Procedural Quest Generation

Improving Skyrim’s Radiant Story

Rui Miguel Moraes de Abreu Quartilho Dias

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisor: Prof. Pedro Alexandre Simões dos Santos

Examination Committee

Chairperson: Prof. Nuno Joao Neves Mamede
Supervisor: Prof. Pedro Alexandre Simões dos Santos
Member of the Committee: Prof. João Miguel De Sousa de Assis Dias

June 2019
Acknowledgments

I would like to thank my mother and brother for their friendship, encouragement and caring over all these years, for always being there for me through thick and thin and without whom this project would not be possible.

I would also like to acknowledge my dissertation supervisor Prof. Pedro Santos for his insight, support and sharing of knowledge that has made this Thesis possible.

Last but not least, to all my friends and colleagues that helped me grow as a person and were always there for me during the good and bad times in my life. Thank you.

To each and every one of you – Thank you.
Abstract

To explore the potential ability of games creating more dynamic, living worlds and stories, we took the existing Radiant Story and AI system created by Bethesda Softworks for Skyrim, which is capable of generating quests at run-time, and sought to understand what were its limitations or what caused it to not be enjoyed, in most parts, by the community and players.

The basis for this project is the research and work done in the areas of procedural generation and interactive storytelling as a way to better dynamically create quests.

As such, we had proposed to improve Skyrim’s Radiant system by making this system create chained, that is, interconnected and sequential, events and quests, while attempting to maintain narrative coherence and characters’ personalities consistent.

The limitations of the base system however, did not allow for the complete development of the suggested structure, but still allowed the analysis of the potential of this type of work, and whether there is worth in continuing to follow it.

Keywords

Story Generation; Dynamic world; Personal narrative; Radiant Story.
# Contents

1 Introduction  
  1.1 Motivation ................................................. 4  
  1.2 Objective .................................................. 5  
  1.3 Organization of the Document .............................. 5  

2 Related Work  
  2.1 Procedural Generation .................................... 9  
    2.1.1 Main Works of interest ............................... 10  
    2.1.1.A Quest Classification and Generation ............ 10  
    2.1.1.B Story Generation .................................. 10  
    2.1.2 Additional works on Procedural Generation of stories .. 12  
  2.2 Agents and Narrative ..................................... 13  
  2.3 Summary of Important Points .............................. 15  
  2.4 Artificial Intelligence in Skyrim ......................... 15  
    2.4.1 Papyrus Script ...................................... 15  
    2.4.2 Radiant AI .......................................... 16  
    2.4.3 Quests ............................................. 16  
    2.4.4 Radiant Story ...................................... 19  

3 Solution  
  3.1 Architecture Design ...................................... 25  
  3.2 Process interaction with game system ..................... 27  

4 Implementation  
  4.1 Issues and Problems ...................................... 31  
  4.2 System .................................................... 32  
    4.2.1 Chain Quests ....................................... 32  
    4.2.2 Base Quests ........................................ 34  
    4.2.3 Main points of divergence from planned structure ... 35
5 Validation and Evaluation Tests
  5.1 Solution Evaluation Method .................................................. 39
  5.2 Result Analysis ................................................................. 40

6 Conclusion
  6.1 Conclusion ................................................................. 49
  6.2 Future Work ................................................................. 49

A Questionnaire ................................................................. 55
List of Figures

2.1 Data tab of a quest ......................................................... 17
2.2 Quest Stage tab of a quest ............................................. 17
2.3 Alias tab of a quest ...................................................... 18
2.4 Dialogue Views tab of a quest ....................................... 18
2.5 Scenes tab of a quest ................................................... 19
2.6 Simple exemplification of the architecture of Radiant Story ........ 21
3.1 Exemplification of the basic solution architecture ................ 26
4.1 Exemplification of the project implementation ...................... 33
5.1 Amount of time each tester spent with the game ................. 40
5.2 Testers’ response regarding interest in random content generation ........ 41
5.3 Testers’ response regarding Skyrim’s radiant quest .............. 41
5.4 Testers’ response towards the implementation of the prototype’s quests .................. 42
5.5 Testers’ response regarding the prototype’s quests: Repetitiveness ................. 43
5.6 Testers’ response regarding the prototype’s quests: Quality .......................... 43
5.7 Testers’ comparison between Skyrim’s and the prototype’s random quests ........ 44
5.8 Testers’ opinion on the most important aspects of random content generation .......... 44
5.9 Testers’ response regarding addition of randomly generated content to a game ........ 45
5.10 Testers’ response regarding usage of randomly generated content in a game .......... 45
# Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>Artificial Intelligence</td>
</tr>
<tr>
<td>ALP</td>
<td>Author Level Plans</td>
</tr>
<tr>
<td>BDI</td>
<td>Belief Desire Intention</td>
</tr>
<tr>
<td>GST</td>
<td>Generalized Story Template</td>
</tr>
<tr>
<td>HTN</td>
<td>Hierarchical Task Network</td>
</tr>
<tr>
<td>MMO</td>
<td>Massive Multiplayer Online</td>
</tr>
<tr>
<td>NPC</td>
<td>Non-Player Character</td>
</tr>
<tr>
<td>QSE</td>
<td>Quantifiable Story Element</td>
</tr>
<tr>
<td>RPG</td>
<td>Role-Playing Game</td>
</tr>
<tr>
<td>TRAM</td>
<td>Transform Recall Adapt Method</td>
</tr>
</tbody>
</table>
Introduction

Contents

1.1 Motivation ................................................................. 4
1.2 Objective ................................................................. 5
1.3 Organization of the Document ................................. 5
In order to give meaning to players’ actions in video games, usually an objective is given, some of them simple and straightforward, others more complex or abstract. In Role-Playing Game (RPG) objectives tend to come in the form of quests, a singular or group of tasks that reward the player in some way, when, usually, successfully completed. However, creating these quests in a meaningful amount and of reasonable quality, requires a large amount of effort, work and time. It also involves several fields of work, mainly writing and programming, not to mention any additional graphic assets and sound effects that may be needed.

The work necessary involves aspects such as creating Non-Player Character (NPC) to give these quests to the player(s) and/or be involved in the quests; the writing of the motivation, reason and story behind the quest; planning and creating the steps/tasks that the player(s) must fulfill to finish the task; possibly different states depending on how the player completed, successfully or not, the quest(s); as well as establishing possible connections between different quests.

As such, a solution that was found was to allow the game itself to generate, at run-time, these quests as much as possible. In other words, automatically generate or activate quests while a game is running depending on need and/or situation. This comes in part as an attempt to reduce the number of hand-made and designed quests that are needed. The generation of quests helps in reducing the workload on those responsible for creating these experiences, while maintaining some of the ability of quests in transmitting the illusion of a living and dynamic world in games, as well as giving purpose and objectives to the player(s).

Our intention is to help increase the number of experiences and events a player can witness, participate in and share, potentially increasing replay value, retaining and entertaining more players, for a longer period of time. In a way, it can be said that part of the inspiration for this project also comes from the players’ sharing of the stories they experienced, the adventures they had, the achievements and discoveries they made in games, and a desire to help increase their number.

These unique experiences and sharing of them can be seen in many games, and we looked at some of them whose gameplay and system(s) had the disposition to allow players to create and share their own unique experiences. We mostly focused on observing those that we could more easily work with, or modify. Most of them were sandbox games, such as Minecraft [12], and strategy games, such as Crusader Kings [13] and Civilization [14].

It is important we know what allows games to provide players with unique experiences. An important aspect that was common between different games’ ability to provide unique experiences was the randomization of environments, locations, enemies, items and/or other parts of a game, which can be seen in games like Minecraft [12] and Civilization [14]. This is usually done through procedural generation of certain parts of a game, such as terrain generation which is the most common and easy to observe. That by itself would not be enough to provide much of an unique experience, but is a relatively important
aspect to use in games, at least in those that aim to give different experiences to players each time they are played.

