012).

We hypothesize that techniques from procedural content generation can be used to generate stories for role playing games to enhance replayability.

1.1 Related works

There seems to be two overall approaches to generating narratives. First approach is to generate the stories as the game is progressing while the player plays the game, I will call these systems for runtime

systems. The other approach is to generate the story before the player starts playing the game, I will call these 'pre-game systems' as they work before the player starts playing.

Runtime systems are often simulations, trying to simulate human behavior in non-player characters (NPCs) to generate a narrative through their actions. Runtime systems usually either try to solve the narrative paradox, where the narrative can go in unlimited directions depending on the choices the user makes in a truly interactive world, or they will try to steer the user back on to the story when he ventures to far from it. Pre-game systems generates stories before the player starts playing and can therefor use techniques closer to that of a human author where the story is planned out beforehand. Lots of research seems to have been conducted in runtime systems and less in pre-game systems.

When talking about runtime systems, Façade is the prime example. In Façade the player takes the role of the friend of the couple Grace and Trip. The game takes place in Grace and Trips apartment and the player can engage in conversation with either of them to eventually unveil that they have marital problems. The user can type in anything he wants to say and the game can understand and make Grace and Trip respond thanks to sophisticated natural language processing systems. Grace and Trip will try to steer the conversation back onto the underlying plot when the conversation moves to far away from it (Mateas & Stern, 2003) (Mateas & Stern, 2005).

The plot in façade is static, what changes is the way the plot is unveiled to the player. Other research has been done in generating new stories with new plots using runtime systems. One such example is the work by Charles, Mead and Cavazza (2001).

Charles et al. developed a system that generates stories based on the realtime interaction between autonomous actors (or NPCs). Each actor has a goal and need to interact with the world and the other actors to reach his goal. The specific implementation by Charles et al generates sitcom-like narratives. They determined that even though characters individual actions are deterministic, the interaction between them creates variability. The character centered approach also has the advantages of being modular and easy to expand with more actors. (Charles, Mead, & Cavazza, 2001)

One of the major problems with these kind of runtime simulations is creating interesting and coherent plots. Sgouros et al tries to solve this using what they call a 'plot manager' in their project. (Sgouros, Papakonstantinou, & Tsanakas, 1996)

The plot manager shapes the main character's interaction with the story by controlling the cast members and specifying what the main character can do. It has a set of rules for social interaction and specifications for each character role in the story. Like Charles et al's system, the plot is shaped by the interference between agents and their social interaction with each other and the player. To try and keep the plot coherent and interesting the plot manager accepts a 'story map' as input. A story map has several points each has a 'plot structure' associated with it. Each plot structure contains a list of the local cast members and their relations and is activated when the user reaches the point. Pair of points also have 'transfer plans' that describes conditions and required resources for moving between them. The player can move between plot points trying to reach he's goal while cast members will either try to help him or stop him depending on their own goals and roles, they might also interact with other cast members. The story is shaped dynamically as the game progresses. (Sgouros, Papakonstantinou, & Tsanakas, 1996)

Figueiredo et al., also developed a game that tries to tackle the issue with incoherent plots in runtime systems. At the heart of their game is the 'story facilitator', it has the same function a dungeon master has in classic table top role playing games.

At the start of the game the story facilitator loads all the pre written episodes for the story and puts them in the 'story memory'. An episode is a part of the story and can be combined with other episodes where each combinations produces a different overall story. An episode has a list of preconditions, a list of participants (objects and characters with goals), a list of narrative intro/setup actions, a list of finish conditions and a list of 'triggers' with narrative actions and conditions. (Figueiredo, Brisson, Aylett, & Paiva, 2008)

The story facilitator selects a random episode with an empty set of preconditions and executes the narrative actions in the episodes intro. Each time a character that is not controlled by a human is inserted on the set, the story facilitator reads the character's properties and stores them in the story memory and then sends the character's goals for the episode to the agent controlling the character. When all narrative intro actions has been executed the story facilitator hands the control to the agents and user to control the characters and make the story from there on. During this the story facilitator records all actions performed by the agents and the user and if the conditions for a trigger is fulfilled the story facilitator executes the narrative actions for that trigger. If all finish conditions is fulfilled the story facilitator ends the episode and looks to find a new episode with satisfied preconditions, if the story facilitator can't find any it ends the story. (Figueiredo, Brisson, Aylett, & Paiva, 2008)

So far we have been looking at some examples of how runtime systems can be used to generate narratives in video games. Below we will look into how pre-game systems can do it.

Infinitale is my 2nd semester Master project. In this project we also tried to procedurally generate narratives for video games. The system takes a number of prewritten stories as input and chops them into sections that are used as building blocks and stitch them back together in new combinations to create a new story (Hagen & Thaves-Warnsdorf, 2013). The system produced relatively satisfying stories but a lot of work is required in writing the stories used as input as the authors needs to make sure the input stories are compatible with one another. (Hagen & Thaves-Warnsdorf, 2013)

Another project much more successful in implementing procedurally generated narratives is Quicksilver: Infinite story by M. J. Sennott. It's a commercial game currently in the development that started as a Master thesis project (Sennott, 2012). Quicksilver: Infinite Story generates a new independent story for each level, or episodes as he calls them. Independent meaning that they don't have to be played chronologically. The stories are in the style of classic action adventure cartoons. The stories are generated based on a custom morphology like the hero's journey, but for action adventure cartoons. The algorithm begins by making high level decisions about the structure of the story and works its way down to finer details. For example:

- What is the nature of the hero's conflict with the villain in the episode?
    - Battle – The heroes are introduced to a problem in the opening scenes, and they know the villain is responsible. They spend the episode trying to find and defeat this villain.
- What are the stakes of the conflict?
    - Town – The well-being of a town the heroes visit is at stake.
- What is the nature of this conflict over this town?
    - Oppression – The villain has taken over the town, and proves violently tyrannous.
- Why was the villain able to do this? Why is he so dangerous?
    - Weapon – The villain uses a mysterious artifact as a powerful weapon.

When the plot has been generated is uses a similar process to generate details about the characters and the setting of the episode. (Sennott, 2012)

1.2 Final problem statement

Table 1 shows the projects discussed in the related works section divided into groups depending on what type of system it is and how much human authored input they require.

	Runtime system	**Pre-game system**
Significant authored input	• Façade • Sgouros et al. • Figueirdo et al.	• Infinitale
Minimum authored input	• Charles et al.	• Quicksilver

Table 1: Projects discussed in related works divided into groups depending on if they are runtime/pre-game system and if they require minimum/maximum human authored input.

In the introduction, it was argued that procedurally generated narratives can save time and money, but this argument is rendered void if a significant amount of human authored input is required. These systems might bring a host of other advantages though. This is why I have made a distinction between minimum and significant authored input was made in Table 1.

As we can see above, Façade is a runtime system that requires a significant amount of human authored input. Façade is great at giving the user the illusion of narrative control and guiding the user back on track when moving to far away from the plot, however the system needs heavy modifications if it should be able to procedurally generate plots and storylines.

Sgouros et al., Figueiredo et al., and Charles et al. all have a simulated world. The advantage is a high degree of freedom at the cost of narrative coherence. Sgouros et al. and Figueiredo et al. try to increase the narrative coherence by including 'story maps' or pre-written narrative events, this in turn increases the production cost.

My old project, Infinitale, generated narratives by combining parts of different stories into one new story. In order for this to work the stories used as input all had to be compatible with each other in style and structure. The easiest way to do this was to keep the stories simple and as a result, the generated stories where simple as well. Because the stories all need to be compatible with each other, it becomes exponentially harder to write each new story. This system is capable of producing simple stories with high coherence.

Quicksilver is a pre-game system with minimum authored input. It's capable of producing episodic narratives but is unable to create larger story arcs spanning multiple episodes.

Using the information gathered in related works, I created two lists with pros and cons for runtime/pre-game systems and Significant/minimum authored input in Table 2 and Table 3.

	Significant authored input	**Minimum authored input**
Pros	Higher narrative coherence	Lower costs Easier to expand
Cons	Higher production costs Harder to expand	Hard to get narrative coherence

Table 2: The pros and cons of significant and minimum authored input

	Runtime system	**Pre-game system**
Pros	High narrative control for player	Can mimic human authors Higher narrative coherence
Cons	Narrative incoherence	Linear narrative

Table 3: The pros and cons of runtime and pre-game systems

If we take a look at our initial hypothesis, that techniques from procedural content generation can be used to generate stories for role playing games to enhance replayability, we see that one of the important keywords is replayability. Since a system with minimum authored input is easier to expand it might be easier to produce a wider range of stories thus increasing replayability even more compared to a system that requires a significant amount of human authored input. If the system requires significant authored input the player might lose interest when the content of the input material has been exhausted. When relying on less authored input the system might be capable of producing more content with a greater variety.

This leads us to the final problem statement:

Can replayability in roleplaying games be enhanced by procedurally generating the narrative for the game using a pre-game system?

2 METHOD

The product consists of two parts, the storyengine and the game. The role of the storyengine is to generate the story to be used in the game and it works completely independent of the game itself. The role of the game is to take the story output from the storyengine and use that in the game.

The product is developed using a classic waterfall approach. To help me design the product I will in the following chapter analyze various topics relating to the final problem statement. In the next chapter I will investigate terminologies and concepts relating to narratives, both interactive narratives and narratives in the classic sense. I will investigate different models used when writing narratives, and I will investigate common archetypes used in narratives. I will also be investigating the roleplaying games genre, I will figure out what defines this genre. In the analysis I will also investigate various procedural content generation techniques that can be used in the development of the storyengine.

In the design chapter I will go through and investigate how the storyengine and the game should be designed. I will go through the method used in the storyengine to generate the narratives and the overall design of the game, including the graphics and gameplay design.

In the implementation chapter I will explain some of the overall technical aspects of the game and storyengine, such as what technologies where used. The implementation of the game and the storyengine is too big to go through in detail, so instead some of the more interesting parts of the implementation will be selected and explained. There will be a higher focus on the storyengine than the game as it is more relevant to the project. Unfortunately, due to time constrains I was unable to fully implement the game as envisioned, so it's unable to load and understand the output from the story engine. Instead, I used the storyengine to generate a single random story and manually implemented that into the game.

In the test chapter I describe the test procedure, and in the results, discussion and conclusion I go through the results from the test and evaluate the product and discuss if it fulfilled the final problem statement.

3 ANALYSIS

To aid us in the design of the storyengine we will in the analysis investigate narratives, including terminology, concepts, structures etc. We will also investigate the fundamentals of the roleplaying game genre, and various procedural techniques that can be used in the game.

3.1 Narratives

Before we can design a method for procedurally generating narratives we need to have a solid understanding of what a narrative is, what it consists of, and how they are structured. In the following chapter we will go through terminology, concepts, tools and frameworks relating to narratives and in the end conclude on how this can be used in our storyengine,

3.1.1 Terminology and concepts

According to Adams the loosest definition of a story is an account of a series of events, either historical or fictitious. (Adams, 2010)

This is rather close to the definition by Barry Ip:

> *"[…] a sequence of events involving entities. A story is bound by the laws of time; it goes in one direction, starting at the beginning, moving through the middle, and arriving at the end … the only time involved is the time it takes to read [the story], and the only order is that of the structure of the essay" (Ip, 2011)*

Adams says that the minimum requirements a story must meet is to be credible, coherent and dramatically meaningful. **Credible** means that the player must be able to believe in the story. In fiction they may have to suspend some disbelief to find the story credible. **Coherent** means that all events must be relevant for the story and must "harmonize to create a pleasing whole." It is okay if some events are not related to the story or are just there to create effect or "color", but they still have to belong in the story. **Dramatically meaningful** means that the events in the story has to include something or someone that the player cares about. The story must be constructed in such a way so that the player takes an interest or even better, identify with one or more of the story's characters. All events in the game must contribute to the players involvement in the story through identification with characters and interest in what happens to those characters. (Adams, 2010)

The narrative is different from the story. Ip says that elements of time and the sequence of events are controlled by the structure of the story and the causation and links between these events are described by the plot, but the narrative determines how these events are expressed, what order they are told in, the duration of each event and the frequency of the events (Ip, 2011). In other words, the plot control the causation between the events, the story control what events are being told and what order they happen in and the narrative control how the story is told (choice of words, point of view, flashbacks etc.)

