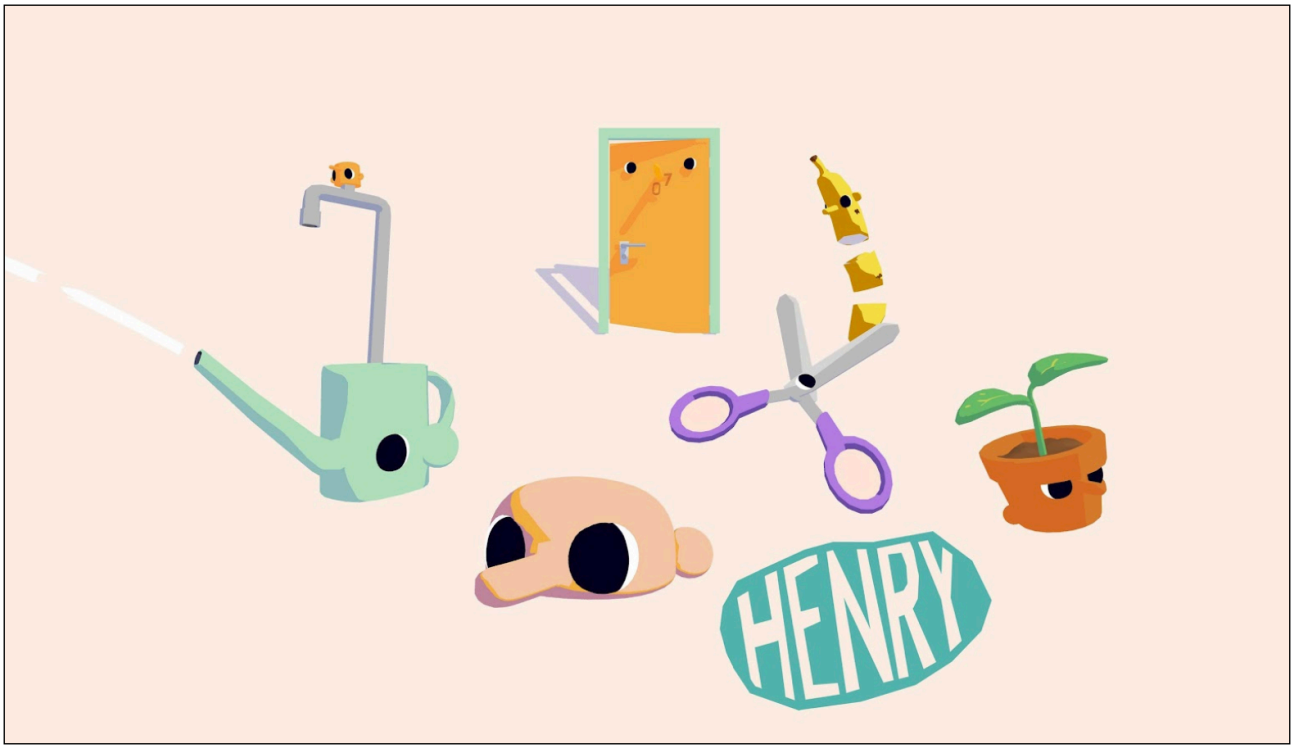
Creating an adaptive music system for the video game prototype Henry.



Acknowledgements

I would like to thank my lecturers at ZHDK. **Manuel Gerber** and **Daniel Hug** were always there and gave me the best support throughout my studies. I would also like to thank **Olav Lervik**, who has helped me with my composing during my studies in Zurich. Likewise a thanks to all the students of the master sound design course in Zurich. Thanks to Oscar Van Hoogevest for persuading me to enrol in the Master's programme in Zurich; **Simon Frankhauser** for answering questions about the study abroad; **Mischa Nüesch** for linking me with Simon; **Aleksi Villberg** for meeting and exchanging gameaudio literature; **Prabhav Bhatnagar** for helping me with the coding. I would also like to thank my advisor **Paul Weir**, who was very helpful during the process of the master's thesis and was able to give me valuable insights. Special thanks to my supervisor **Antti Ikonnen**, who supported my ideas and intentions and was always a helpful contact during my exchange. Another big thank you to **Aaron Abt** and **Tim Bürge** from Lululu entertainment. Without their game, this project would not have been possible. Thanks to all New Media students from Aalto University who supported me in any way during my time in Helsinki. I would also like to thank the entire Sound in New Media class at Aalto University. Thanks a lot for the support!

Many and sincere thanks to my family and friends!

And a very special thank you to my partner **Katherine Newton** who encouraged me to do the exchange in Helsinki.

Author Lucien Montandon

Title of thesis Creating an adaptive Music System for the video game prototype Henry

Department Department of Art and Media

Degree programme Masters's Degree Programme in New Media / Sound in New Media

Year 2022 **Number of pages** 57 **Language** English

Abstract

Henry is a single-player narrative adventure video game with occasional platforming mechanics, being developed by an indie studio called LuLuLu from Zurich. The studio's goal is to produce a playable prototype that can be shown at game conventions in order to attract publishers. To achieve this, the game should feature an adaptive music system, which should serve as a foundation for the further production. The system is supposed to improve the overall gaming experience and it has to be realisable within the given production time of three months.

The aim of this thesis is to develop and explore different types of adaptive music systems in order to gain more knowledge about their implementation techniques but also to understand, which system offers which possibilities and what kind of gameplay mechanics can be linked to the system. Another goal of this work is to distinguish between different dynamic audio terms in the context of game audio. Thus, dynamic, interactive, adaptive, generative and procedural audio are being discussed.