Part of what allows strategy games (Civilization [14], Crusader Kings [13]) to create so many unique experiences comes from the interaction between players. As a mix of competition and cooperation tends to be required to win in such games and with the players, including computer controlled ones, choosing different approaches to reach a victory condition, conflicts between interests will arise, increasing the number of possible scenarios. This can be seen in the experiences that players share.

This is however not entirely practical for games of other genres (non Strategy games) mostly due to the amount of time it would need and take for all intelligent entities to evaluate their objectives, and determine a suitable course of action to fulfill those objectives during run-time, at least at present time. While this point is partially dependent on the quantity and quality of intelligent entities, the number of possible actions and results also play part in it. As such, events and quests are used to create situations which the players can participate in, interact with and influence the game world, helping to increase the dynamism of the game world without having to sacrifice as much performance.

We have chosen to explore and work on these problems by modifying the game Skyrim, from Bethesda Softworks, known for allowing players to act as they wish.

The reasons that led us to choose Skyrim, over other potential games, are its openness and easiness to modify and work with, due to the tool used to create its content (Creation Kit) being made available by the developers themselves; the existing information and documentation on its systems made by the community; the game’s community itself; as well as due to being one of the few games which already had a basis for creating its own unique experiences, which is the Radiant system, with it being capable of generating quests. Thus giving us a small basic sample for comparison.

The Radiant system is defined in two different parts, the Radiant AI, which revolves around making NPC try to complete their personal objectives, which forms their basic behaviour, and the Radiant Story, which is a system that dynamically generates new quests, which is the main part that interests us.

The Radiant Story can be seen as an attempt to fulfill the problems mentioned above but, both in its implementation and usage, various limitations and issues can be witnessed. This has caused players to feel the game to be more repetitive, or become so faster, when dealing with the quests and events generated by the Radiant Story. We detail and describe our analysis of the Radiant systems in more depth a bit later, in Chapter 2 of this document, due to its role and importance in our project.

1.1 Motivation

The time, work and effort necessary for designing and creating quests are the main limiting factor, and reason, for the current industry to make use of one of two main approaches. Carefully and uniquely
hand-crafted quests that require a fairly high amount of development time and effort and usually short in number, like in the Witcher [15], or a large amount of simple and repetitive quests (kill, fetch, etc.), like those most commonly seen in Massive Multiplayer Online (MMO) such as World of Warcraft [16] or Destiny [17]. Most games, like Skyrim, tend to have a mixture of both, to varying degrees of effectiveness.

In the end however, none of these approaches seem to produce truly satisfying results at mid to long-term, ending up being either repetitive, shallow or not having enough quests (motivation) to offer to the player(s). Also, in the cases where a good amount of quality quests is available (e.g. Witcher), it comes at the cost of high development time and effort, which may not always be justifiable.

As such, by building upon the Radiant Story system existing in Skyrim that is capable of generating quests to a certain extent, it should be able to provide a wider array of possible experiences. At least, it should be possible to improve this system to allow the generation of a larger variation of quests. By providing developers with such a system it should also help ease the burden and effort in developing content.

1.2 Objective

Our objective will be to improve the diversity, depth and the interest of the automatically generated quests, by implementing and adding a new component to Skyrim’s Radiant system, based on procedural generation techniques.

In order to accomplish this, we will have to be able to maintain narrative coherence in the back story of the quest(s) as well as attempt to keep consistent the personalities of the characters that end up involved in these quests, while keeping the quest objectives related to said narrative and characters’ personalities.

We intend to verify if dynamically generating quests will allow a wider range of possible experiences that players of a game can have, while avoiding an increase in the work and time required of developers.

1.3 Organization of the Document

In the next Chapter, Chapter 2, we look at the subject field(s) that best supports the accomplishment of our goals and their related work. We also analyze and describe the system that we will be working with.

In Chapter 3 we describe the solution that was used, and that is analyzed in this document.

Chapter 4 describes the issues and problems that had arisen in the development of a prototype, for user testing our proposed solution. It also gives an overview and description of the prototype’s system, as it was implemented.
Analysis on the results of user tests can be seen in Chapter 5.

Finally, the conclusion reached with this project and our suggestions to potential future approaches in similar areas can be found in Chapter 6.
2 Related Work

Contents

2.1 Procedural Generation ........................................... 9
2.2 Agents and Narrative ........................................... 13
2.3 Summary of Important Points ................................. 15
2.4 Artificial Intelligence in Skyrim ............................. 15

7
The aspect of providing variable, unique experiences depending on the actions of a player tends to belong to the area of interactive storytelling, which has seen a large amount of research and work poured into. Interactive storytelling is when the story and narrative change and adapt according to the interaction between a player (or players) and the system (game). Usually, their setting, characters and back story are clearly defined and set, though in some cases there are some dynamic components.

A relevant and interesting notion for us and our problem, and which is related in part to interactive storytelling, is that of emergent narratives. They are stories that emerge from the interaction between players and a game’s system(s). These are personal to a player, not being authored by anyone in concrete and being something akin to one’s own life, that is something that is born from one’s decisions and actions in a world.

However, interactive storytelling remains something hard to achieve at a truly satisfactory level and there is still a lot of room for improvement. Especially in concern to providing meaningful and interesting results, while being practical and viable for more commercial products, that is, without requiring a large amount of human input and hand-made content.

There are a wide variety of games that are able to provide variable, unique experiences to players, and a number of systems that enable games to do such. That is, there are various techniques and systems that allow each player to have different experiences on different game runs (playthroughs) or sometimes even in the same run, when a player attempts different actions. However, they have a number of limitations and, after some time, the experiences they provide cease to be unique or as varied.

Following is a look at two commonly and widely used systems/techniques in providing the player(s) with variable and adaptive situations (experiences). They are procedural content generation (of parts of a world/system) and agents used in narrative.

### 2.1 Procedural Generation

Procedural generation has been part of video games for a good part of their existence. They most notably appear as map and terrain generation, such as in games like the Civilization series, rogue-likes/dungeon crawlers and endless runners. The dynamism of the game world still tends to be heavily scripted, following previously defined guidelines and never straying from them, or relying on actions of players, unless another system is used to change it.


Procedural generation can be simplified as taking different parts (pieces/blocks) it has access to (can use) and creates a new, different whole, by following a set of existing rules.
2.1.1 Main Works of interest

2.1.1.A Quest Classification and Generation

A work that was done revolving around the potential dynamic generation of quests, and also serving as inspiration for us, was “A Prototype Quest Generator Based on a Structural Analysis of Quests from four MMORPGs” by J. Doran and I. Parberry [1]. In it, they analyzed the existing quests of MMORPGs and categorized them depending on the actions required to complete them successfully and the motivation behind them. They also designed a model for the dynamic generation of quests for games based on those categorizations.

Besides this one [1] and out of several works in the area of story generation we looked at, two stood out. This was due to having similarities to our objective, similarities to procedural generation and a decent number of useful aspects that could potentially be incorporated in our project.

2.1.1.B Story Generation

One of those two main works was “A Framework for Coherent Emergent Stories”, by Gail Carmichael and David Mould [2], which talk and take advantage of the fact that stories in games can be or are represented as graphs.

This work mentions that approaches to designing systems for emergent stories range from character-based to plot-based, or described in another way, from a bottom-up approach (characters drive the story) to a top-down approach (story drives the characters).

They state that the framework they designed in this work is closer to the plot-based end of the spectrum. Since we seek to avoid having and/or using a fairly large number of high complexity agents (for the NPCs) in our project, this framework will hopefully serve well as inspiration and guideline, helping us in designing and building our solution.

In their work, they mention that a notion central to their system is categorization of scene nodes as either kernels or satellites. Each scene node should represent a specific event that occurs in the world. Kernels represent plot points and are like a story’s skeleton, without which the story would not survive (would be incoherent or incomplete). Satellites serve to flesh out the story, develop character(s), and illustrate theme(s) and any individual satellite could be skipped without penalty.