Adams' definition of narrative in relation to video games:

> *"The term narrative refer to story events that are narrated – that is, told or shown – by the game to the player. Narrative consists of the noninteractive, presentational part of the story." - (Adams, 2010)*

He also states that the primary function of a narrative in a video game is to present events over which the player has no control.

Games often uses blocks of noninteractive narrative contents. A narrative block can be presented in many ways, a pre-rendered movie, a cut scene using the game engine, voice-overs, a monolog by a character or just text. Adams doesn't consider individual lines of dialog narrative block, a noninteractive dialog on the other hand is.

Ip says that several techniques are used to strike a balance between prescribed narrative and interaction, the most basic being the background story. The background story give a simple description of the game's environment, key characters and main objectives. He also says that simple games do not require extensive back stories and they are only used as a basic method of contextualizing a game's objective. (Ip, 2011)

The, perhaps, most common method for narration in interactive video games is the cut scene. Cut scenes are often seen in more complex games where they are used to complement the back story and are gradually becoming the standard of narration in modern games. Older games may resort to delivery via on-screen text. (Ip, 2011)

The definitions and information by Ip and Adams builds on the assumption that video games can be (or contain) a narrative, not everybody agrees with that. Aarseth believes that a narrative consists of two main parts, story (the content of the narrative) and discourse (how the story is told). A game can contain a story, but the interactive nature of video games means that they cannot contain a discourse (Aarseth, 2012). The story consists of two parts, existents (things that exists in the story) and events. The events can be viewed as the fabula, that is the chronological order of events or as the plot, that is the order the events are presented to the player (Aarseth, 2012).

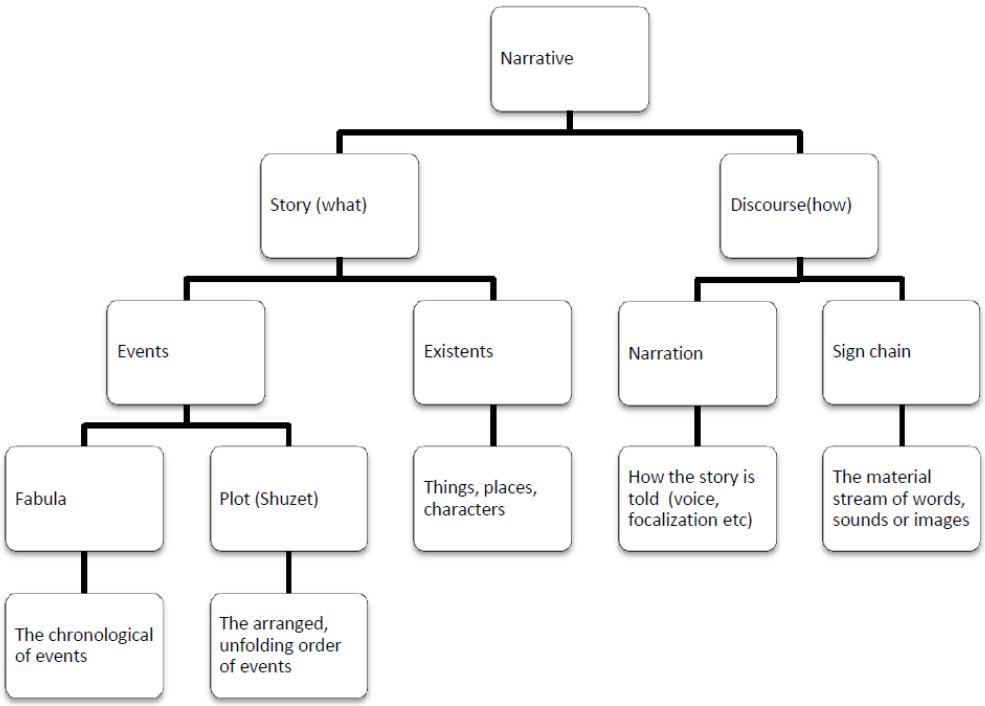


Figure 1: Tree diagram of the elements a narrative is made up of

Aarseth says that there are two types of events, kernels and satellites. Kernels are important for the story, satellites are less important and can be changed without changing the story (Aarseth, 2012). An example

for a Kernel is when Cinderella decides to go to the party at the castle, if we change it so that she doesn't go, we will no longer recognize the story. An example of a satellite would be whether Cinderella loses the shoe on her left or right foot.

3.1.2 Interactive narratives

They say that there are two interactive story systems, story graphs and simulated worlds. Story graphs are a linear story where the player follows a path between predefined episodes, this means that they are only minimally interactive (the half life series). A simulated world is the opposite. Here the player can freely interact with computer-simulated characters in a virtual environment. This often creates incoherent worlds and it does not have a temporal structure that could classify as a story (the sims series) (Sgouros, Papakonstantinou, & Tsanakas, 1996). Sgouros et al doesn't mention this but of course a game doesn't have to belong to any of the two extremes but can fall somewhere in between.

Adams define interactive stories as a story that the player interacts with by contributing actions to it. A story may still be interactive even if the player's actions cannot change the direction of the plot. It's different from other definitions where the player must have agency so that he can change the direction of the plot.

According to Adams, an interactive story has three types of events. Player events, In-game events and Narrative events. Player events are the event caused by player actions, they can either affect the story (dramatic actions) or they can be purely part of the gameplay. In-game events are events caused by the core mechanics of the game, it can be AI reacting to the player or the world. Narrative events are events that the player cannot control, although he can in some cases control whether they occur or not. A narrative event narrates some actions to the player, he does not interact with it. (Adams, 2010)

3.1.3 Dramatic tension vs Gameplay tension

Dramatic tension comes from the players desire to know what happens next in the story. Dramatic tension is called conflict by screenwriters and it's the essence of story telling. Cliffhangers, exciting situations, increase the dramatic tension and ensure the player stick around to see how the situation evolves. Gameplay tension is fundamentally different in the sense that it comes from the players own uncertainty about whether or not he can overcome a challenge and he's desire to succeed (Adams, 2010).

Dramatic tension depends on the reader's or players identification with characters and interest in what will happen to them. Gameplay tension does not require any characters. (Adams, 2010) (Young, 2000)

If the events in the story seem random or repetitive, the player starts losing interest in the dramatic subject and dramatic tension fades. (Adams, 2010)

3.1.4 Monomyth – The Heroes Journey

The hero's journey is developed by Cambell and is often used successfully in movies, books and tv-series (Campbell, 2008).

Model has 12 stages that hero must go through to complete the task ahead of him and return the world to harmony.

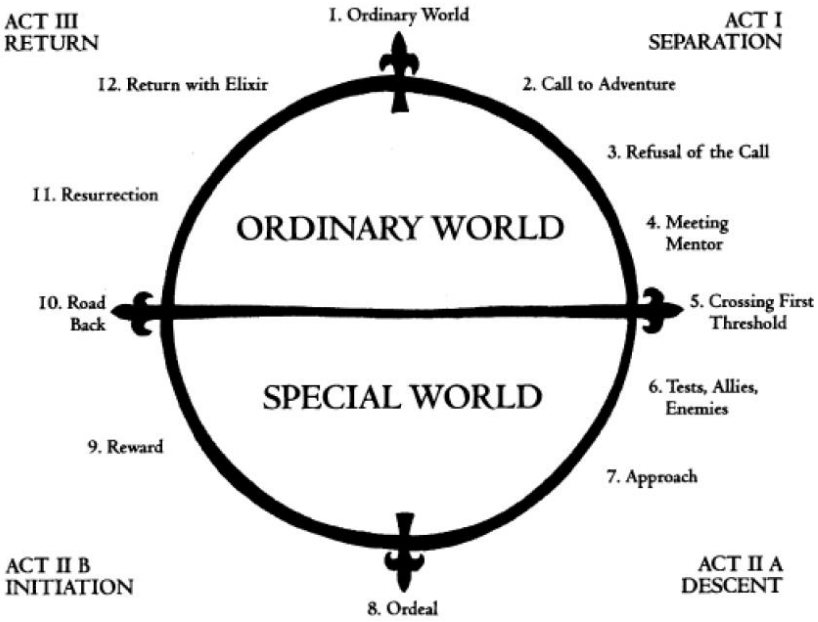


Figure 2: The 12 stages of the hero's journey (Campbell, 2008)

12 stages is divided into two worlds, ordinary world and special world. The 12 stages are described in Table 4 below (Campbell, 2008) (Vogler, 1998).

Stage 1: Ordinary world	It's the baseline which the special world will be compared against. It serves as home and background for the hero.
Stage 2: Call to adventure	The ordinary world is static but unstable. Only a little energy is needed to bring it out of balance. This energy can come in many forms or ways and is often delivered by the archetype 'the herald'.
Stage 3: Refusal of the call	The hero will often refuse the call to adventure, preferring to stay in the ordinary world. Either he's too afraid, is simple unable to do it or something else. This stage is sometimes skipped and the hero answers the call to adventure without refusal.
Stage 4: Meeting mentor	Oftentimes the hero is not ready to enter the special world. He might need physical training, spiritual guidance, a special item or something else completely. The mentor archetype will provide the hero with what he needs.
Stage 5: Crossing the First Threshold	This is sort of a point of no return for the hero. The job of the archetype the 'threshold guardian' is to push the hero into the special world. The threshold guardian could be the villain kidnapping someone close to the hero.
Stage 6: Test, Allies, Enemies	This is the stage where the hero adjusts to the special world. The hero will encounter a number of tests and he might meet a number of people who will try to help (Allies) or stop (Enemies) him.
Stage 7: Approach to Innermost Cave	This is where the hero enters the heart of the special world. There is usually threshold guardian. This is where plans are made, a romance might develop and this is also the time for heroic speeches.

Stage 8: The ordeal	In this stage the hero will experience his death and rebirth. It might not mean literal death but can be an experience that changes him fundamentally. This is also the stage where the main hero either dies or flees.
Stage 9: Reward	This is the stage where the hero gets the reward, it can be something literal like a cure for a disease that has been plaguing the city or something more abstract. This is also a stage where a love scene can play out.
Stage 10: The road back	The hero must decide if he wants to stay in the special world or return to the ordinary world, most heroes chooses to return. On the road back the hero will often encounter a surprise, a final unexpected plot twist. It can be the villain returning or maybe a close ally of the villain seeking revenge.
Stage 11: The Resurrection	It's like a second ordeal and is usually the final showdown. The villain is defeated for good. The hero is often cleansed from the smell of dead but the lessons learned are retained.
Stage 12: Return with the Elixir	The hero will often bring proof of his accomplishments. In many stories the real reward is not treasure but rather the change within the hero. A true hero returns with something that can heal the wounds that made him leave the ordinary world in the first place.