In the main part of the thesis, the various adaptive systems, that have been developed, are introduced. Some of them have been tested with the help of gameplay videos, while others have been implemented directly with FMOD and Unity. Each system is analysed for its future potential and its adaptability. Thereby, the complexity of the individual systems and the game mechanics they can be connected to, are estimated. The work also outlines the feedback from the developers and what kind of improvements to the system could be made.

The resulting adaptive system has, on the one hand, the ability to adapt to the player's progress and, on the other hand, it can adapt to game actions that are detached from any progress. Horizontal and vertical adaptive composing techniques are an integral part of the system and therefore contribute to the variability of the music. Among other over-complex systems, this one has prevailed because both actions are part of the core mechanics of the game. One of the main findings of the thesis is that through this close linkage to the core game mechanics, the system can adapt very well to additional levels in the future.

This thesis can serve as a useful resource if one is interested in learning about adaptive music systems and their implementation with FMOD and Unity. Also, the explanation of the different dynamic audio terminologies, in the theoretical part of the thesis, can be a help to get an overview in this topic.

Keywords gameaudio, adaptive music, sound design, FMOD, Unity, videogame

Table of content

1. Introduction

Personal Background

During my studies in Zurich and Helsinki, I worked as a sound designer and composer on several games. Creating an audio concept for a game and implementing it has always been something that has kept me busy since I started my Master in sound design at the Zurich University of the Arts. In this short overview I would like to present the games I worked on, because the experiences I made during the production of these games can be considered as a basis for my thesis. So let's take a brief look at these different games and their approach to sound design and music.

GameMakerStudio[1] In 2019, I created smaller 2D retro games with the software GameMakerStudio. I was mainly interested in gaining experience in programming and implementing sounds and music.

Fishery (2020) is a creative aquarium simulation and management game. The main goal of the game is to design, create and maintain your own aquariums. The game is in Early Access on Steam. I recorded six soundtracks for the game in collaboration with Matthias Gusset[2]. The songs are played as a linear soundtrack in the background during gameplay.

Bämeräng (2021) was my first officially released game for which I did the sound design and music. Bämeräng is a fast-paced boomerang fighting game for two to four players. The game features a soundtrack that changes when a new round starts. The sound design is based on various ancient instrument samples layered on top of each other and processed with different audio effects. The soundtrack for the game was featured on Bandcamp in the April 2021 issue of Highscores.

> *"Bämeräng sounds like a psychedelic vision of the future unearthed from 1980s rural Australia. Swiss composer Lucien Guy Montandon explores deep grooves with lots of digital woodwinds, rubbery synths, and tribal percussion, plus the occasional bass drop. This is not music that follows the latest trends or finds inspiration in retro games, it's just gleefully otherworldly and fun"(Jarman, 2021).*

I also signed a deal with a label called Blackscreenrecords[3]. Therefore the soundtrack was released on vinyl through their shop and their worldwide distribution network for vinyl game soundtracks.

Eine Laune der Natur (2021) is a narrative driven mini game which I am working on. It tells the story about the grieving process after a loss through suicide. For this game I created and implemented all of the content myself. It also features a simple adaptive music system.

Marble Sadnes (2021) was developed as part of the game project course at Aalto University in collaboration with Henry Lämsa and Patrick Boman. The aim of the game was to create a clone of an existing game. We chose Marble Madness (1984). The video game contains adaptive music and a soundscape with unique dark and sad sound design concept.

In Pieces (2021) was the result of another collaboration during the game project course at Aalto.
The main quest features a broken in game cd player, which the player has to fix for completing the main quest.

---

[1] **GameMakerStudio** is a cross-platform game engine developed by YoYo Games.

[2] https://open.spotify.com/artist/28zL2v3cJ1dM0CW8XSF1dg

[3] https://blackscreenrecords.com

Aims & Motivation

Looking back to these games now, makes me realise my progress in creating sound design and music for video games. Started with a classic linear approach in fishery compared to marble madness, displays a big difference in the skillsets that I have learned during my studies. I now would create something completely different for the fishery game, although I like the tracks I recorded back then. Another factor I would like to mention is that there is still a lot for me to learn, which is one of the reasons why I am writing this paper.

Before I started working for video games, I mainly used to work in the context of linear media. Working for video games now presents me with new challenges. For me, the medium combines many aspects of artistic creation, this is the reason why I have always been fascinated by it. Another motivating reason for this thesis is that I want to continue developing in the field of game audio as a sound designer, composer and audio programmer. During the last two years I have learned to implement audio assets myself. One of the goals of this work is to develop these skills. This includes working with the middleware software, but also creating scripts in the game engine itself. I think this is a powerful skill that can make a big difference when developing a video game as a sound designer and composer.

Over the past few years, I have learned the role and responsibilities of a freelancer in the industry. With this project, I want to be able to move even further in that direction. The back and forth between my own ideas and the vision of the studio is a big and important part of working together in game audio as a freelancer.

The precise examination of different terminologies also plays a central role in this work. By clearly distinguishing certain terms, I want to become more aware of my own possibilities and also expand and professionalise my vocabulary.

The main aim of this work is to develop and explore different types of adaptive music systems in order to gain more knowledge about their implementation methods in terms of its architecture, but also in terms of the music that the system contains. Another goal of this work is therefore to understand the advantages and weaknesses of each system in order to find the best possible solution for the game prototype.