They define for their framework that “a Quantifiable Story Element (QSE) quantifiable story element (QSE) is a numeric or boolean variable representing some story element in a quantifiable way”[2]. A collection of state values, one for each QSE, defines the story state, which is used to calculate and determine the priority values for the scene nodes. The value increases as that element feature is more prominent.

This ties into how the outcome of a scene modifies and adjusts the specified story state values after
A scene has been presented, since certain scenes complete certain objectives such as a comic relief scene being used to lower the story’s tension.

Other relevant terms, in the context of this work [2]:

**Prerequisite**: condition(s) that determine the availability of a scene and each scene can have zero or more prerequisite

**Priority**: given to each node and used to make available to players the node with the highest priority, calculated as the sum of modifier values (aggregate score of a category of QSE)

The other main work was “Minstrel Remixed: Procedurally Generating Stories” by Brandon Tease, Noah Wardrip-Fruin and Michael Mateas [3]. It consisted in a reverse engineering and re-imagining of the work “Minstrel” by Scott Turner, a system which aimed to generate coherent stories. “Minstrel Remixed” is, quoting the authors, “an imperfect recreation of the original “Minstrel” based on written descriptions, since neither the source nor a working copy is available”.

In “Minstrel Remixed”, like in the work mentioned above [2], graphs are used as primitives to represent a story. This system, both the original and remixed, were made up of two main components, the Transform Recall Adapt Method (TRAM) system, a case based reasoner which modifies story details, and the Author Level Planning system, which enforces constraints and improves stories as they are generated.

**TRAMs** can be described as bundles of transformation operators, that can be selected and swapped depending on necessity and intention. They are responsible for editing story graphs and designed for helping recall useful information from the story library.

Simplifying, TRAMs are able to find cases in the story library which aren’t easy to recognize as being applicable.

If equating this work [3] to procedural generation, then one of the parts or pieces for building the stories would be the story templates, Generalized Story Template (GST).

**GSTs** have the same structure of completed story graphs but are typically full of holes or temporary variables that need to be filled or changed. In Minstrel Remixed, the first step in any story creation is always to use the given requirements to select an appropriate GST, which is then installed into the story graph for the remainder of the system to work with.

These aspects of Minstrel Remixed have some similarity to Skyrim’s Radiant Story system for quests (detailed in a later section (4) of this chapter). However, Radiant Story is far more simplistic with it simply choosing the participants of a quest and not trying to adapt its quests like Minstrel Remixed adapts its stories. In a way, we can say that Skyrim makes use of GSTs but not TRAMs.

Additionally, there’s the Author Level Planning system which guides the story construction at a high level through a number of Author Level Plans (ALP) it contains. The ALPs’ job is to look at the current state of their target, the current story graph, and plan modifications to it with the objective of filling the
entirety of the story graph in a desirable manner. This serves as a director for the story, something that Skyrim's Radiant Story also does not use, besides filling in the empty spots, such as the participating characters (NPCs) or locations.

Both of these works’ algorithms, [2] and [3], give and offer a way of generating a potential narrative story, while our objective is based on generating and choosing the actions (quests and events) required to (help) drive a narrative in a game. These works done on generating the narrative story itself offers us clues, ideas and a basis for our project to work on, since the actions that are requested of a player must have a certain level of relation and coherence with the provided narrative.

2.1.2 Additional works on Procedural Generation of stories

There are some other works that, while not as relevant for our project as the two mentioned above, contain interesting and different approaches to story generation, and should be taken into consideration.

“Automated Story Generation with Multiple Internal Focalization”[4] by Byung-Chull Bae, Yun-Gyung Cheong and R. Michael Young, explores the generation of narratives with more than a single point of view, that is, presenting the story from the perspective of different characters. They use a planning based approach to generating the stories, where a story is represented by a plan which is made up of a series of plan steps and their temporal and causal relationships and where each plan step corresponds to an event.

“Making Sense of Emergent Narratives - An Architecture Supporting Player-Triggered Narrative”[5] by Simon Chauvin, Guillaume Levieux, Jean-Yves Donnart and Stéphane Natkin, explores the various aspects of emergent narrative and attempts to better define it and its characteristics, developing a system that tries to interpret the actions of a player in order to then adapt and modify certain parts of a game.

“Out-of-Character Approach to Emergent Game Narratives”[6] by Simon Chauvin, Guillaume Levieux, Jean-Yves Donnart and Stéphane Natkin, has an approach that is on a higher, meta level, one where they enable the player to control certain parts of the story’s aspects, focus and/or content outside of the normal play style of a game, by allowing them to trigger processes that then modify the game. In this work they define and categorize various aspects and characteristics of emergent narratives, much like in their other work mentioned above[5].

The interesting aspects of this work are mainly the definition of some important properties of emergent stories/narratives, which are:

**Coherence:** the existence and usage of consistent rules to maintain the coherence of the game world

**Agency:** the ability given to players to have an impact on the game world

**Possibility Space:** the number of possibilities that enable a wider array of situations to arise in the game world
Uncertainty: the ability of the game in creating surprising or unexpected situations or events

Co-direction: the role of experiencing and participating in the narrative and its creation given to players

They also mention how research in interactive narratives has two general approaches to story creation and generation, in relation to adding more narrative content while taking into account the players’ actions, which are top-down and bottom-up approaches that were mentioned earlier, in [2].

Going a bit more in depth, the bottom-up approach consists in the story being emergent and resulting from the interaction between the player and autonomous agents. This follows the principle that the characters drive the story and not the story drives the characters. Meanwhile, the top-down approach has a manager responsible for ensuring that the generated stories respect certain authored conditions.

These approaches that can also be described as being a character-driven narrative (bottom-up approach) or plot(author)-driven narrative (top-down approach), with procedural generation tending to be closer to a top-down approach.

“Modelling Narrative Conflict to Generate Interesting Stories”[7] by Stephen G. Ware and R. Michael Young, uses planning based algorithms and the aspect of many stories emerging from conflicts between characters to develop a system for generating stories.

“Personality Traits in Plots with Non-deterministic Planning for Interactive Storytelling”[8] uses Hierarchical Task Network (HTN) planning as basis for their work and then attempt to add non-determinism to the planning algorithm. This is in order to expand the number of possible plot variations, additionally having a decision-making process based on personality traits to determine the behaviour of the characters participating in a story, thus generating a story with a focus on characters.

There are more works that attempt to generate stories but make use of a different approach through agents.

2.2 Agents and Narrative

Agents are used in a multitude of ways, but in this section we will be mostly referring to them when they are used as individual entities, such as NPCs.

They are one of the ways usually used to provide a challenge to the player, one that (might) adapt and act differently depending on the situation. Agents are also used to provide some dynamism to NPCs, whether as a player’s allies or opponents, and sometimes used to simulate something akin to a manager.

The main limitation of agents is the time that it takes the AI to evaluate the situation, or state of the game, and choose an appropriate course of action. In many cases it is not an issue, but as the more complex the situations become and larger the number of actions available to the Artificial Intelligence (AI)
and other entities, the longer it will take to complete the evaluation and choose an (appropriate) action. The number of such entities is another factor to consider when faced with those limitations.

Overall, using a larger amount of Agents and each with higher depth and detail would usually be preferable, in order to create and enable a far more dynamic world. Sometimes this is not feasible however, usually due to the limitations mentioned. The best known results, of such a type of approach, would be those of “Crusader Kings II” [13], and other similar games, developed by Paradox Interactive.

In part due to aiming to create more character focused stories, that is use something closer to a bottom-up approach, the majority of works in the area of generating and creating dynamic stories explore and make use of agents as characters.

However, in large worlds with a large number of characters, having all those characters be detailed and have depth is not entirely possible, due to the mentioned processing limitations, though it is a decreasing concern. As such, most approaches of this type either have a very small number of characters or have characters with differing levels of detail and depth.