Table 4: The twelve stages of the hero's journey (Campbell, 2008) (Vogler, 1998).

Rollings and Adams (Rollings & Adams, 2003) investigated how the hero's journey is utilized in games. They found that especially the journey back is often not included in the game. Table 5 shows how the hero's journey is often used in games

The ordinary world	Prologue to the game and introduction to the hero and in some cases an introduction to some events from the special world which is about to collide with the ordinary world.
The call to adventure	This is often an event that occurs, to which the hero has some personal investment. Like in Mario Bros where the princess gets kidnap and Mario wants her back. Like in books and movies this varies a lot.
Refusal of the Call	In many games the refusal of the call is left out or only touched on very briefly.
Meeting with the Mentor	This is typically a wise old man but can in fact be any source of information and can be filled by a combination of characters or a library what matters is the hero is presented with the information he needs
Crossing the First Threshold	This can be a simple scene were you see the hero leave the ordinary world, to a boss fight with the main villains right hand man.
Test, Allies, Enemies	This is the main body of the game, were many archetypes can be introduced to the player. However depending on the game you might only meet enemies and no allies at all.
Approach to the Inmost Cave	Preparation to the final ordeal, opportunity to get well equipped mentally, physically or both before the final battle
The Ordeal	This is typically the main boss fight
Reward	Most games end at this stage. In some cases showing the remaining story as a final cut scene. However for a few games this is merely the beginning of the final phase.
The Road Back	Most games do not go as far as this in their stories but leave these story elements to a final cut scene or omit them all together.
The Resurrection	
Return with the Elixir	

Table 5: How the Hero's journey is normally used in video games. (Rollings & Adams, 2003) (Hagen & Thaves-Warnsdorf, 2013).

3.1.5 Archetypes

The hero's journey contains many different archetypes. An archetype is not a character but rather a role that a character can play. It's possible for characters to change roles throughout the story, for instance a character that plays the Ally archetype might temporally take on the Mentor archetype or maybe even the Hero archetype (Vogler, 1998) (Campbell, 2008) (Rollings & Adams, 2003). The most common archetypes are The Hero, The Mentor, The Threshold Guardian, The Herald, The Shapeshifter, The shadow, The Ally and The Trickster (Vogler, 1998).

The Hero is the most known archetype. A hero is not defined by his strength, power or braveness, but rather the sacrifices he has to make. The hero is often the protagonist but it doesn't always has to be. Other characters can take on the hero role temporally. One such example it when the character Ron from the Harry Potter books, sacrifices himself in a deadly game of chess so that Harry can continue his quest.

The Ally is another common archetype. They can serve a variety of functions including love interest, sparing partner, companion and many more. They often challenge the protagonist to be more open and balanced.

The Mentor (sometimes called the Wise Old Man or Wise Old Woman) is another important archetype. The role of the mentor is to protect and guide the protagonist. The Mentor can also provide the hero with gifts to help him complete his quest, either in the form of physical gifts such as a magical sword or more abstract such as good advice.

The Threshold Guardian will test the hero with an obstacle he has to overcome to continue his journey. The Threshold Guardian is often the villain or an ally of the villain, but it doesn't have to be. The Threshold Guardian can also be a friend testing the hero's determination or skills to make sure he is ready for the journey.

The Shapeshifter is an ever-changing character. It's often the opposite sex of the protagonist and might be a love interest as well. The Shapeshifter will constantly change mood, appearance, personality and even loyalty. The Shapeshifter can both be a positive and a negative force either trying to help or destroy the hero.

The Shadow is the villains, enemies and antagonists of the tale. Villains and enemies will directly try to destroy the hero but the antagonist will often try solve the problems the way that fits him best thereby indirectly interfering with the hero's plans. The antagonist might even be an ally that disagrees with the hero and goes behind his back to do what he thinks is best for the hero. Monsters are usually also considered shadows.

The Trickster is usually the clowns and comical sidekicks. They can be either shadows or allies or even an independent party with his own agenda. A trickster will often try solve their problems using wits. Tricksters are often used to bring comical relief to the story.

The Herald is a messenger. He will issue challenges to the hero and announce change. They are used to destabilize the ordinary world the hero lives in. He could be the one announcing to the player that the princess has been kidnapped and that it's up to him to save her.

3.1.6 Actantial model

The actantial model was developed by Algirdas Julien Gremas and is used as a tool to describe a story or a single action in classical fairytales (Hebert, 2006).

The model considers an action as consisting of six parts called actants see Figure 3. An actant can be either a person (such as a prince), an object (such as a magical sword) or even a trait (such as courage).

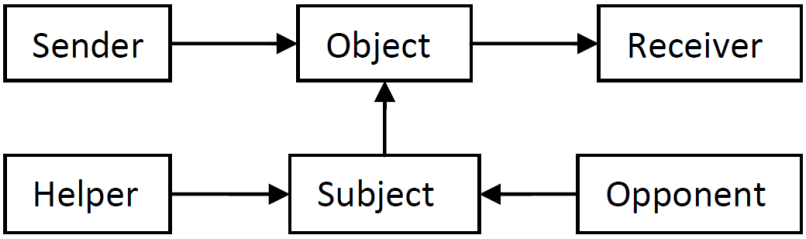


Figure 3: Graphical reprensentation of the actantial model

The subject is the actant the action revolves around, for instance 'the prince'. He desires the object, for instance the princess. The sender is the actant that can grant the subject the object, for instance the king. The receiver is the actant that will benefit from the success of the subject, in classical fairytale this is often the same as the subject. The helper will help the subject to reach his goal, this could be a magical sword and the opponent will try to stop the hero, for instance 'the witch' (Hebert, 2006).

The actantial model is rarely used to describe entire fairytales but works really well on single actions.

3.1.7 Sum up on narratives

A lot of information was gathered in the sections above and a lot of it is highly

Adams said that a game can contain three types of events. Player events, In-game events and Narrative events. The storyengine cannot be responsible for the player events, and the in-game events are events caused by the game such as actions by the AI that controls the enemy. The story engine can only control the narrative events. A narrative event is a block of narrative content. A narrative block can be presented in many ways, a pre-rendered movie, a cut scene using the game engine, voice-overs, a monolog by a character or just text on the screen.

Aarseth divides the events that happens in the game into two other types. He says an event can be a kernel or a satellite. Kernels are important for the story and changing kernels will fundamentally change the story. Satellites are less important for the story and can freely be changed without changing the story. There is no need for the storyengine to generate all the satellites and should focus on the kernels instead.

We also found that there are two types of tensions that can come from a game. Dramatic tension that comes from the players interest in the story and Gameplay tension that comes from the player trying to overcome challenges in the game. The storyengine should only focus on dramatic tension, and shouldn't try to generate situations that result in gameplay tension, that's the games responsibility.

We also looked at the hero's journey and the associated archetypes described by Cambell and Vogler. The hero's journey is nicely structured and could be used as a template that should be filled by the storyengine. We most likely won't need the full twelve stages of the hero's journey. We can use the stages most commonly used in video games as described by Rollings and Adams. The actantial model described by Hebert can be used as a template for more specific conflicts.

Adams said that a story must as a minimum be credible, coherent and dramatically meaningful. I don't fully agree with him, and I don't think that exactly what he meant. A story can indeed be highly uncredible, uncoherent and really boring but still be a story. However a good story must as a minimum be credible,

coherent and dramatically meaningful. These terms might be good in the test of the final product to test the quality of the generated stories.

3.2 Role Playing Games

We know that the game is going to be a roleplaying game as we found in the introduction, to make sure our product does in fact fit the genre, we will in the following section go through the fundamentals of roleplaying games and in the end make a small summery with requirements for our game.

Computer roleplaying games are grown out of tabletop roleplaying games. A key aspect of tabletop roleplaying games is improvisational drama. Most computer roleplaying games however have only borrowed the themes and core mechanics of tabletop roleplaying games and not the roleplaying activity itself. (Adams, 2010)

Adams says that the purpose of roleplaying games is "to experience a series of adventures in an imaginary world, through an avatar or a small group of characters whose skill and powers grow as time goes on" (Adams, 2010)

Some essential parts of computer roleplaying games are quests, story and character growth. When talking about character growth we are not talking about growing as a character in relation to the story, but rather a growth in relation to the gameplay. The character should gain more power and abilities as the game progresses. Some of the typical challenges in roleplaying games are combat, logistics, economic growth, exploration and puzzle solving. Exploring and combat are big parts of most roleplaying games and are needed for the story to progress. In most games the stories and challenges are prescripted, in a few cases, the layout of individual levels are randomized on each play. (Adams, 2010)

Roleplaying games almost always tell a story, characterized as a long quest in pursuit of some important goal. Most roleplaying games revolve around the theme "Only YOU can save the world".

The story is often presented as a journey that will take the player to new locations. It's often possible to go back to previously visited locations even though there might not be anything to do there. In addition to the main quest there are usually also side quests that are unrelated to the main quest. They are usually given to the player by an NPC who has a problem that they hope the hero(s) can solve. The player can usually reject the quest without any penalty. The quests can often be used to progress your character further or earn money or items. (Adams, 2010)

Figure 4 illustrates how the story is usually structured in a roleplaying game.

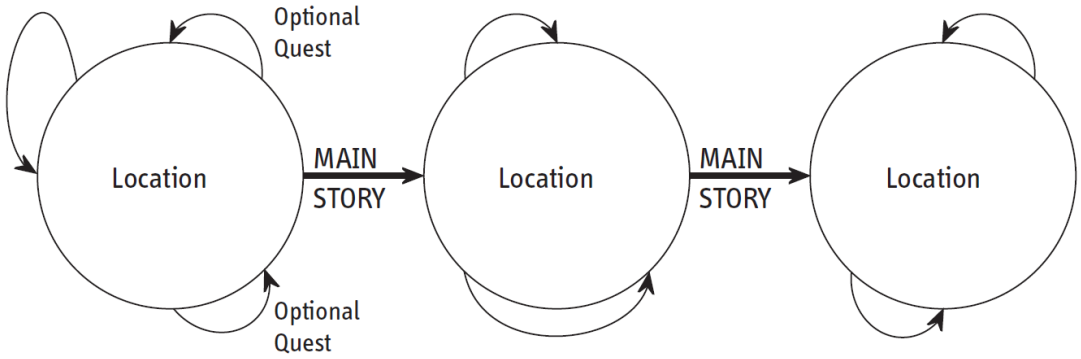


Figure 4: Normal structure for a roleplaying game.

Roleplaying games are usually set in a fantasy or sci-fi universe. These settings allows for a huge variety of enemies, aliens and monsters that don't exists in the real world. They also make it easier for the player to suspend his disbelieve. It's not required that the game takes place in any of those settings though, the only requirement is that the game world invites exploration. (Adams, 2010)

3.2.1 Sum up on roleplaying games

In a roleplaying game the player should be able to control an avatar or a small group of characters. To keep things simple it might be best to settle on a single avatar for this project.

Essential parts of roleplaying games include quests, story and character progression, exploration and combat. Side quests are outside the scope of this project, but the story should be in the shape of a large quest. It's oftentimes a journey that will take the player to new locations. This fits well with story graphs as described in the narrative chapter above, where you follow a predefined path between episodes.

Roleplaying games are often set in a fantasy or sci-fi universe, but doesn't have to be.

3.3 Procedural techniques

In the following chapter we will investigate some different procedural techniques that can be used when developing the story engine.