Scope & Structure

In November 2021, Tim and Aaron from Lululu Entertainment reached out to me and asked if I would like to collaborate on the development for their prototype game called Henry. As I had already worked with Tim and Aaron on the game Bämeräng back in 2020, I was pleased that they asked me again. The goal of the pre-production was to develop a fully functional first level of the game by the end of April 2022. The prototype would serve as the basis for the game's mechanics and would be shown at showcases to possibly attract a publisher in the future. During the whole process from the end of 2021 until the end of April 2022, I was therefore in constant communication with the developers.

The overall scope of this thesis contains the process of creating an adaptive music system for the game prototype. I was also responsible for all the music and the audio assets (interaction sounds and ambience), but the focus of this thesis is on the development of the adaptive system.

After the introduction I will familiarise the reader with the game Henry, its overall gameplay and its core gameplay mechanics. This part is intended to make the most important mechanics of the game understandable and to provide the necessary knowledge about the game.

Before focusing on the practical content, the paper examines and discusses various game theory concepts that relate to dynamic music. In the chapter called “Tools” I describe the software and equipment I used during the project. In the following chapter, I will explain different gameaudio terminologies.

In the main part of my thesis, I will reveal the production process of the music system for the game. Each system will be carefully described and analysed with its complexity and benefits. The exact steps in the implementation, including the work in the middleware and scripting in unity, will be explained. New problems and hurdles of the system will be revealed and discussed. Furthermore, the musical material in the respective systems will be indicated.

In the chapter called “Ambience” I explain the creation of the different ambiences and their implementation. In addition, I go into the underlying system, which uses the same basic principles as the music system. In the last chapter, I turn to the overall conclusion. This chapter also describes my own experiences and gives an outlook on what is to come in the next few months.

Research Questions

As I already mentioned above, the game should feature an adaptive music system, which should serve as a foundation for the further production. The system is supposed to improve the overall gaming experience and it has to be realisable within the given production time of three months. This leads to the following main question for the thesis:

- What kind of adaptive music system can be implemented, that serves as a foundation for the further production?

To address this question, the thesis focuses on several smaller topics. In this way, each system is examined in terms of its complexity and its ability to be implemented in new levels is tested. An important aspect is to determine which gameplay elements can be linked to the system and what the benefits of each system will be. This also leads to the point of asking what improvements could be made and why.

2. Henry Game Prototype

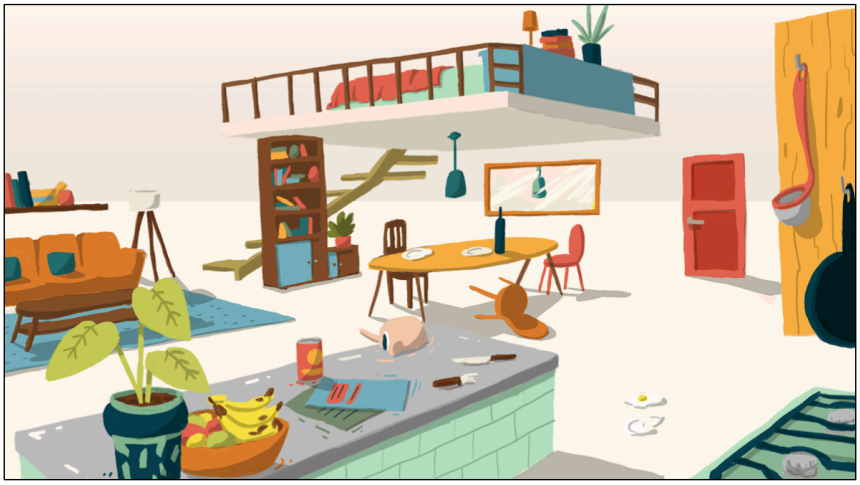


Figure 01 | MoodSketch made by Tim Bürge

Genre: Narrative Adventure
Mode: Single player
Platform: Nintendo Switch, Windows, Mac OS X, Engine: Unity
Controls: Mouse & Keyboard / Controller, Playtime: ca. 2h
Release: Q4 2023 / Q1 2024

The following text is written by the developers themselves. As this gives us the best idea of what the game is all about, I asked them if I could use some of the description for my thesis.

Henry is a funny little character with only half a head as his body, but with the peculiar and powerful ability to possess and control any object within his reach. Using his ability, the player embodies the everyday objects in Henry's life and experiences his heartwarming life story, reaching from child- through adulthood to the end of his days. Players can explore how different combinations of objects interact, discover every object's special ability and progress through a variety of levels, each providing a unique situation of Henry's life.

Gameplay

The gameplay in Henry revolves around the objects in Henry's everyday life. When players use Henry's ability to take control of an object, each of them moves differently, has a unique set of properties and special abilities. By switching between these avatars and combining their unique aspects in different interactions, players learn how to combine them to unlock new objects and areas and to progress in the game. Henry is played from a third-person perspective and contains some limited 3D platforming elements.
The focus and joy of Henry is in discovering the story about Henry's life through the objects he encounters. The story guides the player through various scenarios and gives context to the tasks and interactions between objects which the player can explore in their own unique way.