Some of the works we looked at made use of agents to create a manager or director for their stories, or, as is known in pen & paper RPGs, a Dungeon Master. The Dungeon Masters in pen & paper RPGs are a player responsible for managing the narrative of the story. As such, those works end up with an agent playing the role of Dungeon Master, guiding or controlling the narrative, which is closer to top-down approaches. That agent would observe the world’s state and player’s actions and based on that it would modify the story, activate quests and/or events, and so on, making this an interesting alternative to procedural generation.

“Flexible story generation with Norms and Preferences in CRPG” [9], by Edward Booth, John Thangara-jah and Fabio Zambetta, is a work that makes use of a Belief Desire Intention (BDI) based agent to create a Game Manager responsible for managing the story that is presented to players. In a way, it simulates the Dungeon Master from pen & paper role-playing games such as “Dungeons & Dragons”. It uses norms and preferences to help the Game Manager agent better evaluate how to better continue the story.

“Ice-Bound - Combining Richly-Realized Story with Expressive Gameplay” [10] is work by Aaron A. Reed, Jacob Garbe, Noah Wardrip-Fruin, and Michael Mateas, which takes an incomplete story and uses an agent as an author-surrogate and an approach similar to building blocks to engage the players in completing the story, generating and narrating the story depending on how the building blocks, designated as “symbols”, are combined and arranged.

An example that is often cited by other works is “Façade”, a game developed by Michael Mateas and Andrew Stern[11], and about which they published some academic works detailing and analyzing it. “Façade” is a work inserted in the study of interactive stories and makes use of a number of agent behaviour techniques and of a drama manager. It has two very detailed agents responding to the player’s
actions but has a limitation of the context being highly constricted and clearly defined.

2.3 Summary of Important Points

Of the related work that was found and analyzed, one of the most important aspects we found was the categorization of quests, made by Doran and Parberry [1].

One other that seems to be useful, or practical, is the usage of graphs, as shown by "A Framework for Coherent Emergent Stories" [2] and "Minstrel Remixed" [3], to represent stories. This allows one to attribute to each node (of a story graph) a number of attributes, such as prerequisites, post-conditions, as well as meanings (as values or labels). These attributes help developed systems to evaluate and choose which node is the most suitable to follow the currently active story node and state of the world, and to modify and/or adapt an existing node depending on the needs and intentions of the story. Also, the ability to alter and adapt graphs suggested and used by "Minstrel Remixed" [3] is interesting and with a high potential for increasing the number of possibilities that can be generated.

Another useful and interesting point is the categorization and definition of narrative aspects and content, which are important in motivating and driving the player to action.

As mentioned previously, the usage of fairly complex agents in large amounts is limited by the processing power available and the time available to maintain a real time simulation (of the game world). Given that, the usage of procedural generation seems to be a better approach for solving our problem, and will most likely end up being a top-down approach (where the story controls the characters).

2.4 Artificial Intelligence in Skyrim

In Skyrim, the Radiant system is responsible for the AI aspects of the game, being made up of 2 components or subsystems. The Radiant AI, responsible for the existing in game agents which are the NPCs (designated as actors), and the Radiant Story, responsible for handling the dynamic quests in Skyrim.

To help designers and programmers in creating and controlling quests, Skyrim makes use of Papyrus Script, a scripting language, and one of the fundamental parts in developing this project.

2.4.1 Papyrus Script

Papyrus is a scripting language that works by receiving events from the game and then sending function calls to it.

“At its heart, Papyrus is object oriented in nature.”

As Papyrus is a language made for the development of Bethesda's games, most, if not all, information can be obtained from their site for supporting Creation Kit (their game world creation tool).

One of the main aspects we will be taking advantage of is that Papyrus allows us to initiate a quest at practically any time and situation, due in part to Papyrus Fragments. These are small scripts that run at certain times, such as when a quest step begins or ends.

Papyrus is what will allow us to develop the system whose job will be to evaluate, select and control which quests to use or activate dynamically at run-time.

2.4.2 Radiant AI

One of the parts of the Radiant system is the Radiant AI, which defines the base behaviour of NPCs and is made up of packages. The packages are the building blocks of the behaviour of NPCs, describing which actions to perform at a given time, allowing developers to maintain an illusion of a living world.

The important point of these packages is that when a quest is initiated it may add new packages to the participant NPC(s), allowing their behaviour to change at run time. Combined with the dynamic generation of quests, this gives us a larger range of options to use in giving more unique experiences to players. In this project however, this will not be the focus but merely a useful and interesting tool to consider in the development of our project. This is useful mainly due to being possible to add Papyrus Fragments to certain packages, allowing actions of NPCs to affect quests, though it might make it harder to maintain coherence (of the narrative and/or the NPCs' personalities).

2.4.3 Quests

The quests in Skyrim are made up of different parts, of which the following are the most important and play the most vital role in creating quests, and as such are important in our project:

- **Data:** general information about the quest, including which in-game event triggers the quest (if there is an associated event)
- **Stages:** the steps that make up a quest, and what happens when a step begins and/or ends (scripting)
- **Alias:** descriptors of all involved participants (Characters, Items and Locations)
- **Dialogue Views:** the dialogue (tree) that a player can have with another character involved in the quest
- **Scenes:** scripted, clearly defined, sequence of actions that involve some or all of the characters involved in the quest

The Data tab, see fig. 2.1 is mostly general information such as the name of the quest, but also contains a small but important part which is a mark of what trigger events (situations) can initiate that
The component most useful and important to us will be the Stages part, as seen in fig. 2.2, due to allowing us to create scripts (pieces of code) that are ran either when a new stage starts or when a stage is over. This allows us to trigger and initiate new quests when a stage is reached, and as such, further improve and create better correlated quests, basically granting us better control over when a quest begins, continues or ends.

The Alias tab, which can be seen in fig. 2.3, is a list of the required and optional participants, for a quest, be they Characters, Items or Locations. These participants are what makes the difference between a normal, standard quest and a dynamic(ally generated) quest, designated as Radiant Quest, as they can be chosen explicitly by naming or choosing them directly, or abstractly, only specifying which
aspects they must adhere to, such as whether they are male or female (for characters), if they belong to a faction, and other possible conditions.

The standard quest will have only aliases that explicitly declare who represents them, while a Radiant Quest can have as many of its aliases be abstract as it wants.

The aliases, besides the conditions limiting what (Character, Item, Location) will be chosen to portray it, also have other properties that are added, with a higher priority, to their respective chosen representative. This is particularly useful in the case of NPCs as it opens the possibility to increase the dynamism of the world, by adding new packages (the NPCs’ behaviour modules) to NPCs, thus altering their behaviour.

Dialogue views, as in fig. 2.4, are the dialogues that a NPC has available during the associated
quest, when interacted with by a player. These dialogues can also have associated scripts, when certain options in the dialogue are chosen, as such, it provides developers and designers with different ways to progress a quest. However, since the dialogue has to be written before being made available, it is harder to provide with interesting or meaningful conversations for players in dynamic generated quests, as there is a high risk of causing dissonance or contradictions in the quest’s narrative, and so the dialogue has to be written in the most generic way possible.

Scenes, see fig. 2.5, are an act played by actors that belong to the quest, where we can have the actors execute actions they normally would not do, such as talking to other NPCs, make the actors move to where we want or need to have them and a number of other actions. The interest of Scenes to our project would be the ability to provide some more realistic or engaging experiences, by allowing NPCs to act in ways that can not be controlled effectively by the behaviour system of packages (Radiant AI). We can, for example, create battles between NPCs, have NPCs being assaulted or robbed by other NPCs, and other situations that allow players to get involved and engaged directly with the world, without directly informing the player that it is a quest, which creates more potential situations of unique experiences. Though the scenes have to be pre-made since they are not dynamic, by having a Radiant Quest with only a scene and then use them when needed/possible it is possible to simulate some dynamism with scenes.