3.3.1 Generative Grammar

Generative grammars are used to describe the grammatical syntax of languages. Below in Figure 5 is an example of generative grammar in use.

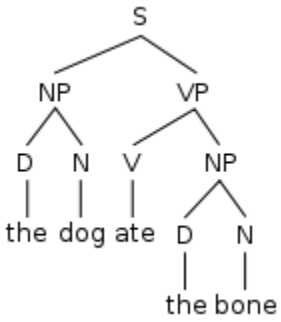


Figure 5: Generative grammar of a sentence

The rules for this system would be:

- S (sentence) → NP (noun phrase) + VP (verb phrase)
- NP → D (determiner) + N (noun)
- VP → V (verb) + NP
- D → the
- N → bone
- N → dog
- V → ate

Generative grammars with an alphabet consisting of game specific concepts can be used to describe different elements of a game. Dormans wrote a paper on how he successfully used generative grammar to generate levels for a game (Dormans, 2010). A grammar that describes possible levels of an adventure

game could have symbols such as: 'key', 'lock', 'room', 'monster', 'treasure' and the rules might look like this:

1. Dungeon → Obstacle + treasure
2. Obstacle → key + Obstacle + lock + Obstacle
3. Obstacle → monster + Obstacle
4. Obstacle → room

To generate a level you would start with the dungeon and replace it with what is on the right side (Obstacle + treasure). You then recursively replace Obstacle with 2, 3 or 4. These rules can generate a large variety of strings such as: key + monster + room + lock + monster + room + treasure.

Dormans points out that generative grammar can be used in different ways to produce content for games.

3.3.2 Name Generation

In our 8th semester project we successfully used two techniques to generate city names and character names (Hagen & Thaves-Warnsdorf, 2013). These techniques can be reused in this project.

3.3.2.1 Character names

The character name generation algorithm is based on the work of internet user Joshua Smyth (Hagen & Thaves-Warnsdorf, 2013). The algorithm generates names with a length of either 5 or 6 letters. Names follow the form VCC-VCV or CVC-CVC where each V is replaced with a random vowel and each C is replaced with a random consonant. With a 50% change the last letter is removed from the name to create a 5 letter name. The algorithm can produce names such as Yivha, Xifei, Pojlo, Xagloh and Ilhalo.

During our 8th semester project we found that the quality of the generated names improved if we did not use the letters x, y and z as they are somewhat uncommon in names. This will produce names such as Ilhalo, Pojlo, Digja and Emrapo.

Another solution would be to apply weight to letters depending on how common they are in names so that x, y and z are rare but not completely excluded.

3.3.2.2 City names

We found that city names can often be divided into a prefix and suffix. One such example is 'London' consisting of the prefix 'Lon' and suffix 'don'. The algorithm we ended up developing takes a list of common prefixes and a list of common suffixes for British cities as input, it then pairs a random prefix with a random suffix to generate a city name. This algorithm can produce names such as Inverport, Finwick, Balladen, Nanting, Dunham and Mynyddpool.

4 DESIGN

The following chapter describes the technical design of the storyengine and will also describe the graphical design of the game itself and how the gameplay is designed.

4.1 The storyengine

The storyengine generates the stories in steps. The first step is to figure out what events lead up to the beginning of the story. The next step is to find out who the opponent is, and what he's motivation is. The third step is the find what actions/events should happen in the game. The forth step is to generate information about the rest of the characters in the game such as the mentor, the herald and the hero. The fifth step is to generate the conversations, and the last step is to convert all the information into a format that's readable for the game.

4.1.1 Step 1: Generating the events leading up to the beginning of the story

I had to come up with some kind of method to generate a reason for the hero to leave the ordinary world. The solution is to generate a chain reaction of events that would eventually lead to an event that would cause the hero to enter the special world. I came up with a method for doing that, I call it "casual chain generation".

The algorithm takes two lists as input. A list of actions and a list of effects. In our case the list of effects are events that would force the hero to leave the ordinary world, examples include 'Hometown is in danger' or 'Son is deadly sick'. The list of actions consists of events that would not directly affect the hero, such as 'War brakes out' or 'The king dies'. Each action also has a list of possible consequences. Consequences can either be effects or other actions, for instance the action 'war breaks out' can have the consequences 'The king dies' (action) and 'Hometown is in danger' (effect).

To generate the chain, the algorithm starts by selecting a random action from the action list, it then picks a random consequence associated with that action. If the consequence is a new action the program once again pick a random consequence associated with the new action. It will keep doing so until the consequence is an effect, at that point the algorithm is done.

Consider the following actions:

Actions	
Name	Consequences
War breaks out	• The king dies (action) • Civil war breaks out (action) • Rebels start plundering (action) • Hometown in danger (effect) • A loved one is killed (effect)
The King dies	• Civil war breaks out (action) • War breaks out (action) • Throne successor is kidnapped (action)
Civil war breaks out	• Rebels start plundering (action) • Hometown in danger (effect) • A loved one is kidnapped (effect) • A loved one is killed (effect)

Rebels start plundering	• Civil war breaks out (action) • Hometown in danger (effect) • A loved one is kidnapped (effect) • A loved one is killed (effect)
Throne successor is kidnapped	• Civil war breaks out (action) • War breaks out (action)

With the actions above the algorithm would first pick a random action, for instance "The king dies". It will then pick a random consequence for that action, for instance "Throne successor is kidnapped". This is a new action so we go to that action and pick one of the consequences associated with that action, for example "Civil war breaks out". This is also an action, so once again we pick a random consequence for that action, for instance "A loved one is kidnapped". The last one is an effect so the algorithm ends here and returns the generated list, it would look like this:

The king dies → Throne successor is kidnapped → Civil war breaks out → A loved one is kidnapped

This will allow for easy generation of background story while providing the hero with the motivation needed to leave the ordinary world. The system should be implemented so that it's easy to expand with more actions and effects. If needed, the system can be expanded to ensure that some actions can only be used once. For instance it would be incoherent if the king dies multiple times.

Even though the method in this case is used to generate a reason for the hero to leave it doesn't have to. It can be used in any situation where somebody might want to generate a chain reaction leading to some final event/effect.

4.1.2 Step 2: Generate the Opponent and find his motivation

When finding the opponent in the story it is important that he fits into the events that was generated in step 1. To find the opponent we go back in the chain of events until we find the first action caused by a person and that person will be the opponent. Take the example from before:

The king dies (action) → Throne successor is kidnapped (action) → Civil war breaks out (action) → A loved one is kidnapped (effect)

We go back in the chain until we find the first action caused by a person, in this case we only need to go back one step to "Civil war breaks out". Who is responsible for that? That would be the 'Leader of the revolution'. So we know that the Leader or the revolution is the opponent but he also needs a motivation, so with each action is a list of motivations, in the case of "Civil war breaks out" possible motivations could be "Overthrow the king" or "Gain independence". A random motivation will be chosen.

This works system works surprisingly well. Consider this other example:

Civil war breaks out (action) → Rebels start plundering (action) → Hometown is in danger (effect)

We go back to the first action that is caused by a person, in this case "Rebels start plundering". This action is caused by the Rebel Leader, and he can have one of the following motivations; "Get rich" or perhaps "Take control of the city". If we put it together we see that a civil war breaks out, because of this rebels start plundering and this puts the hometown in danger, the opponent is the Rebel leader and he wants to get rich.

4.1.3 Step 3: Find what actions/events should happen in the game

At this point we know the chain of events leading up to the beginning of the story, we know who the opponent is and we know his motivation. We know need to find out how the story is going to unfold to the player. To do this we will use generative grammar as described in section 3.3. We want the story to follow the hero's journey, so the first rule in our grammar is:

Story →	Ordinary world + Call to adventure + Meeting mentor + Crossing first threshold + Test, Allies, Enemies + Approach to innermost cave + The ordeal + Reward

The rule "Story" should be replaced by eight new rules each corresponding to a stage in the hero's journey. Some stages from the hero's journey is not included as it was found in the analysis, that some of the stages are rather unusual in video games.

Each of these then have new rules. For instance the 'Crossing the first threshold' could look like this:

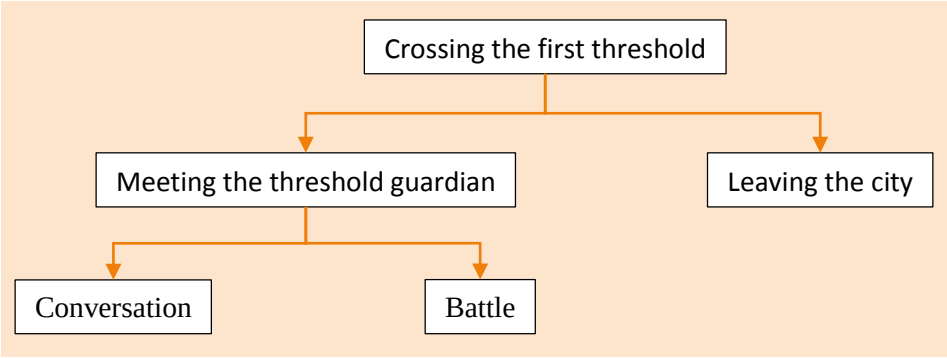


Figure 6: Example rule for generative grammar for the story structure

The full list of the rules and symbols used to generate the story structure can be found in appendix 12.4.

4.1.4 Step 4: Generate characters

There are 4 characters we know will always be in the story; The hero, The Herald, The Mentor and The Opponent. For each character we will generate a name using the method described in section 3.3. We then go through the story structure generated in step 3 and look up all the allies that will be joining the hero in the 'Test, Allies, Enemies' stage. Each ally will be given a unique ID and a name.

4.1.5 Step 5: Generate conversations

To generate conversations I decided to design a method that can take a list of lines that needs to be said. Each line can either be marked as good news, bad news or neutral, and each line will also have marked who should say it. When generating a conversation the method goes through each line and randomly responds to each line depending on how it's marked, for instance if it's marked as good news the other person might respond with something like "That's great!". If it's marked as neutral no response will be inserted. For instance we might input the following lines:

- Him: "I just got a dog" – Good news
- You: "What breed is it?" – Neutral
- Him: "It's a fighting breed" – Bad news

The method would go to the first line "I just got a dog" and see that it's marked as good news so it might insert the response "I'm happy to hear that". It then goes to the next line "What breed is it?" and see that it's marked as neutral, so no response is inserted. We then look at the last line "It's a fighting breed" and

see that it's marked as bad news, so we might insert something like "That can't be good". The final conversation will look like this:

- Him: "I just got a dog"
- You: "I'm happy to hear that"
- You: "What breed is it?"
- Him: "It's a fighting breed"
- You: "That can't be good"

Oftentimes a conversation starts with an introduction/greeting. These introductions often follow a fixed template. It was decided to use generative grammar to generate the introductions in the conversations. Below is an example of a conversation introduction, started by the other person where both know each other beforehand.