Story beats are built of series of interaction chains (e.g.: finding and cooking an egg, or watering a plant to make it grow and bloom). Since there is always a slight overlap in properties between objects, players can solve interactions in multiple ways. Filled with a lot of humour and a mixture of surprise, chained interactions create unique and memorable moments. Completing these tasks will reward the player with

longer bits of narration, new locations or additional objects. The objects in Henry's life are the heart and core of the gameplay. Through them the player explores the game world. The object gameplay can be grouped into the following parts: Controlling / Possession, Movement, Abilities, Systemic Properties[4]

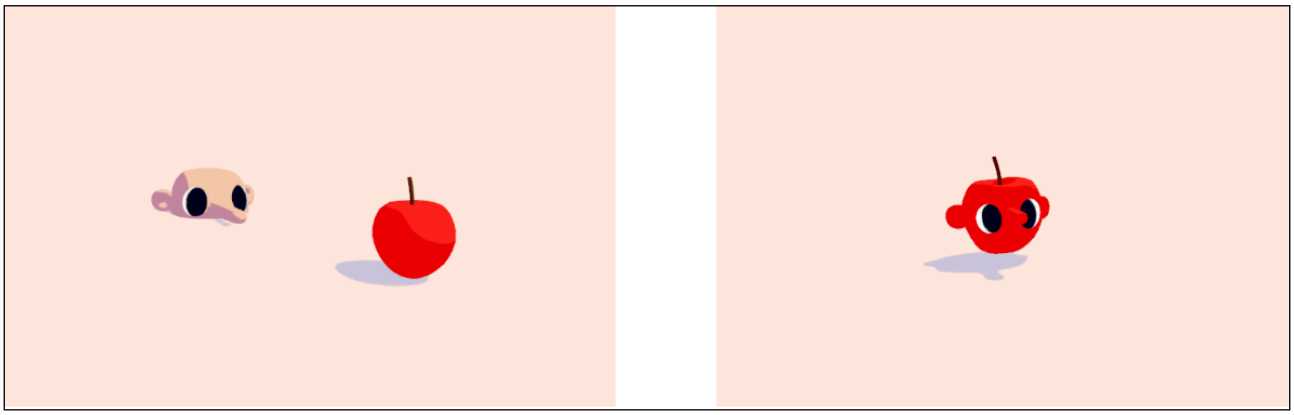


Figure 02 | Possessing gameplay mechanic

Controlling & Possession

Controlling / Possession is Henry's main ability, it allows the player to take control of objects and perform actions as / through them. When the player makes Henry possess an object, Henry's body becomes the object. Visually, this is communicated by the object gaining Henry's facial features. Taking control of an object not only serves as a vehicle of in-game progress by completing tasks with them, it is also a big part of the narrative theme. Each object serves its purpose in the context of the story and aims to give the player a unique feel and experience.

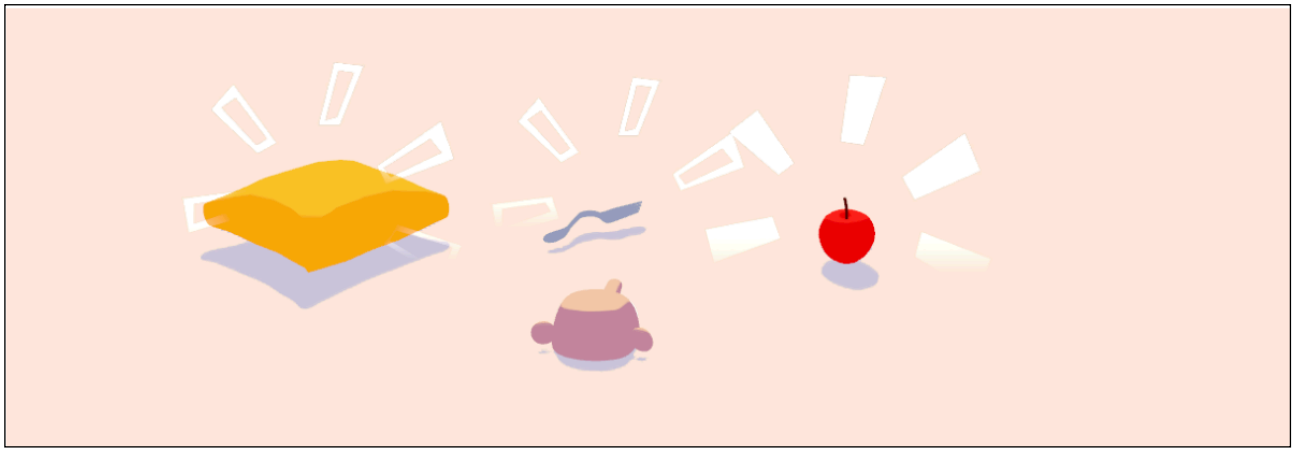


Figure 03 | Henry has to decide what kind of Object he wants to possess

[4] Since sound design and music are closely linked to gameplay mechanics, I will often refer to this keywords in the upcoming chapters

Movement

An object's movement capabilities are an important part of its character as well as an important device through which we can design levels and progression. Many objects have different movement / character controllers and with that also different control schemes. From trudgingly walking to jumping, rolling, bouncing or floating; all can serve a specific purpose in an interaction chain or in the feel of an object. Movement makes up a big part of an object's identity, but they may also pose limits to how objects can be combined, depending on how they are placed in the world. An object without a jump ability will have a hard time reaching high places, for example. By using this in our design of the rooms, we can constrain the possible options to something that does not feel too easy or too overwhelming.

Abilities

Many objects have unique abilities, which can be triggered by the player when they have taken control of it with Henry. Abilities can serve different purposes depending on the object. They can range from transforming, moving or generating objects, to changing the systemic properties of an object, to simply fulfilling a task in the story or just adding a little stylistic flavour to an object.