2.4.4 Radiant Story

The Radiant Story is the system responsible for dynamically generating quests, allowing a pseudo-infinite number of quests, being a solution with some similarities to procedural generation for maps. The building pieces that are used to make up new quests are the existing NPCs, locations and items in
the game’s world, and most importantly template quests, that is, quests whose participants (characters, locations, items) are abstract until the system generates a new quest, designated in Skyrim as “Radiant Quests”. As they are still quests, they can also possess specific participants, as well as create new ones if necessary.

This way, the Radiant Story system manages to provide a functional and tested system, that seemingly solves the objectives that we had initially stated. However, it has a number of issues, that made players find it repetitive, forced and boring. Additionally, an exemplification of its structure can be seen in fig. 2.6.

The “Radiant Quests”, that serves as a simple template quest, has only the actions involved in the quest specified, mainly the steps that make up the quest, with most if not all participants not being defined concretely until that quest is initiated. The main issue that this project will aim to work with and improve on are these template quests and events, which are severely simple and disconnected, being simple “Kill ‘NPC’” or “Recover ‘Item’ from ‘Location’”, becoming repetitive and boring fast.

The Radiant Story system tries to choose the participants of a Radiant Quest, whose conditions for activation/creation were met, at run time, attempting to follow the limitations specified on the participants that the template quest requires. In choosing the participants in a newly initiated Radiant Quest, the Radiant Story system prioritizes locations where the player has not yet been, but this parameter is not possible to change in the quests themselves, being an internal parameter of the Radiant Story system.

This leads to another issue, which is how to choose or create the intervening characters, locations and items, though this will not be the focus of this project. The problem in this point lies in the fact that the intervening participants can be poorly connected, for example, a farmer can have lost his shovel in the deepest part of a catacomb filled with monsters, on the other side of the continent, creating a dissonance in a player’s mind, between the game’s story and game-play, and breaking their immersion or suspension of disbelief, thus ruining their experience. Although some limitations can be placed in choosing the participants, due to the system internally prioritizing unexplored areas, situations like the exemplified one are relatively common.

Additionally, in the Radiant Story system there exists a component named Story Manager Event Node, which contains a group of possible trigger events for quests (possible situations that can initiate a quest). Each one of those events contain a hierarchical tree of quests, with additional conditions necessary to trigger and create such a quest. As such, in order for the system to know which quests are Radiant Quests and can be triggered (created), they must be added to the Story Manager Event Node. A quest also requires being marked in the Data tab of the quest itself which of the trigger events will be responsible for it. Given this, it is important to take notice of this trigger event association, for better control in initiating, directing the chained quests, and ensuring little to no issues arise from this.

There are some other games that have something akin to quest generation. However, just like with
Figure 2.6: Simple exemplification of the architecture of Radiant Story

Skyrim, they are usually systems that are limited to choosing a quest from a pool of existing, potential quests and filling in some blanks, such as characters or items. As such, they are effectively random, but have the problem of being mostly simplistic, garnering the same issues as Skyrim’s.
3

Solution

Contents

3.1 Architecture Design ........................................ 25
3.2 Process interaction with game system ....................... 27
In this chapter, and the following sections, we describe the structure of the suggested solution. Our intended solution to quest generation is closer to a top-down approach (story drives the characters), in relation to story and narrative, since the system would be telling the participants what to do.

### 3.1 Architecture Design

With this project we aimed to improve upon the simplistic template quests and events used to generate “new content” for the player to interact with. The course of action we proposed was to create a generator of chained quests, that is, quests made up of interconnected smaller, simpler quests.

To help with this, we used Doran, J., and Parberry, I. work\[1\] as guideline in order to create the most basic, atomic quests that would make up the chained quests. As for how the generation of the chained quests was done, we used their prototype quest generator as base and inspiration.

While the Creation Kit tool of Skyrim does not provide a tool to directly, or specifically, create and manipulate chained quests, new quests can be started by a simple script command. Said command is also used by Skyrim's developers themselves to create, initiate and control the chained quests that exist in the game. Though the existence of such a tool would help in decreasing development time the development of such a tool was not be part of our work, for this project.

A challenge with generating these interconnected sets of quests is in maintaining their narrative coherence between the different quests that make up a chain quest. That is, to avoid running into situations as described previously which could cause a dissonance between the game and its narrative, part of our objective was to maintain a coherent story for each chain quest. This comes from the need to give players the story and reasoning behind the NPCs’ request(s), as mentioned by Doran and Parberry \[1\], in order to better convince and motivate players to perform the given task(s).

In order to create and control a chain quest, we have Chain quests and Base quests, as can be seen in the figure fig. 3.1.

Chain quests are in charge of selecting the appropriate base quests for a given, or randomly selected, type of chain quest, have the control over which quests and events to activate in the game and know which stage a player is at. As such, they represent the structure of the sequence of events that form a story. Additionally, the different types a chain quest can have, mainly affect how the overall structure is constructed and what selection method for base quests is used. Meanwhile, the base quests consist of the simplest and singular actions a player can or has to do, at a certain stage, in order to advance and progress in the chain quest.

As for choosing which base quest (action) to activate (request the player to perform) in order to progress in the chain quest, we make use of graphs as in the works mentioned above, \[2\] and \[3\]. By having a base quest be similar to a node and attributing it certain values, such as combat involvement,
we would be able to better categorize the quests and facilitate choosing the most appropriate ones. So, we have a chain quest be equivalent to, or represented as, a graph, as presented in the figure fig. 3.1.

One other part of the solution, was to have a separate component for the narrative or story, which can also be seen in fig. 3.1. This would allow a single type of narrative/story to have different possible (or a larger number of) types of chain quests, which could, more easily, increase the number of possibilities.

However, this point is not considered (developed) in the scope of this project, so the testing and analysis of such an approach would be the work of a future project.

If looking at the work of Minstrel Remixed, we could say that the Chain Quest component would be a form similar to the GST. In case the narrative component was added, we could have it communicate with the Chain Quest, in order to allow a dynamic adaptation of the quest and its story, and help in maintaining narrative coherence with the gameplay. This is represented in fig. 3.1 by the "Chain Quest Adaptation and Narrative Coherence Maintainer" component. This would result in the potential addition of the TRAM system, from Minstrel Remixed [3], or one similar to or adapted from it.

So, in summary we have the following components:

**Chain Quest**: Represented as a graph; Manages the structure (possible sequence of events) and conditions to advance in the story

**Base Quest**: Represented as a node of the graph (Chain Quest); A basic form that describes (represents) the most simple of actions that can be executed by the player to advance the quest

**Narrative Quest**: Responsible for managing the story/narrative of a quest and communicating with a corresponding Chain Quest, allowing it to adapt according to that narrative’s needs, intentions and/or progress

So, the Narrative Quest would contain the story content, for example, “Helping a village survive” or “Taking back a castle”, and all its associated story aspects, such as participants and potential situations. The Chain Quest would hold and manage the actions, and their connections, that the player can execute
to progress with the corresponding story, such as asking the player to “Kill bandits”, “Gather weapons” or “Hunt food” when the story starts. The Base Quest would correspond to the individual singular actions, which following the example above, one would be to “Kill bandits”, another to “Gather weapons” and the final one would be “Hunt food”.

Additionally, in order to help the system know which base quests are the most appropriate, besides simply stating the type of the quest, would be for them (each base quest) to have a group of attributes that classify what type of actions are they most inclined towards.

These attributes could be, for example:

- **Combat**: How much combat is a basic quest expected to have
- **Travel**: How much is a player expected to travel, whether things are far away or not from the starting point
- **Talk**: How much interaction with NPCs is in the quest
- **Interact**: How much interaction is the player expected to have with objects or items, such as in puzzles or traps
- **Stealth**: How much is the player expected to actively avoid confronting NPCs
- **Gather**: How much is there a need for obtaining items to complete the quest, for quests such as, e.g., “Gather herbs” or “Collect Bandit trophies”

More, or maybe less, attributes may be required, depending on the (basic) quests that are developed, and the type of game it is (if it was not Skyrim).