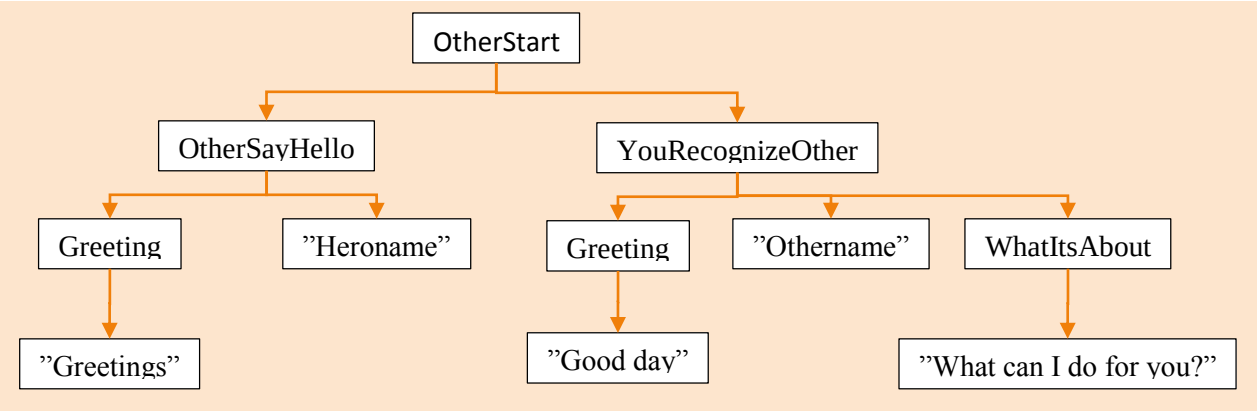


Figure 7: Generative grammar used to generate an introduction to a conversation where the participants know each other

This would result in a greeting like this:

- Him: "Greetings heroname"
- You: "Good day othername. What can I do for you?"

And from then on, the rest of the conversation could continue as described above with the dog.

So the full conversation could be:

- Him: "Greetings heroname"
- You: "Good day othername. What can I do for you?"
- Him: "I just got a dog"
- You: "I'm happy to hear that"
- You: "What breed is it?"
- Him: "It's a fighting breed"
- You: "That can't be good"

An conversation intro started by the hero where the parties don't know each other might go something like this:

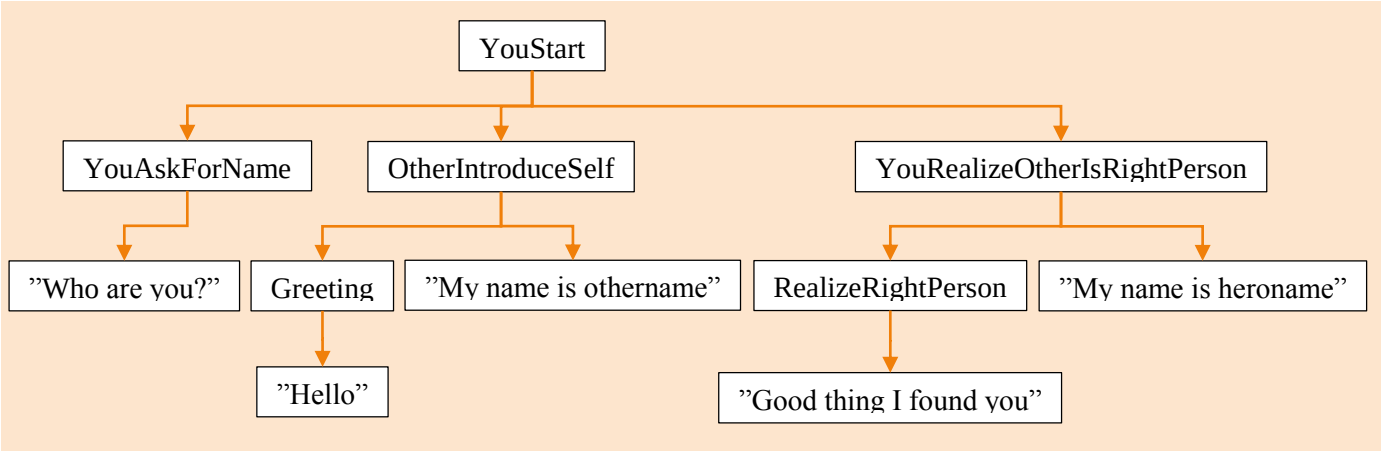


Figure 8: Generative grammar used to generate an introduction to a conversation where the participants don't know each other

This would result in a conversation introduction like this:

- You: "Who are you?"
- Him: "Hello, my name is othername"
- You: "Good thing I found you. My name is heroname"

4.1.6 Step 6: Convert to a readable format for the game

It was decided that the story output from the storyengine should be saved in an xml file. The game can then read the xml file and implement the story from that.

4.2 The game

Below we will go through some of the graphical design and gameplay design in the game itself.

4.2.1 Graphics

The graphics used in the game is not original content made specifically for this project. To save time on the production of the game it was decided to download existing assets from the internet. Two main asset packs was used in this game. One pack contains characters and animations while the other pack contains static scenery. The two packs are not a 100% compatible in style, but where chosen for their large collection of assets.

Below is a promotional image of the character pack used in the game.

Figure 9: Promotional image of the asset pack "Chibi realm" found on the unity asset store

Figure 10 is a promotional image of the other main pack, showing a selection of the assets in the pack.

Figure 10: Promotional image of the asset pack "Make Your Fantasy Game" found on the unity asset store

Figure 11 below is a screenshot from the game. It shows what the two packs look like together in the game.

Figure 11: Screenshot from the game showing the two character packs working together

4.2.2 Gameplay

The game is designed as a classic hack n' slash roleplaying game. The game has some of the most common roleplaying game mechanics such as an inventory system where the player will be able to equip different items that will change the stats of the player such as the attack damage or the defence, and a shop where the player can buy and sell items. The player controls the character from a 3rd person perspective, this will make it easy for the player to see what items are equipped.

If a test participant is not an experienced gamer it's possible that he will not be able to finish the game. It's a bias in the test if the player is unable to finish the game and see the entire story including the end. To avoid this it was decided to design the game so that the player cannot die. The player will take damage and the health bar will go down giving the illusion that he can die, but it will never actually reach zero.

5 IMPLEMENTATION

Below I will go through some of the more interesting parts of the storyengine and the game itself.

5.1 The storyengine

The story engine is developed in C# using the .NET framework

5.1.1 Causal chain generation

All actions and effects are stored in an xml document. This will make it easy to add new actions and effects without changing the source code.

An action follows the format:

```
<action id=”X”>
   <description>Description of action</description>
   <consequence type=”Y” id=”Z” />
</action>
```

Code 1: Example of what an action looks like in xml format

Where X is a unique id for that action, Y is either ”action” or ”effect” depending on the type of the consequence and “Z” is the unique id for that action or effect. An action can contain multiple consequences.

An effect follows the format:

```
<effect id=”X”>
   <description>Description of effect</description>
</effect>
```

Code 2: Example of what an effect looks like in xml format

Where X is a unique id for that effect.

5.1.2 Generative grammar

The generative grammar consists of two parts. A list of rules and a recursive function that executes the generative grammar.

A rule is a struct that look like this:

```
struct Rule
{
   public string name;
   public string[] replace;

   public Rule(string _name, string[] _replace)
   {
      name = _name;
      replace = _replace;
   }
}
```

Code 3: The struct for a rule to be used in generative grammar

The name is the name of the rule and the array called replace is what the rules should be replaced with. For instance the rule; Dungeon → Obstacle + treasure will look like this:

```
Rule myRule = new Rule("Dungeon", new string[]{"Obstacle", "treasure"});
```

The recursive function that executes the rules look like this:

```
string ExecuteRule(string currentRuleName, List<Rule> allRules)
{
   //currentRuleName is the name of the rule to be executed
   //allRules is a list of all the rules

   //First we find all the rules in 'allRules' called 'currentRuleName'
   //and save them in a list called 'currentRules'
   List<Rule> currentRules = new List<Rule>();
   foreach (Rule r in allRules)
      if (r.name == currentRuleName)
         currentRules.Add(r);

   //If we could not find any rules named 'currentRuleName' it must be a
   //symbol so return it
   if (currentRules.Count == 0)
   {
      return currentRuleName;
   }
   else
   {
      //We found one or more rules name 'currentRuleName' so we pick one
      //of them
      Rule nextRule = currentRules[MyRandom.Next(currentRules.Count)];

      //We then execute all the rules in the replace array for next rule
      string temp = "";
      foreach (string s in newRule.replace)
      {
         temp += ExecuteRule(s, allRules);
      }
      return temp;
   }
}
```

Code 4: Recursive function that executes the rules in generative grammar

5.1.3 Seeds

The story engine takes a seed as input and uses that as a seed for the random class built into .NET to make sure the generated stories will always be the same when provided with the same input seed. The seed needs to be an integer, but I wanted to allow the user to provide a string as the seed. To allow for this I use .NETs "GetHashCode()" method on the entered string to get a numeric value. The method generates a hash code of the datatype integer, based on the string, and is for instance used to be able to compare if two strings are equal in content, but the hash code can also be used as a seed in the random algorithm since the same string will always return the same hash code.

5.2 The Game

The game is developed in C# using the Unity3D game engine

5.2.1 Conversations

Since the conversations in the game will be non interactive, we decided to present them in the narrative form of a cutscene.

When the player talks to npcs in the game it was decided to move the camera down in between the characters and a little to the side, and then let the camera point at who is talking. This makes the conversation cutscenes feel more cinematic. Because the camera is only changing the direction it's pointing and doesn't actually move we never break the 180 degree rule of film making that says the camera should never cross the axis of action going from the first character to the second.

Figure 12 is a sketch showing how the camera is placed during conversations. The first character is located at $\overrightarrow{v_1} = (x_1, y_1, z_1)$, the second character is located at $\overrightarrow{v_2} = (x_2, y_2, z_2)$ and the camera is located at $\overrightarrow{v_3} = (x_3, y_3, z_3)$. It's worth noting that y denotes up in the Unity3D game engine.

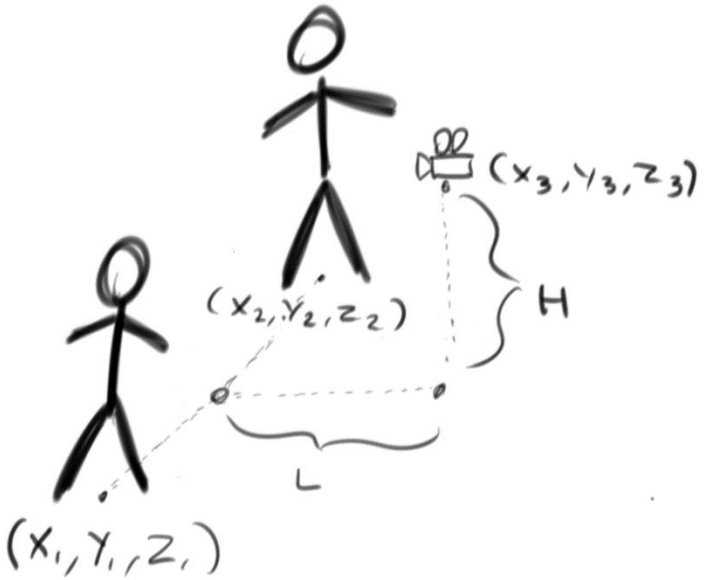


Figure 12: Illustration of the position of the camera in the game world during a conversation between two characters.

To find the position of the camera we first find the center between $\overrightarrow{v_1}$ and $\overrightarrow{v_2}$, this is done by talking the average:

$$center = \frac{(\overrightarrow{v_1} + \overrightarrow{v_2})}{2}$$

We then need to move the camera to the side, perpendicular to the vector going from $\overrightarrow{v_1}$ to $\overrightarrow{v_2}$ along the x,z plane. We do this by taking the cross product between the vector going from $\overrightarrow{v_1}$ to $\overrightarrow{v_2}$, and the vector $\overrightarrow{up} = (0, 1, 0)$

$$perpendicular = (\overrightarrow{v_2} - \overrightarrow{v_1}) \times \overrightarrow{up}$$

To control the distance from the center we can normalize the vector p and scale it by L, and to control the height of the camera we can add $\overrightarrow{up}$ scaled by H.