Systemic Properties

In Henry, many controllable objects have a limited but unique combination of systemic properties that can interact with each other. For example, objects containing water are conductive to electricity; an object being heated can burst into flames or be cooked; a plant being watered grows; objects dropped on something bouncy are flung really high; and so on. Systemic properties are a big part of how a player may experience their own unique playthrough, since in some occasions multiple objects can have a specific systemic property and the player may choose which one to use for a task. Furthermore, they also serve as an outlet for the player's creativity, since through clever combining and chaining of properties, tasks may be completed in unusual ways.

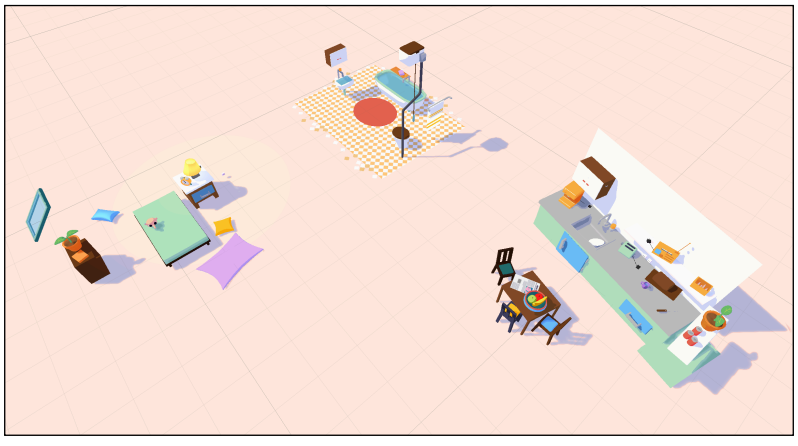


Figure 04 | Level 1 prototype

3. Tools

Integrating music and sound design in a game is a multi layered process, where I have to hop from one software to another. For this project I have been using different softwares and devices for creating and implementing the assets. For the purpose of transparency, I will briefly explain each software and device as follows.

Version Control
GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere. Because I work closely with the developers and always have to be up to date with the newest version of the game, this software is necessary. It allows me to work on so called branches, where I can implement my work and later I push my ideas to the developers branch.

Bitwig (Digital Audio Workstation)
A digital audio workstation (DAW) is an electronic device or computer software application used for recording, editing and producing audio files. For this project I am working with Bitwig. Bitwig Studio is a digital audio workstation that was developed by Bitwig GmbH. It is available for Linux, macOS, and Windows. The main purpose of Bitwig is to be used as an instrument for live performances. However, it can also be used for composing, recording, arranging, mixing, and mastering.

Audio Editor
Ocenaudio is a cross-platform, easy-to-use, fast and functional audio editor. It is the ideal software for people who need to perform basic editing operations, such as trimming, cutting or copying audio files.

Middleware Software
A middleware software is a computer program that provides services to other computer programs. Middleware is often used to provide a common interface between different computer systems. There are two common middleware Software when it comes to audio implementation in video games. One is called FMOD and the other is called Wwise. Since the developers are already familiar with FMOD, and they also used FMOD in their last game, it is obvious that we will work with the same software again for the prototype of the game. Especially with a prototype, things have to move quickly.

Field Recorder
A field recorder is a type of audio recorder that is designed to be used in a wide range of recording environments. Field recorders are typically small, portable, and rugged. All field recordings were made with a Roland R-26.

Game Engine
A game engine is a software development environment designed for people to build video games. It provides a framework for game developers to work within, allowing them to create games more quickly and easily. The most popular game engines are Unity3D and Unreal Engine. These engines allow developers to create games for a variety of platforms, including PC, consoles, mobile devices, and virtual reality headsets. The prototype Henry is built in Unity

Visual Studio
Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. It can produce both native code and managed code. I used visual studio to create scripts in unity for the implementation of certain sounds.

4. Interactive, Adaptive, Generative, Procedural

Audio plays an important role in most video games these days. It can be used to create an immersive experience, to help communicate important game information, or to add excitement and suspense to a specific moment while playing the game. The most common and simplest way to integrate audio into a game is in linear form. All you need is a start event and a stop event in the code at the point where you want the music to play or to stop.

> *"With linear composed music, the music begins playing through a musical piece when the associated level is loaded. If the music reaches the end of the piece, the music loops. When a new level is loaded, the music either abruptly changes or quickly fades out, and is replaced with the new level's associated music" (Plut & Pasquier, 2019, p. 1).*

Among several other games, a good example of linear music is the remaster of Tony Hawk's Pro Skater 1 + 2, which was released in 2020. The soundtrack consists of different songs that starts when a new run is started and ends when the two-minute run is over.

However, with the interactive possibilities of video games, new ways of implementing audio have evolved over the years. These concepts and techniques extend the idea of linear music and therefore different techniques and terms have evolved in the field of game audio. In this chapter I will explain and distinguish the most important terms related to these techniques.

I will intentionally speak only of audio. By audio I am referring to sound design (sound effects, ambience) and music. I am aware that one could also make a distinction between music and sound effects or ambience. At the moment, it focuses primarily on the possible applications and techniques, so I will not make a clear distinction between music and sound effects or ambience. For this reason I will use the terms music and audio interchangably. For simplicity, I will sometimes refer to an invented gameplay example of the game Henry.

First it is important to know that these terms are not set in stone and are often used in different ways, as the following examples will show.