### 3.2 Process interaction with game system

In Skyrim’s system, the process would start with the system notifying or requesting to start a chain quest, due to the completion of a requirement or activation of a trigger.

The first part, of the proposed system, to start would be the Narrative Quest component. It would create the basis for a story, depending on the situation of the player and their actions, as well as the cause of triggering the chain quest.

A Chain Quest would then initiate, generate a graph, and start selecting the appropriate Basic Quests. This selection would depend on what was the trigger or condition to initiate the chain quest, and what type of chain quest was initiated. It should also take in consideration the “tension curve” existent in stories, that is, how a story starts somewhat slowly, starts escalating, reaches the peak, and then eases into the ending.

Afterwards, it depends on the player, as they complete Basic Quests (the graph’s nodes), opening new possible Basic Quests to complete, that would eventually lead to the end of the Chain Quest.

At the same time as the player advances the Chain Quest, the Narrative Quest component would
communicate with the Chain Quest as the player progresses. This would permit the Chain Quest to alter itself dynamically (the graph, its composition and basic quests), according to the player's actions, while maintaining the necessary narrative coherence, at least better and more easily.
4

Implementation

Contents

4.1 Issues and Problems .................................................. 31
4.2 System ............................................................... 32
First, we will go through the issues and problems that occurred in the development of this project. Afterwards, we will describe the structure and main components of the prototype as they were implemented.

4.1 Issues and Problems

The main setback to the development of the prototype, to be used for user testing, was Skyrim’s development system itself, Creation Kit. This was mainly due to its “black box” nature, increasing the difficulty of debugging and finding issues within the prototype, as well as its sometimes unexpected and erratic behaviour.

Creation Kit also had a number of limitations that prevented a closer simulation to the intended system, requiring a number of workarounds.

One limitation of the system, is that the (dynamic/radiant) quests are not instanced depending on need and/or circumstance, but simply activated with the dynamic values chosen by the Radiant Story system. As such, no more than one instance of an explicit quest will be active at any given moment, until it is completed. For example, given a dynamic quest “Quest A”, only one version of that quest will be active with there being no second, or third, and so on, “Quest A” active, until it is completed.

To work around this issue we have multiple explicit copies of each of the singular basic quests, acting as a buffer of sorts for that type of quest. For the sake of showing this project’s solution working, this was a good compromise, though preferentially we could have the system create a new instance of a basic quest at any moment in the game, without having to explicitly create these copies. Though a slight modification would be necessary due to us having to consider the existence of a limited number of each basic quest for the implementation, our solution should work with instantiating of quests.

Another similar limitation were the size-fixed arrays, where it was needed to have a predetermined, non-dynamic, size for them. This interfered with the potential number of steps involved in a chain quest and indirectly lowered the number of possible outcomes. It also limited, to a lesser degree, the development process and possible approaches to solve some problems.

There were also some limitations for inserting adaptive dialogues, as nearly all cases of text had to be entirely pre-made. In other words, if one wanted to have even slightly different dialogue for small things, such as changing the name of an item, one needed to write the entire line again. As such, as an additional workaround, it was necessary to have different flags for the different cases and situations, or something else that would act in a similar way to flags, rather than just allow the system to change the text.

This limitation was found mainly on the dialogue of NPCs, which is one of the fundamental parts in transmitting a sense of a living world. Fortunately, the display of objectives did allow for a certain level
of “dynamic” text.

Given all these issues, the amount of work necessary to add different items or other such aspects, did not change much, if at all, from that of a hand-made approach. As a part of this project’s objectives was to reduce the level of required work of developing side, or low priority, content, it can be said that this project failed to accomplish its objective in that regard. Hopefully, with a different base system, or game, this could be remedied. At least, it could be handled better, if the base system was to be made with this included in it, or expecting it to be.

Due to all of these issues, and given the work already included in the prototype, a rather crude and simple prototype for testing was created, consisting of two basic chain quests, each one with 4 steps. As such, given the relative simplicity, and low amount of content, of the testing prototype, the results of the user tests are expected to not be as positive as they could be.

4.2 System

The system, as was developed, is made up of 2 main components, the Chain Quests and Base Quests. Initially, as mentioned in the previous chapter, it was thought of having a third component, corresponding to the quest’s narrative, but the system’s many limitations caused the inclusion of such a component fairly difficult. So, as can be understood, the initially suggested approach of having a separate component for a quest’s related story was not taken, for this project. Refer to 4.1 for a simple visual explanation of the implemented structure.

4.2.1 Chain Quests

The formation and selection of the set of Base Quests that make up each of the steps of a Chain Quest, are handled by the Chain Quest itself, before formally starting. In other words, when a Chain Quest is activated, it will select the Basic Quests it needs, for each of the steps that the Chain Quest has.

All Chain Quests have a property that identifies what type of quest they are. For example, whether they’re a Chain Quest that involves helping a village or building a house, which are the two types used in the prototype. It could also be something like conquering or defending a keep, and so on.

This was mostly in order to allow, or make it easier for, the system to select the best basic quest evaluation and selection process possible. To describe in another way, it’s purpose was to serve as a search parameter for finding the, most appropriate, quest selection process.

As mentioned above, due to limitations of the system, Creation Kit, and its scripting language, Papyrus, a Chain Quest includes a simple array, of a concrete maximum size, of potential quests. On initiation, it will populate the array with the base quests it chooses and determines to be necessary.
Figure 4.1: Exemplification of the project implementation

For this project, a proper algorithm (or system) for, more dynamically, determining the logical coherence of quests was not properly implemented. Only a simple function was made, in part due to the limitations of the system.

As such, the selection function of the base quests was limited for each stage or step, of each type of Chain Quest. In other words, depending on the type of Chain Quest and for which stage a Base Quest is being selected for, there will be a predetermined group of Base Quests it can choose from. As this group of Base Quests is, in a way, pre-selected, a simple random number is then sufficient to "choose" the Base Quest to use.

In the prototype, we also had an array serving as a limiter for the number of quests each step could have. Although, besides one step on each of the designed Chain Quests, all other stages had a pre-defined amount of quests, in the prototype used for testing.

In a more advanced version it could serve to control a potential dispersion of the quests. At least, something similar could be used to avoid having the connections between Base Quests, that make up a Chain Quest, from dispersing too much. Or it could instead, allow a control of the branching of the Chain Quest's possible paths, or developments, creating a greater impact in the player's actions.

As a Chain Quest is a group of sequential, interconnected quests, each(Chain Quest) has its own individual story. So, a Chain Quest tends to also have a curve of tension, or escalation, just like stories do. However, once again, since a proper selection system was not implemented, this point was also not properly solved or evaluated, though the curve was still simulated in the prototype. This is still important
to how Base Quests are (should be) selected, so it remains as a point to be looked at, given potential future work(s).

To summarize the implemented selection process:

1: Chain Quest chooses method to use for selection depending on type of chain quest
2: Chain Quest checks which stage/step it is currently selecting for
3: If there are multiple possible types of a Basic Quest that can be added to a step, one is randomly selected, with all having the same chance
4: Chain Quest adds the selected Basic Quest (one of the copies of its type) to its array of Basic Quests

As mentioned in the previous section on issues that were had during this work, a need for flagging which text should be displayed was necessary. The common case was making use of a property of a Quest entity in Creation Kit, which was the step indicator (of a quest).

This allowed a developer to define the steps that would make up a quest, and number them, attributing them an identifier. Additionally, one of the options available (as part of the development program, Creation Kit) that could be used as condition for texts, as well as some other components such as objects and actions, was this step(stage) indicator that a quest was currently on. As such, some extra, in-between, steps are added, and depending on the situation, the quest would be moved to one of those additional steps, either as a small progression of a step or for substituting a step.