The full equation to find the position of the camera can be seen below.

$$\overrightarrow{v_3} = \frac{(\overrightarrow{v_1} + \overrightarrow{v_2})}{2} + \frac{(\overrightarrow{v_2} - \overrightarrow{v_1}) \times \overrightarrow{up}}{|(\overrightarrow{v_2} - \overrightarrow{v_1}) \times \overrightarrow{up}|} \cdot L + \overrightarrow{up} \cdot H$$

Where $\overrightarrow{v_1}$is the position of the first character, $\overrightarrow{v_2}$ is the position of the second character, $\overrightarrow{v_3}$ is the position of the camera, $\overrightarrow{up}$ is the up vector, and L and H are constants.

In code, finding the position of the camera will look something like this:

```
Vector3 v1 = new Vector3(x1, y1, z1); //Position of first character
Vector3 v2 = new Vector3(x2, y2, z2); //Position of second character
Vector3 up = new Vector3(0, 1f, 0);

float L = 2f; //Distance from the center
float H = 2f; //Height above ground

//First find center between v1 and v2
Vector3 cameraPosition = (v2 - v1) / 2f;
//Move along the perpendicular vector
cameraPosition += Vector3.Cross(v2 - v1, up).normalized * L;
//Move camera up above the ground
cameraPosition += new Vector3(0, H, 0);
```

Code 5: Code to correctly place the camera between two character talking to eachother

Figure 13 below shows what the conversations look like in game.

Figure 13: A screenshot from the game showing what the conversations look like in game.

6 TEST

Since we are testing for replayability, the optimal way to test the storyengine would be to implement a game that can take the output from the storyengine and generate the game from that, and then ask the test participants to play the game as many times as they want. One group would have a new story each time they play, and a control group would play the same story each time. However, since there wasn't time to implement such a solution, we have to come up with an alternative testing procedure. As described in section 2, a single story was generated using the storyengine and then implemented into the game. The story used can be found in appendix 12.5.

6.1 Test procedure

Before the test begins the test participant will receive a short introduction where it will be explained what is going to happen and a short introduction to the game including the controls. The full introduction can be found in appendix 12.1. After the introduction, the test participant will be asked to play the game from start to end. When the test participant has completed the game he will be asked to fill out a questionnaire. The full questionnaire can be found in appendix 12.2. A test observer will be present during the test in case the player has any questions during the test, and the controls explained during the introduction will be printed on a piece of paper and lie next to the computer during the test in case the player forgets the controls during the test, see appendix 12.3. The test participant will be free to leave at any time.

6.2 Questionnaire

The questionnaire will be in the form of an online questionnaire. The full questionnaire can be found in appendix 12.2. The first part of the questionnaire will consists of basic demographic questions and questions relating to their experience with gaming. These data will most likely not be very relevant to the project, but in case of unusual test results it will be beneficial to be able to confirm that it's not caused by an unusual test group.

In the analysis we found that a story must as a minimum be credible, coherent and dramatically meaningful. The second part of the questionnaire will be used to verify that the story fulfills these minimum requirements. The test participants will be asked to what degree they agree with the three statements that they found the story to be credible, coherent and dramatically meaningful.

The third part will investigate the replayability of the game. The test participants will be asked how much they agree with the statement; "I want to play the game again". After that, it will be explained that the story was procedurally generated and they will again be asked how much they agree with the statement; "I want to play the game again", to see if the fact that the story is procedurally generated affects their desire to play again.

The last part of the questionnaire will consist of qualitative questions. The test participant will be asked if he answered the two statements from the third part differently and why/why not. The test participant will then be asked if he has any general comments on the story. The test participant will also be asked about general comments on the game such as gameplay, controls, etc. The answers to this question will most likely not be of any value to the project, but previous experience has shown us that if this question is not provided, some test participants will just put general feedback under one of the other qualitative questions. Lastly, to ensure that the test was not affected by any bugs the test participant will be asked if he experienced any bugs, problems or similar during the test

7 RESULTS

The test was conducted on the 21st and 22nd of May 2014 on Aalborg university in Copenhagen. A total of 29 test participants, 24 male 5 female, between the age of 19 and 42 took part in the test. The raw test results can be found on the CD in excel and pdf format. Below we will go through the results. It's worth noticing that in the raw test results there are 30 responses, however two responses was submitted 7 seconds apart containing the same answers. We assume that the same response has been submitted twice and have therefore removed one of them from the results below.

When asked how often they play video games, 11 participants answered daily (one or more times each day), 7 answered weekly (one or more times each week), 3 answered monthly (one or more times each month), 4 answered less then monthly and 4 answered never as seen in Table 6.

How often do you play video games?				
Daily	Weekly	Monthly	Less then monthly	Never
11	8	3	4	4

Table 6: Answers to the question "How often do you play video games?"

For the question "Do you have experience with using a joypad to control games?", 12 rated them self as very experienced, 7 as somewhat experienced, 5 as minimally experienced and 5 said that they never or very rarely use a joypad, see Table 7.

Do you have experience with using a joypad to control games?			
Very experienced	Somewhat experienced	Minimal experience	Never/Very-rarely used a joypad
12	7	5	5

Table 7: Answers to the question "Do you have experience with using a joypad to control games?"

When asked to rate to what degree they agree with the statement "I found the story in the game to be credible", 3 strongly agreed, 15 agreed, 8 neither agreed nor disagreed, 3 disagreed and no one strongly disagreed as seed in Figure 14.

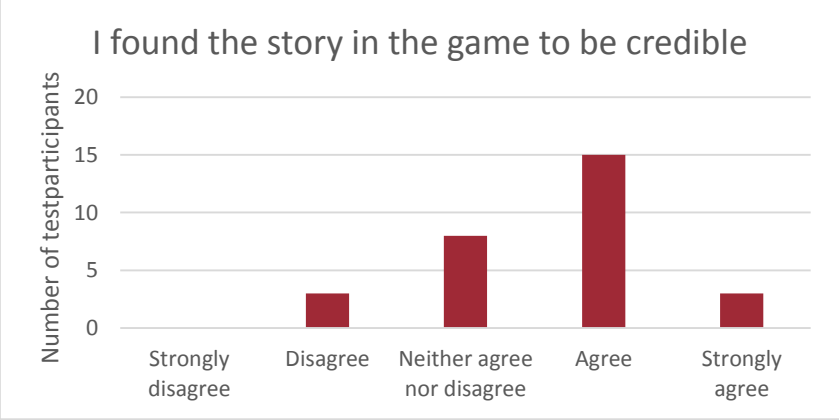


Figure 14: Answers to the question "I found the story in the game to be credible". On the x axis is possible answers and on the y axis is the amount of test participants that answered that.

When asked the statement "I found the story in the game to be coherent", 3 strongly agreed, 20 agreed, 4 neither agreed nor disagreed, 2 disagreed and no one disagreed as seen in Figure 15.

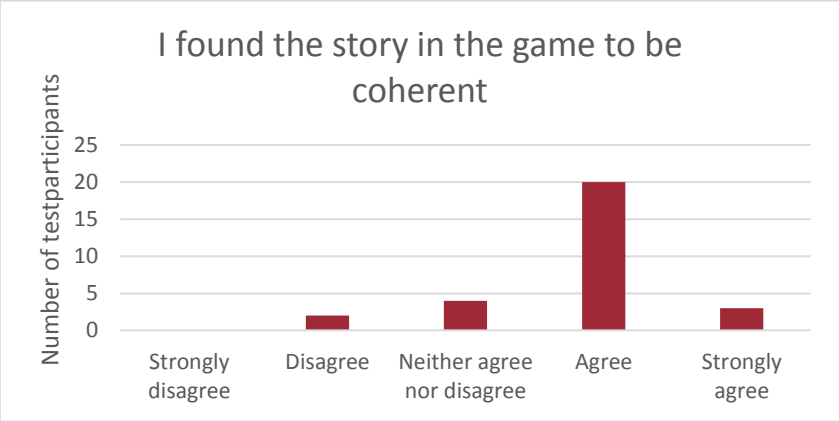


Figure 15: Answers to the question "I found the story in the game to be coherent". On the x axis is possible answers, on the y axis is amount of test participants that answered that.

For the statement "I found the story in the game to be dramatically meaningful", 3 strongly agreed, 9 agreed, 5 neither agreed nor disagreed, 8 disagreed and 4 strongly disagreed as seen in Figure 16.

Figure 16: Answers to the question "I found the story in the game to be dramatically meaningful". On the x axis is possible answers, on the y axis is amount of test participants that answered that.

To calculate the mean and standard deviation for each of the three questions above, each of the five answers was converted into a numeric value from 1 to 5, with one being "strongly disagree" and 5 being "strongly agree". The calculated means and standard deviation can be found in Table 8 and Figure 17.

	I found the story in the game to be credible	I found the story in the game to be coherent	I found the story in the game to be dramatically meaningful
Mean	3.6	3.8	2.97
Standard deviation	0.8	0.7	1.25

Table 8: The calculated means and standard deviation for each of the three statements about the story quality.

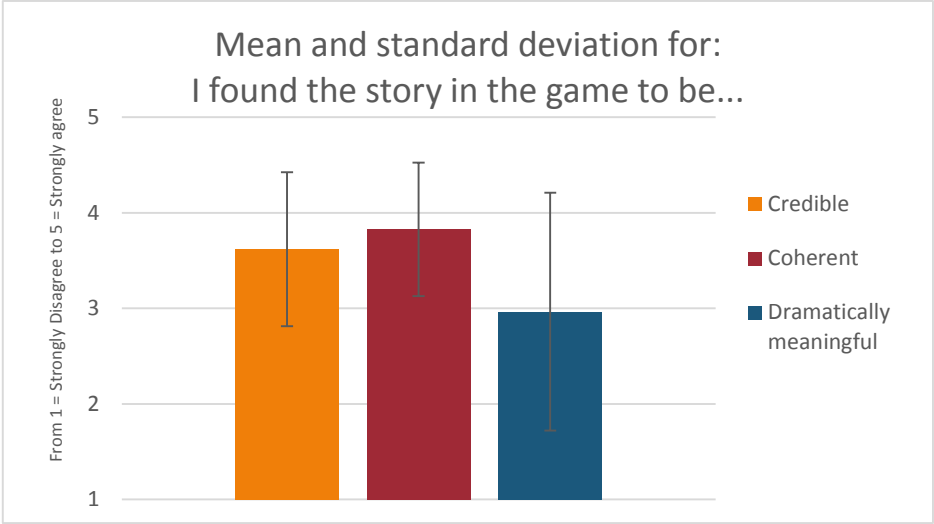


Figure 17: Graph illustrating the calculated mean and standard deviation for the three statements about story quality.

When asked to rate to what degree they agreed with the statement "I want to play the game again", 1 strongly agreed, 11 agreed, 11 neither agreed nor disagreed, 5 disagreed and 1 strongly disagreed. After it was revealed to them that the story was procedurally generated and they were asked the same statement again, 9 strongly agreed, 14 agreed, 5 neither agreed, 1 disagreed and no one strongly disagreed. This can also be seen in Figure 18.