> *"The music interacts with the variables introduced by players and adjusts itself accordingly. This type of music may be called "interactive," "dynamic," or "adaptive," with some subtle shades of differences to the terms" (Phillips, 2014, p. 168).*

> *"In the video game industry, interactive music is sometimes referred to as adaptive music. In this book we have chosen to use these terms interchangeably, even though there are subtle differences between the two. "Interactive music" is used more often when a player has direct control over the music, as in musical games like Guitar Hero (2005). "Adaptive music" is used when the player has indirect control over the music. Adaptive music may use other factors in the game world itself to change the music dynamically, including the time of day, whether, number of enemies, or player health" (Sweet, 2014, P. 36).*

> *"Adaptive music is sometimes called "interactive music", and is music that reacts to a game's state" (Plut & Pasquier, 2019, P. 2).*

> "many games today utilize adaptive-interactive audio, that is, each player constructs her own unique soundscape by moving and interacting with their avatar (Droumeva & Fraser, 2011, p.135)

> "Dynamic audio is audio that is designed to change, either in response to the user, or in response to changes in the gameplay environment. Dynamic audio encompasses both of what we call interactive audio and adaptive audio" (Horowitz & Looney, 2014, P. 76).

> "Earlier, I defined dynamic audio as audio that is changeable, a broad concept that encompasses both interactive and adaptive audio. It is audio that reacts both to changes in the gameplay environment and/or in response to the player." Karen Collins (2008, p. 139)

As some of these examples clearly show, the terms interactive, dynamic or adaptive audio are often used in similar contexts when talking about game audio, and it seems that there are no strict rules for the terminology of these definitions. Nevertheless, there is a slight tendency to use dynamic audio as a comprehensive term. I consider it useful to have an encompassing term that includes both adaptive audio and interactive audio. For this reason, I will refer to dynamic audio as a generic term for both interactive and adaptive audio.

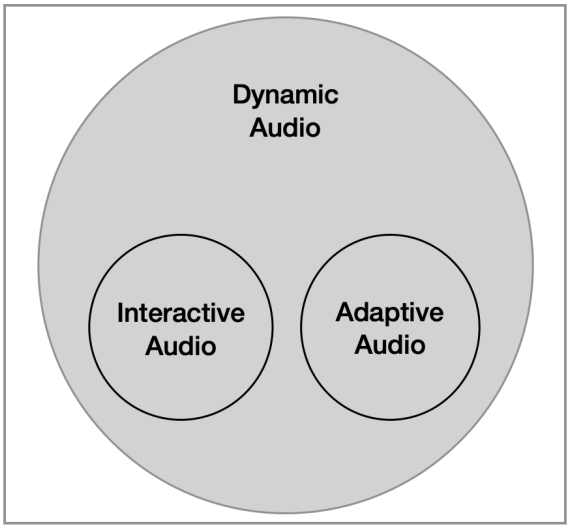


Figure 05 | Dynamic Audio

Interactive Audio

When playing a video game, the players interact with the virtual world and thus also with the sounds that are triggered by the various actions. For example, when the player jumps or opens a door in the game, the corresponding sound is played exactly when this action takes place. So one could consider that the player interacts with his activities and the sounds that go with them, and therefore audio in video games could in principle be called interactive. Since this cannot be considered fundamentally wrong, it is not particularly precise, if we want to analyse game audio more accurately. To understand this, let's imagine a sports game that plays licensed linear pop songs during the menu or during the gameplay, like the Tony Hawk game, for example. The player starts a new run, the song is played. After one minute, the player decides to stop the run, the music stops, the new run starts and the new song is getting started.

Do we call these interactions with the music in the game interactive audio? In the first example, one could argue yes here (because the player decides when to pause or start the game), which is not entirely wrong. Others would say it's just the soundtrack and background music of the game, which is acceptable. This example shows that the topic is more complex. It is therefore logical that with this complexity, different terminologies must also be taken into account.

So what does interactive audio mean when we look at it more closely in relation to game audio? Let's make a concrete example with help of our Henry game.

> In the kitchen Henry can find a little radio. Let's imagine, that he needs to change the frequency for receiving another radio program, which then activates a certain task, that Henry has to solve next.

In this example the player interacts directly with the audio, that he or she is perceiving. This occurs also in a more abstract form when we think of sounds, that are connected with the direct input from the player.

> *Karen Collins points out that, "In Super Mario Bros., for instance, an interactive sound is the sound Mario makes when a button has been pushed by the player signalling him to jump" (Collins, 2008, p. 4).*

In our Tony Hawk example, this would be the sound when you do an ollie.

> *"Interactive audio is a sound event that occurs in response to the player directly. In other words, if a player presses a button on a controller, the character on screen might move in some way, and that event creates a particular sound—it might be a sword swish, or a footstep—it doesn't really matter. If the player presses the button again, the sound will re-occur. The sound involved is an example of a simple interactive sound effect"(Horowitz & Looney, 2014, p. 77).*

Another example of interactive audio that seems a little more complex, is the ocarina flute in The Legend of Zelda: Ocarina of time (1998). During the course of the game, the player learns different songs from different characters. The player then can interact with the flute by pressing the right button combination which ends in a melodic phrase. These songs, when being played correctly, are connected to a game mechanic. Sun's Song for example is useful for freezing the undead and turning day into night or vice versa. What makes this example different from the more basic Ollie Jump sound is, that the button pressing and resulting sounds are in addition deeper linked to a gameplay mechanic.