To give an example of Chain Quest, out of the 2 that were implemented in the prototype, one was to “Build a House”, with both having 4 stages. The first step of the “Build a House” was to simply talk to a certain NPC, that would serve as the point of contact for the quest. The second step was made-up of 2 or 3 Basic Quests, that consisted in obtaining items to give to the previous NPC, with the exact Basic Quests being randomly selected. The third step involved removing a NPC occupying the area for building the quest’s house. The final, fourth step consisted in simply talking again to the first NPC that he could go ahead with the construction.

4.2.2 Base Quests

As stated previously, in the system’s description section and problems section above, the main problem with the Base Quests themselves were the necessity to have duplicates of a type. As Quests (in Skyrim) can not be newly instanced at run time, this forced the need to have a number of copies, of a type of, Basic Quests pre-made.

As such, there will always be a limited number of quests of a certain type that can be selected and used, for a single run of the game, in this work’s prototype. Though they could potentially be reset if necessary, there would still be a limit to the number of concurrent quests of each type.
Again, this is a problem particular to how Creation Kit handles quests, so with another system used as basis, such a problem should not occur. Additionally, for the prototyping and testing of this project, such a limitation could be considered relatively minor, by just duplicating each (quest) entry of a type, that is, having a number of copies of a quest available.

In the end, the Base Quests are made up of their steps, that is, how they start, what middle steps are involved and how they end, as well as the base quests participants and the text needed for dialogue or display. In a simple exemplification, first it is told to the player what is required/necessary to be done, then there are the steps the player takes to complete those objectives, which may or not be tracked, and finally the delivery to the requester, if necessary.

4.2.3 Main points of divergence from planned structure

Rather than an array containing the Basic Quests, the Chain Quests should have a (directed) graph. The nodes would be the Base Quests that were selected. Meanwhile, the connection between nodes would serve as a base precondition, indicating which Base Quests would need to be completed to advance in the Chain Quest. It could also, potentially, have been used to form different branches (or paths) in a Chain Quest.

Ideally, we would have the selection be based upon a base quest's type and its parameters, or attributes, as was similarly done in the construction of stories in the reference works of "A Framework for Coherent Emergent Stories" [2] and "Minstrel Remixed" [3].

In the case of a system where it is possible to create arrays of variable dimension, it would enable the consideration of Chain Quests with different number of stages, rather than just a fixed number of steps. In the version for user testing, the number of steps(stages) used was 4.

The Base Quests were intended to have properties, for facilitating and improving the selection process by the Chain Quests, as stated in the previous chapter chapter 3. Unfortunately, due to issues with the base system, it was not possible to define and implement these properties. As such, the selection system consisted of a simple randomizing of a group of predefined basic quests, as stated in previous section about the chain quests.

So, due to how the end version of the prototype turned out, we were unable to determine the usefulness and efficiency such an approach would have.
Validation and Evaluation Tests

Contents

5.1 Solution Evaluation Method ........................................... 39
5.2 Result Analysis ............................................................ 40
In this chapter we describe and analyse the testing method used for this prototype, as well as the results obtained from it.

5.1 Solution Evaluation Method

In order to evaluate the performance of the prototype, a comparison between the game without the system and the game with the system had to be made. So there was a need for testers to play both versions.

As such, the initially intended method to test and evaluate our hypothesis and its solution, through which we believed we would obtain the most meaningful results, was by creating and releasing a mod to the existing community, and requesting their feedback.

This came from part of the objective being to increase the dynamism of a game’s world and allow an additional amount of experiences a player can have, even in different playthroughs. By requesting the existing community we intended to take advantage of the experiences the players have already had playing the original version of the game, thus reducing or even removing the needed for a long testing period.

To complement the reason above, this came from the existence of a reasonably large community, that has already experienced Skyrim’s world in many different ways. As such, we would have a larger number of responses and that were better reflective of this work’s objective, even if the quality of the responses themselves were to be lower and harder to interpret and process, when compared to local tests.

This however, did not provide the expected results, having only obtained 8 responses through this method. The remaining answers to the questionnaire were gathered from other people which were personally contacted, that did have some experience with games, but may not have been related to the game the project was based and developed on (Skyrim).

The test itself consisted in asking the players’ to complete the developed quests, and then answer a questionnaire, which can be see in the appended document, appendix A.

The players were provided with the developed prototype, as a mod, along with previously created save files to decrease the test time. The testing itself was not overseen, and as such not properly enforced, letting the testers perform it when they could or had availability. This was due to the planned approach, of requesting the already existing online community to test the mod, though the same method was used for those directly contacted.

In some of the questions, of the questionnaire, a scale of “1” to “5” was used, with “1” expressing an entirely negative view towards the question’s content and “5” a highly positive outlook. For the precise description of these values for each question, refer to appendix A.
Figure 5.1: Amount of time each tester spent with the game

5.2 Result Analysis

To start with, there were 22 responses to the questionnaire. All of them, except for one, had a game time superior to 24 hours in Skyrim, refer to fig. 5.1. Of these testers, 9 had played less than 4 playthroughs, which had been defined as completing at least one “major quest line” with different characters, or in other words, how many characters had they created that had completed at least one of the stated quests. The remaining 13 all had more than 4 playthroughs, with 7 of them having more than 11 playthroughs. As such, most of them should have a decent level of knowledge of the base game, to compare relative to the prototype, providing a fairly solid sample for the tests.

Half of the testers (11) were between 25 and 30 years old. 22.7%, 5 answers, were between 18 and 24. The remainder were 1 (one) younger than 18 and 1 (one) older than 30.

The majority of the answers were neutral to positive, in regards to players’ impressions towards randomly generated content in games, as can be seen in fig. 5.2. This indicates that this type of approach is not rejected by the gaming community.

In regards to Skyrim’s dynamic quests, the answers were mainly on the negative side, see fig. 5.3. However, the answers also seem to show that there is not a rejection, at least not total, towards such type of approach, supported by the number of neutral and slightly positive responses in fig. 5.3. The negativity towards Skyrim’s approach were mostly in regards to its implementation or execution, and the results that derived from there.

As expected, due to the nature of the prototype and its implementation, the answers to the questions on the provided quests, were similar to the ones regarding Skyrim’s dynamic quests. That is to say, they were mainly on the negative side. This can be seen and inferred by observing the answers to the questions: fig. 5.4, fig. 5.5, fig. 5.6 and fig. 5.7.
Figure 5.2: Testers’ response regarding interest in random content generation

Figure 5.3: Testers’ response regarding Skyrim’s radiant quest
Figure 5.4: Testers’ response towards the implementation of the prototype’s quests

Additionally, these reactions were mostly closer to a neutral response, indicating that these quests, both the ones from Skyrim and from the prototype, were mostly seen as uninteresting or dull. This is proved, confirmed, by the answers of fig. 5.7.

The majority of answers indicated that the aspects of most importance in regards to randomly generated content, were that results of the process had to be logical, coherent and cohesive, especially in regards to the gameplay (though it should still consider the narrative), and that they had to promote variety to the actions the players can or have to perform in the game (that is, different from the common approach in that game). For a more exact breakdown of the results, refer to fig. 5.8.

Finally, the answers regarding the usage of randomly generated content in games, fig. 5.9 and fig. 5.10, confirm that, despite its problems, there is not a rejection towards such an approach. As such, randomly generated content, can still be used as a support tool in adding and extending content, though its exact usage and implementation need to be further analysed and improved.
While doing those quests, did you get the impression you were just being given a standard, generic quest?

22 responses

Figure 5.5: Testers’ response regarding the prototype’s quests: Repetitiveness

How would you rate the quests you were asked to complete:

22 responses

Figure 5.6: Testers’ response regarding the prototype’s quests: Quality
In comparison to the "base", Radiant System generated, quests (from the non-modded Skyrim), how much do you think the quests you were asked to complete had potential, interesting parts:

22 responses

**Figure 5.7:** Testers’ comparison between Skyrim’s and the prototype’s random quests

What aspect matters most when it comes to randomly generated content. Select your top 3, in any order:

22 responses

**Figure 5.8:** Testers’ opinion on the most important aspects of random content generation
Do you think/believe that the idea or intention of having randomly generated content in any game could be good for helping provide content in a game or entirely replace it?