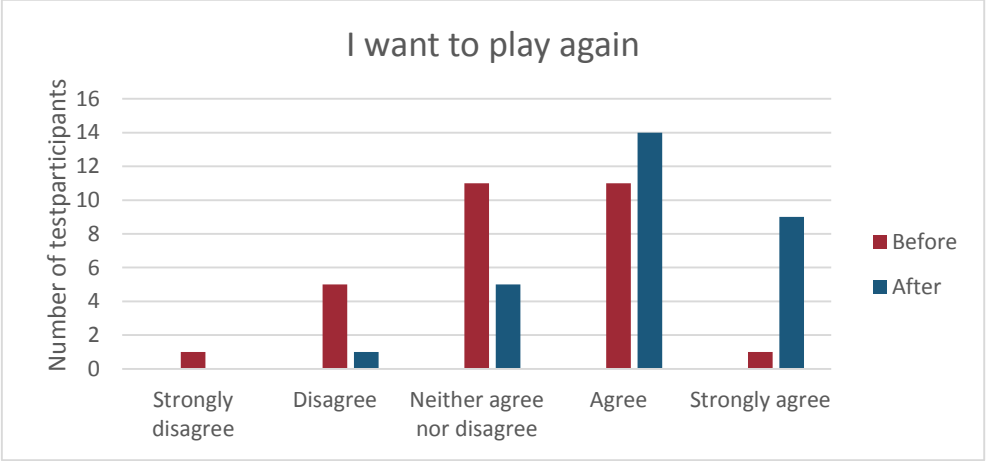


Figure 18: Graph showing how much the test participants agree with the statement "I want to play again" before and after is was revealed that the story is procedurally generated.

To help us see if there is a significant difference in the answers before and after it was revealed that the story is procedurally generated, each of the possible answers where given a value from 1 to 5, with 1 being strongly disagree and 5 being strongly agree. With these numbers the mean and standard deviation was calculated as seen in Table 9 and Figure 19.

"I want to play again"	Before being told the story is procedurally generated	After being told the story is procedurally generated
Mean	3.2	0.89
Standard deviation	4.1	0.78

Table 9: The calculated means and standard deviations for the statement "I want to play again" before and after it was revealed that the story is procedurally generated

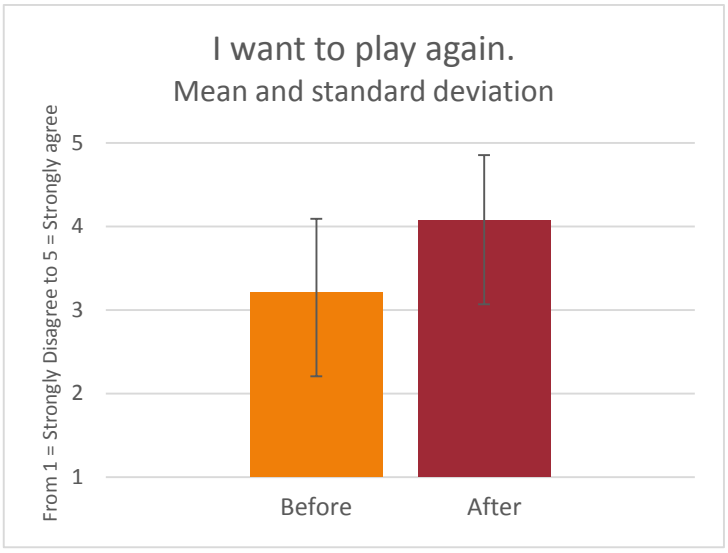


Figure 19: Graph showing the calculated means and standard deviations for the statement "I want to play again" before and after is was revealed that the story is procedurally generated.

8 DISCUSSION

First of all we have to discuss if I even succeeded in creating stories that fulfill the minimum requirements that it should be credible, coherent and dramatically meaningful. When asked if they thought the story was credible 18 out of 29 agreed or strongly agreed and the calculated mean lies only slightly closer to "agree" then "neither agree nor disagree" and with the standard deviation stretching all the way down below "neither agree nor disagree". So we cannot conclude that the story was credible although there is a tendency towards it. When we look at coherence we scored slightly higher, only 6 out of 29 did not agree that the story was coherent, and the calculated mean lies just below "agree". This suggests that the story was coherent. The same thing cannot be said about the story being dramatically meaningful. Only 12 out of 29 agreed or strongly agreed that the story was dramatically meaningful, and the calculated mean is almost exactly on "neither agree nor disagree" with a big standard deviation stretching from below "disagree" to above "agree". This means that the story was not dramatically meaningful, and indeed some of the comments from the test participants confirm this. One person said *"it's seems like the story is made up a bit too fast and there could have been more afford put into it. seems like the script and what the characters a saying is rushed through a bit too fast - they don't have much to say to each other."*

Other test participants pointed out that although the story was fine, it was too short to really get to know any of the characters. One participant said: *"The details were few - maybe too few for me to really get involved. […]"*, another said: *"There wasn't enough time or events to get to know the characters well. One only got a hint of their personalities, but not much else."*, and a third said: *"The Story was fitting for the length of the game but because the game was short, the story was limited and lacked depth."*

When looking at the calculated means for the question "I want to play again" before and after it was revealed that the story was procedurally generated, we see a rise from "neither agree nor disagree" to "agree". Furthermore when looking at each individual test participant no one had a decreased desire to play again, after they knew the story was procedurally generated compared to before they knew. This suggests that a procedurally generated story has a positive effect on the replayability of a game. This is further supported by some of the comments from the test participants. One test participant said *"[…] I would not feel very inclined to play a game with the same story twice. I would, if it was different."*, and another said *"Think that if the game's story is random every time then it makes the replayability of the game very exciting indeed and worthwhile playing more than once."*.

For some people, the fact that the story is procedurally generated did not affect their desire to play again. For one test participant the game was good enough that he wanted to play it again even with the same story: *"[…] it was a fun game [so I] would still play it before i knew it changed"*, another test participant did not like the game due to it not being challenging enough and a new story wouldn't change it. He disagreed with the statement "I want to play again" both before and after he knew the story was procedurally generated and gave the explanation for his answers: *"because the game was not challenging enough, and I wasn't significantly amused while playing it"*.

It should be noted that the test group is slightly skewed as the large majority is men. It should also be noted that although 4 test participants answered that they never play video games and 5 answered that they never/very-rarely use a gamepad, the test observer didn't notice any of the players having any major problems controlling the game.

9 CONCLUSION

When we cannot confirm that the story is credible, and when we know the story was not dramatically meaningful we can conclude that the procedurally generated stories generated with this method does not fulfill our minimum requirements found in the analysis (see section 3.1.7). The generated stories needed to be longer to allow for people to get into it and deeper to allow for the players to connect more with the characters.

Because of the test setup it's hard to conclude if the procedurally generated narratives improves replayability. Because the players only got to play one story they can't get a real feel of how much or how little the stories will change between each game and so it becomes more of a test to see if people like the idea of procedurally generated narratives rather than this specific implementation. Even then, it's not possible to conclude on anything, but the test results do seem to suggest that the replayability would improve with procedurally generated narratives for most people.

10FUTURE PERSPECTIVES

A lot of things can be improved on this project. Most obviously the game could be improved to be able to take the output from the storyengine and use that in the game instead of the static story it has now. Taking things one step further, the storyengine could be incorporated into the game creating one whole product rather than two separate.

The storyengine itself should also be improved to create longer and more complex stories. One component that could be added to the story engine is to generate background stories for each character in the game that can slowly be revealed though the story. At the moment the only character with something close to a background story is the opponent who has a "motivation" for doing what he does, but no real background story.

When developing the storyengine I tried to make it completely independent from the game, but failed to make the game independent from the story. For instance the story dictates when the player should enter or leave, cities or caves. The storyengine would be a lot more modular and easy to implement into other projects if it's completely "level-layout"-agnostic. For instance, the storyengine should decide who the player should talk to, what order he should talk to them in, and what information he should get from them but the storyengine should not decide where the player should talk to them.

11 REFERENCES

Adams, E. (2010). *Fundamentals of Game Design* (2 ed.). Pearson Education.

Boyang, L., & Riedl, M. O. (2010). An Offline Planning Approach to Game Plotline Adaptation. *AIIDE*.

Campbell, J. (2008). *The hero with a thousand faces* (17 ed.). New World Library.

Charles, F., Mead, S. J., & Cavazza, M. (2001). Character-driven story generation in interactive storytelling. *Proceedings of the Seventh International Conference on Virtual Systems and Multimedia*, p. 609.

Dormans, J. (2010). Adventures in level design: generating missions and spaces for action adventure games. *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*, p. 1.

Figueiredo, R., Brisson, A., Aylett, R., & Paiva, A. (2008). Emergent stories facilitated. *Interactive Storytelling*, pp. 218-229.

Hagen, L., & Thaves-Warnsdorf, F. (2013). A method for procedural randomly generated narratives in video games. *Aalborg University Copenhagen, 2nd semester medialogy master project*.

Hartsook, K., Zook, A., Das, S., & Riedl, M. O. (2011, August). Toward Supporting Stories with Procedurally Generated Game Worlds. *Computational Intelligence and Games*, pp. 297-304.

Hebert, L. (2006). *Tools for text and image analysis: An introduction to applied semiotics.* Paris: ADAGP.

Ip, B. (2011). Narrative Structures in Computer and Video Games: Part 1: Context, Definitions, and Initial Findings. *Games and Culture*, pp. 103-134.

Mateas, M., & Stern, A. (2003). Façade: An experiment in building a fully-realized interactive drama. *Game Developers Conference*, pp. 4-8.

Mateas, M., & Stern, A. (2005, June). Structuring content in the Façade interactive drama architure. *AIIDE*, pp. 93-98.

Rollings, A., & Adams, E. (2003). *Andrew Rollings and Ernest Adams on game design.* New Riders.

Sennott, M. J. (2012). Quicksilver: Infinite Story Procedurrally generated episodic narratives for gameplay. *Diss. University of Southern California*.

Sgouros, N. M., Papakonstantinou, G., & Tsanakas, P. (1996). A framework for plot control in interactive story systems. *AAAI/IAAI*, pp. 162-167.

Togelius, J., Yannakakis, G. N., Stanley, K. O., & Browne, C. (2012). Search-based Procedural Content Generation. *Applications of Evolutionary Computation*, pp. 141-150.

Vogler, C. (1998). *The writer's journey.*

Young, M. R. (2000). Creating interactive narrative structures: The potential for AI approaches. *Psychology*, pp. 1-26.

Aarseth, E. (2012). A narrative theory of games. *Proceesings of the international conference on the foundations of digital games*, pp. 129-133.