Maybe the most obvious form of interactive audio are interactive music games, such as music-based games like the rhythm-action genre. For example the remaster of Rez (2017). As these games are mainly focusing on the musical interaction, it would make sense to distinguish the term interactive even a bit more. Stevens & Raybould (2011) are referring to an additional interactive category, when it comes to music based games. In these systems the player has direct control over the audio state itself. These systems are called performative system, because they are allowing the player to perform, while interacting with the audio.

> "For truly interactive music, there needs to be a feedback cycle between the game engine and the music state. Rather than simply receiving input from the game and responding accordingly, the game state should receive feedback from the music as to its current position and the time to the next musical transition point" (Stevens & Raybould, 2011, p. 238).

It makes therefore sense that this sort of music games can be framed as performative systems, because they intend to be used as a performing tool. Knowing this, I would quickly return to our example with the radio in the kitchen. In our example Henry can change the music from the radio, but he cannot change the patterns of the music or the content of it, neither does the musical content somehow reacts to his input. The game itself doesn't get any information about the music (except which program that is being played, but that I consider not as the content of the music, like the tempo or a transition for example). In the view of Stevens & Raybould our example is not a performative system. But I would argue that a player could actually instrumentalise the radio to create a performance. Imagine if he or she changes the frequencies in a specific rhythm, while also jumping. A self created rhythmic pattern would emerge. Our system might not be necessarily a performative system by its nature, but I can be misused as a performative tool.

With this example I want to clarify that the term "performative system" describes the architecture of the system itself. For me, it is a specific terminology for a system that includes interactive audio, but can also include other forms of dynamic audio.

Whitemore (2014) describes another layer of interactive audio in video games:

> "Interactive music occurs when a player is making gameplay decisions based on what he/she consciously hears in the music, and thus is reacting directly to how the music is behaving" (p. 120).

If we want to illustrate this explanation with a specific gameplay example, we could imagine the following situation.

> Imagine a puzzle in Henry, where you're exploring an ancient cave. For opening a secret door with a treasure in it, you have to put four similar looking spheres in the right order in front of the door. Each sphere plays a specific tone if you possess it. The door only opens, when you rearrange the spheres from low to high pitch.

> "Interactive music occurs when a player is making gameplay decisions based on what he/ s h e consciously hears in the music, and thus is reacting directly to how the music is behaving. But with adaptive music, the interaction is between the player and other aspects of the game design such as visuals and game mechanics" (Whitemore, 2014, p. 102).

All these examples show that the concept of interactive audio in relation to gameaudio can be viewed from different perspectives. In my thesis I will refer to interactive audio when the player directly changes the audio with his or her input. In this thesis I will explicitly use the term interactive audio only and not differentiate between performative or non performative systems.

Adaptive Audio

The main difference between interactive audio and adaptive audio is the cause of change in the audio itself. While interactive audio directly is intertwined with the player's inputs, adaptive audio is connected to game states or events, which are being affected through the player, or through the game world itself. So the player's action has sort of an indirect relation ship to an adaptive audio system, although it appears that the player's behaviour directly changes the music. What changes the music is the event in the game, that can be triggered trough the player decisions or an other event that occurs in the game.

> *"With adaptive music, the interaction is between the player and other aspects of the game design such as visuals and game mechanics" (Whitemore, 2014, p. 102).*

> *"Adapted audio is created in the game environment in response to what is happening in the game, rather than directly to the user. Adaptive audio changes as the game changes" (Horowitz & Looney, 2014, p. 77).*

> *Adaptive audio occurs in the game environment in reaction to gameplay rather than to the user directly. Adaptive audio changes as the game changes (Horowitz & Looney, 2014, p. 77).*

> *"Adaptability is reflected in the ability of the music to react to the game-play. this is done by sending events from the game engine to the music engine where it is decided when to respond to the events. theoretically the response could occur at any point in the music, but normally the response is set to occur immediately, at the next beat, at the next bar, or at the next marker if the system offers possibilities for creating markers in the music. This is what I call adaptive music" (Kaee 2008, p. 84).*

Collins (2008) describes adaptive audio, as sound that is reacting to the game states. It responds to different in-game parameters such as time-ins, time-outs, player health, enemy health. As example she refers to the game Super Mario Bros (1983), where the music's tempo increases, before time is running out. Another common example of adaptive music is combat music that turns on when the player enters a fight and fades out again, or changes to a less intense music when the fight is over. In Skyrim (2011), staccato like strings and timpani enter, as soon as an enemy targets the player.

It is important to mention here that music as well as ambience or sound effects can be adaptive. Unlike a linear medium, the audio mix in a game must be constantly adaptable to its gameplay.
Sinclair (2020) describes the essential skills of adaptive mixing as using code for automating parameters, setting up mixer groups and using so called snapshots to create a mix that adapts to the gameplay and the environment.

> *"The soundtrack of games and movies can be broken down in terms of its three most important structural components: music, dialog and sound effects. Each serves a different purpose, and the mix is, or should, ultimately be dominated by one of these three elements at any given point based on the desired perspective and emotional impact" (Sinclair, 2020, p. 253).*

To achieve such a flexibility the mix has to adapt during the gameplay. Such adaptive systems can range from simple to multi-layered nested systems. More complex systems are able to adapt to multiple gameplay situations and are influenced by various game states and parameters.

> *"Current adaptive scores can tie themselves to almost any gameplay state you can think of, from player health to the number of times the player has been to a specific location" (Horowitz & Looney, 2014, p. 101).*

What kind of gameplay situation can we think of, when we are examine adaptive audio?