22 responses

Figure 5.9: Testers’ response regarding addition of randomly generated content to a game

If you answered yes to the previous question, what for, or how, should randomly generated content be used:

12 responses

Figure 5.10: Testers’ response regarding usage of randomly generated content in a game
6

Conclusion

Contents

6.1 Conclusion ................................................................. 49
6.2 Future Work ............................................................. 49
6.1 Conclusion

It can be said that a large amount of parts of the initial objectives were either not completed or largely simplified or changed, in order to make them work with Skyrim. This can be seen by taking the above sections on Problems and on Implemented Structure/Architecture, and then comparing them with the stated objectives.

Given such points, this work can hardly be considered a success. Whether it should be truly considered a failure we look to the results of user testing.

The results from the questionnaires, relative to the tests on the developed prototype, were mediocre. While not a good result, which can be attributed to a number of different points, it does show that there is not a complete rejection in regards to this type of approach in games.

Overall, the results of this project can be said to be a failure, in regards to its initially stated objectives. However, it does allow to affirm that such an approach is a possible, potential, way to further improve games and their content.

Finally, it should be noted that more work is still needed in this regard, in order to obtain better and more positive, or more conclusive, results.

6.2 Future Work

Regarding future works, relating to this one, a potential study subject would be an integration between an action selection system, akin to the one this project attempted to create, and a story generation system. In chapter chapter 2, a few examples of works related to story generation are mentioned.

One other clear route to take, would be to improve upon the system suggested and tested in this document. Additionally, it would be recommended that such a project be developed in a system different from Skyrim’s Creation Kit, due to the issues referenced thus far, as well as other potentially existing problems not found in this work.

Another suggested direction to take would be the development of a dynamic event-like system, where the AI entities existing in the world would receive an event to follow. A possible representation of events would be a group of rules and/or instructions, that the NPCs should follow, with a group of objectives for the NPCs to complete. It should also set some other flags or provide additional information that may be necessary for the NPCs, such as teams and their composition.

So, similar to traditional quests, those events could represent something like a rescue mission, a castle siege, and so on. However, unlike with the typical quests, the entities themselves should try to advance or complete an objective without the need for a player’s intervention. Whether NPCs’ are able to do so successfully, would be left entirely to the designer’s intention or objective and the system’s implementation.
This has some similarities to the Radiant AI system, and its NPCs’ packages. However, in order to advance the existing systems, there’s a need to either deepen the potential AI of existing entities or introduce a different component to handle similar kinds of interactions. Essentially, further breakdown of parts and delegation of tasks, based on the new parts.

Advancing from the previous suggestion, a direction to take could be investigating the potential development of a system that would handle dynamic constructs, in contrast to what typically are static, pre-made, constructs. For a simple, standard, example, we could have a town in a RPG that would grow or fall, or even be completely created from the very foundation, without requiring player input or having been entirely made by a developer. Furthermore, it would not have to be limited to such constructs or objects, with them potentially being, for example, the basic, generic quests described and used in this project. Applying this suggestion to chain quests, would lead to a work similar to the initial premise of this project.

Another approach, by delving further upon this last suggestion, could be to investigate making the states of these constructs, where applicable, dynamic. In other words, rather than having a, standard and limited, set of possible states that a construct can be in, which is the common approach, we would have a system that would, dynamically, alter the condition(state) of said construct. The Radiant AI system and its NPCs’ packages, can be seen as a somewhat simple(r) example of such an approach.

Finally, there is also the continued improvement of NPC related Artificial Intelligence. This is being continuously studied and improved upon, but has still to achieve a good balance between the effects achieved and the resources needed. At least, at the time of writing, it seems to be hard to reach a high enough level of intelligence, without having to compromise the processing (time) required by other parts of the system and/or severely increasing development costs.

Also, in this author’s opinion, there is a need to take a better, deeper, look at how the players are motivated to play a game, what aspects of games can be dynamically generated, and how do these two relate to one another, in order to gain the most benefit. An additional final note is that, despite this work and suggestions referring mostly to games, many of these points could potentially be applied to different types of interactive (digital) media and could also be considered in their research and development.
References


Games


A

Questionnaire
Radiant Improvement Thesis Questionnaire

A questionnaire regarding the mod developed for Skyrim, to improve the Radiant System.
The Radiant System is Bethesda Softworks' attempt to create a technically infinite number of content (Quests).
The Radiant System tries to create new quests, by using a group of generic quests as basis.

This thesis, and the developed mod as prototype, tried to take this system further by having it generate Chain Quests (group/set of sequential, usually interconnected, quests).

* Required

1. Your Age (Years) *
   Mark only one oval.
   ○ less than 18
   ○ 18 to 24
   ○ 25 to 30
   ○ 31 or more

2. Number of Skyrim Playthroughs (Complete at least one major quest line - e.g. College of Winterhold, The Companions, etc. - with a different Character): *
   Mark only one oval.
   ○ 0
   ○ 1 to 3
   ○ 4 to 7
   ○ 8 to 10
   ○ 11 or more

3. Hours of Skyrim Playtime: *
   Mark only one oval.
   ○ Less than 24 hours
   ○ 24 to 100 hours
   ○ 101 to 200 hours
   ○ 201 or more hours

4. Rate how you feel about randomly generated content in any game (be it maps, items, etc.) *
   Mark only one oval.

1 2 3 4 5

Terrible and boring ○ ○ ○ ○ ○ Great and interesting
5. What aspect matters most when it comes to randomly generated content. Select your top 3, in any order:

- Number of possible, generated, results or outcomes
- Which part is randomly generated, i.e., whether it is the map, the items, etc.
- Whether the results of the random generation make sense/ are coherent, i.e., the results are logical and cohesive
- Whether they allow or promote different perspectives or playstyles
- Its insertion in the game, in relation to the story
- Its insertion in the game, in relation to the gameplay, i.e., how it combines/works/meshes with the remaining game systems

6. (Optional) What games do you think did Randomly Generated content well and in what area (maps, items, etc.):

7. Did the quests you were requested to complete feel or seem to be hand-tailored (created entirely by someone)?

- No. They seem entirely machine made / mass produced / dull
- Yes. They seem entirely man made

8. While doing those quests, did you get the impression you were just being given a standard, generic quest?

- Yes. It seemed entirely like the typical forced quests (kill 10 rats, gather 5 herbs)
- No. It seemed to be a hand-tailored, carefully made quest

9. How would you rate the implementation of the Radiant System or the radiant quests existing in Skyrim (such as the bounty quests offered in bars):

- Terrible and boring
- Great and interesting
10. How would you rate the quests you were asked to complete: *
   Mark only one oval:
   
   1 2 3 4 5
   Terrible and boring ☐ ☐ ☐ ☐ ☐ Great and interesting

11. In comparison to the “base”, Radiant System generated, quests (from the non-modded Skyrim), how much do you think the quests you were asked to complete had potential, interesting parts: *
   Mark only one oval:
   
   1 2 3 4 5
   Worse than Skyrim's ☐ ☐ ☐ ☐ ☐ Better than Skyrim's

12. Do you think/believe that the idea or intention of having randomly generated content in any game could be good for helping provide content in a game or entirely replace it? *
   Mark only one oval:
   ☐ Yes
   ☐ No

13. If you answered yes to the previous question, what for, or how, should randomly generated content be used:
   Mark only one oval:
   ☐ To replace how content is created
   ☐ To replace how content is created except the story
   ☐ To help or extend existing hand-made content, but not replace it
   ☐ Dependent on the game's objective, approach (design) and how it is used or done
   ☐ It does not matter as long as it is well done and makes sense
   ☐ Other: __________________________

14. (Optional) Share your thoughts on Randomly Generated Content and/or the Radiant System, or make suggestions

   ____________________________________________________
   ____________________________________________________
   ____________________________________________________