12 APPENDIX

12.1 Test introduction

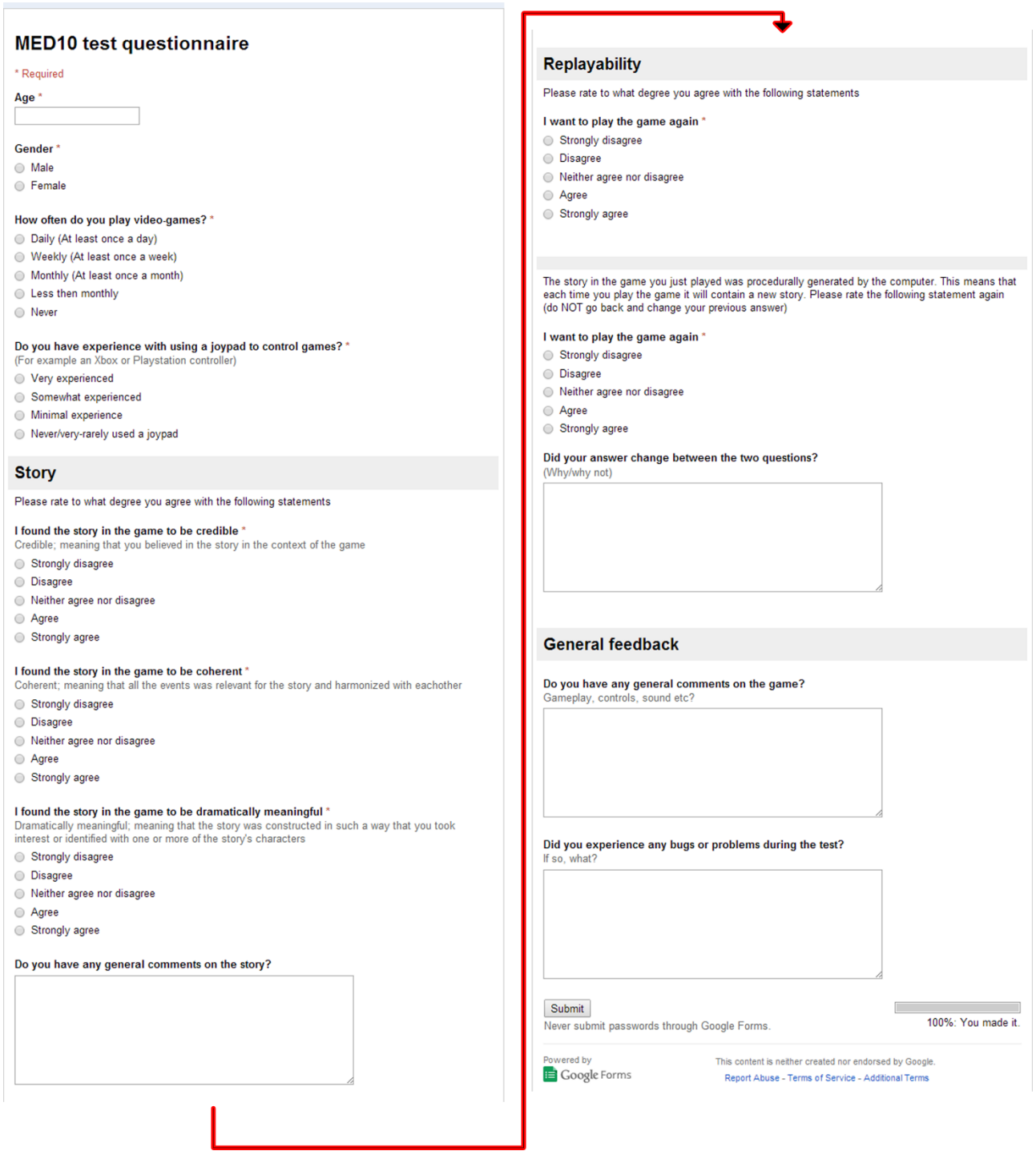
Before we start the test I would like to give you a small introduction.

The test will consist of two parts, first I will ask you to play a small game and after that I have a questionnaire for you to fill out.

The game is a 3rd person role playing game. The character is controlled using an xbox 360 controller. The left stick will be used to move your character around, and the right stick will be used to move the camera. Your primary attack is the right trigger on the back and your secondary attack is the left. The A button can be used to interact with other characters in the game, the Y button can be used to open or close your inventory and the B button can be used to drink healing potions if you have any.

The controls is printed on a piece of paper that will be adjacent to the computer in case you forget any of them. I will be present during the entire test in case you have any questions.

12.2 Questionnaire



MED10 test questionnaire

* Required

Age *

Gender *
- Male
- Female

How often do you play video-games? *
- Daily (At least once a day)
- Weekly (At least once a week)
- Monthly (At least once a month)
- Less then monthly
- Never

Do you have experience with using a joypad to control games? *
(For example an Xbox or Playstation controller)
- Very experienced
- Somewhat experienced
- Minimal experience
- Never/very-rarely used a joypad

Story

Please rate to what degree you agree with the following statements

I found the story in the game to be credible *
Credible; meaning that you believed in the story in the context of the game
- Strongly disagree
- Disagree
- Neither agree nor disagree
- Agree
- Strongly agree

I found the story in the game to be coherent *
Coherent; meaning that all the events was relevant for the story and harmonized with eachother
- Strongly disagree
- Disagree
- Neither agree nor disagree
- Agree
- Strongly agree

I found the story in the game to be dramatically meaningful *
Dramatically meaningful; meaning that the story was constructed in such a way that you took interest or identified with one or more of the story's characters
- Strongly disagree
- Disagree
- Neither agree nor disagree
- Agree
- Strongly agree

Do you have any general comments on the story?

Replayability

Please rate to what degree you agree with the following statements

I want to play the game again *
- Strongly disagree
- Disagree
- Neither agree nor disagree
- Agree
- Strongly agree

The story in the game you just played was procedurally generated by the computer. This means that each time you play the game it will contain a new story. Please rate the following statement again (do NOT go back and change your previous answer)

I want to play the game again *
- Strongly disagree
- Disagree
- Neither agree nor disagree
- Agree
- Strongly agree

Did your answer change between the two questions?
(Why/why not)

General feedback

Do you have any general comments on the game?
Gameplay, controls, sound etc?

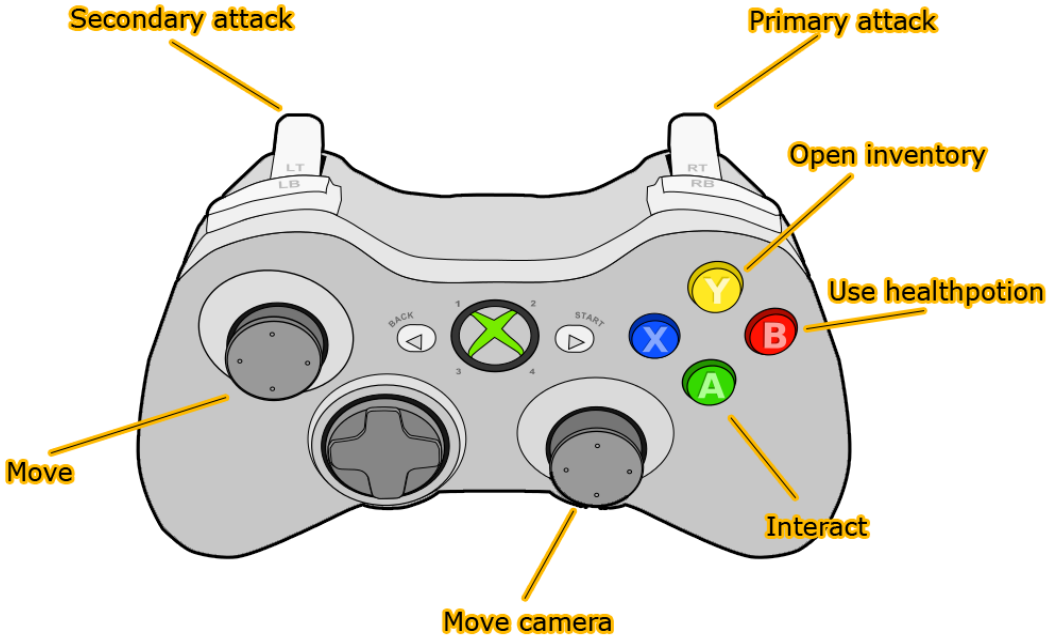
Did you experience any bugs or problems during the test?
If so, what?

Submit
Never submit passwords through Google Forms.
100%: You made it.

Powered by Google Forms
This content is neither created nor endorsed by Google.
Report Abuse - Terms of Service - Additional Terms

12.3 Game controls

MAIN GAME



SHOP/INVENTORY

12.4 Generative grammar rules for story structure

Rule	**Replace with (Red = new rule, Green = symbol)**
Story	Ordinary world + Call to adventure + Meeting mentor + Crossing first threshold + Test, Allies, Enemies + Approach to innermost cave + The ordeal + Reward
Ordinary world	Background story
Call to adventure	Puzzle + Conversation with herald
Meeting mentor	Conversation with mentor + Receive gift
Crossing first threshold	Leave town + Conversation with threshold guardian + battle
Crossing first threshold	Conversation with threshold guardian + Battle + Leave town
Crossing first threshold	Conversation with threshold guardian + Battle + Puzzle + Leave town
Test, Allies, Enemies	Battle + Get Ally + Puzzle
Test, Allies, Enemies	Battle + Get Ally
Test, Allies, Enemies	Puzzle + Battle
Test, Allies, Enemies	Battle + Test, Allies, Enemies + Puzzle
Test, Allies, Enemies	Puzzle + Battle + Puzzle
Test, Allies, Enemies	Battle
Test, Allies, Enemies	Cave
Test, Allies, Enemies	Puzzle + Cave
Test, Allies, Enemies	Battle + Cave + Battle
Approach to innermost cave	Enter town + Leave town
The ordeal	Conversation with opponent + Battle
Reward	End of game
Get Ally	Ally join + Test, Allies, Enemies + Ally leave
Get Ally	Ally join + Test, Allies, Enemies + Ally die
Cave	Enter cave + Test, Allies, Enemies + Leave cave
Cave	Enter cave + Get Ally + Leave cave

12.5 The story used in the game for the test

```
<backgroundstory>
	This is the story of Siljyq
	Siljyq lives in the small town of Trey in the kingdom of Llanfirth
	Everything was fine in the kingdom of Llanfirth until one day when the dead rose
from grave.
</backgroundstory>
<puzzle />
<conversation who="Herald">
	<other>Excuse me, are you [heroname]?</other>
	<you>Yes. </you>
	<other>Oh good, I need to talk to you.</other>
	<other>You don't know me. My name is [othername] </other>
	<other>I have bad news, Hometown is in danger</other>
	<you>That can't be good</you>
	<you>What should I do?</you>
	<other>Go talk to [MentorName], the Mayor, he will know</other>
</conversation>
<conversation who="Mentor">
	<you>Excuse me, are you [othername]?</you>
	<other>Yes. </other>
	<other>I'm [othername] </other>
	<you>Oh good, I need to talk to you.</you>
	<other>Can I help you?</other>
	<you>I'm [heroname] </you>
	<you>Have you heard, Hometown is in danger</you>
	<other>Yes, Rapguf did this.</other>
	<you>Who is Rapguf?</you>
	<other>He is Zombie King</other>
	<you>Oh no!</you>
	<other>You must Kill leader of opposing force. Can you do that?</other>
	<you>Yes</you>
	<other>Fantastic</other>
	<other>One more thing. You should take this, it's a dangerous quest</other>
</conversation>
<RecieveGift what="Sword" />
<conversation who="Thresholdguardian">
	<other>Stop. Don't go any further.</other>
	<you>Who are you?</you>
	<other>I am Enlito, Rapguf's Right hand</other>
	<you>Rapguf's Right hand? Then you must die</you>
</conversation>
<battle />
<puzzle />
<leave what="town" />
<enter what="cave" />
<puzzle />
<battle />
<puzzle />
<leave what="cave" />
<enter what="town" />
<leave what="town" />
<conversation who="Opponent">
	<you>Before we fight, I just want to know one thing. Why did you do it?</you>
	<other>I was kicked out of hell</other>
	<you>That can't be good</you>
	<other>Now prepare to die</other>
</conversation>
<battle />
<GameEnd />
```