> Henry is in his bedroom. While moving around in the bedroom an upbeat groove is being played. You can hear the birds outside chirping. Henry now possess the clock, he changes the in game time from daytime to nighttime. When this event occurs, the music transforms into a more gentle and soft ambience track without any pulse and the birds were being replaced by crickets, that you can hear from the distance.

In this example the music is adapting to the gameclock event in the game and the moment when the world state is changing from day to night. Collins (2008) points out that adaptive audio also response to the player and his or her actions. An example for that would be if the music starts do decrease when the player's health is dropping to almost zero.

With the techniques of adaptive music transitions from one part to another can be easily created with so called vertical or horizontal implementing techniques. Vertical techniques refer to an approach in which different audio groups such as bass, drums and keys are layered, often mixed as something that is known as stems. These stems are getting stacked in the middleware. With the possibility to control each stem's volume with a parameter, the composer can now fade in and fade out the corresponding stems. This can also be applied to ambience or sound design events in the game. It doesn't have to influence the music. In a short documentary about Rockstar's Red Dead Redemption (2011), this techniques, of grouping stems together and using them in an adaptive system, are explained on one of their official YouTube channels. With such a vertical approach you can easily play audio in sync with each other and let different stems fade in and fade out at the same time. All connected to game events and controlled by parameters. Following example will introduce the idea of vertical and horizontal techniques in adaptive audio:

> Henry is in his bedroom. While moving around in the bedroom an upbeat groove is being played. You can only hear the drums playing. Henry now moves to the bathroom. This event raises the volume of the bass stem in our arrangement. You now can hear bass and drums playing in sync. If Henry goes back to the bedroom, the bass track decreases its volume, if Henry moves towards the kitchen instead, the keys stem gets added.

> Henry is in his bedroom. An upbeat song is being played. When entering the bathroom the song changes immediately to a soft and more ambience like track. When going back to the bedroom, the upbeat songs is being played again.

That being said, a horizontal concept allows us to jump between different sections of audio content in our middleware software. It is called horizontal, because the playhead changes in a horizontal way for accessing a new section. It's basically identical to an arrangement in a recording software, where you can attach so called play markers and jump straight to them with a mapped key. Vertical and horizontal approaches can also be combined with each other to build even more complex systems. It is essential to note that you can achieve similar things with both concepts, so there is no wrong or right way to use them.

In this thesis both of the concepts will play a significant part of some of the systems for the game.

Generative Audio

To begin with I would like to address that generative or procedural audio (that I will cover in the next section), has nothing to do with interactive, adaptive, or dynamic audio. However, that does not mean, that generative or procedural audio can't be embedded in a dynamic audio system.

Kaee (2008) distinguishes that dynamic music can be divided into two categories from a technical point of view. Specifically, variability and adpadtivity. By variability, he refers primarily to the techniques of generative music. When people think of generative music today, it is often Brian Eno who is referred to. In the 1970s, he experimented with various randomness and probability methods. This is how the often quoted ambient album "Music for Airports" (1978) came into being, which he realised with various tape loops of different lengths.

> "The album was essentially a continuation of Eno's experimentation with the tape machine as a compositional tool, as well as his exploration of generative music, music created by systems" (Carr, 2019).

Eno was mostly interested in composing music that could be played in public places without being disruptive. The music should blend with the sounds in the public space and leave enough room for conversations. It should also be possible to interrupt the music, e.g. by announcements, without suffering from it. (*nathanidiothend, 2007)*. In the 1990s, Eno coined the term generative music when he became interested in using computer software to produce music that was constantly changing and based on probabilistic algorithms.

> "… it was not until the 1990s that Eno began to explore a process whereby computer software participated directly in the creative process, producing compositions that constantly shifted and evolved using mathematical algorithms based on probability and chance" (Winifried, 2017, 189).

However, composing music under the effect of chance and probability is not a method that was first invented in the 20th century. Even earlier, artists such as Mozart, composed music according to the principles of probability and chance.

People used to play composition dice games at private parties. One of these composition games was Mozart's "Musikalisches Würfelspiel" from 1787. The players could generate a 16-bar minuet with the help of a dice. Karlheinz Stockhausen also experimented with random techniques. Klavierstück XI (1956) is considered one of the first classical examples of the so-called mobile form. The score of Klavierstück XI consists of only one sheet of music with nineteen separate sections, each of which is played randomly by the player. There are tempo and dynamic indications for each individual section. The whole piece is finished when one of the sections has been repeated twice (Kaee, 2008).

Another and perhaps well-known example of generative music is wind chimes. The wind modulates the chimes in an infinitely random order. This creates new patterns and sequences of sounds that never repeat and are constantly changing (Winifried, 2017).

The concept of wind chimes can also be seen as a kind of primal sound installation. It is a generative system based on physicality. It is worth mentioning here that the composer builds an intelligent system to realise his idea of generative music.

> *"Generative music systems are only as good as the composers who program the algorithms and rule sets that create the score. As composers are able to communicate and program their ideas more effectively to computers, better scores will be generated by these systems" (Sweet, 2014, p. 320).*

Today there are different forms and ways of producing such generative music systems. Be it with the older methods like tape loops or with a modular digital environment like for example Bitwig Grid[5] or with programming environments such as Pure Data[6] and Max[7], or simply with analogue modular synthesizers and some sequencers.

A good example of generative music can be heard on the website GenerativeFm[8]. The algorithmic systems play non-stop Generative Music that never repeats itself and always changes. Therefore, the music is very pleasant to listen to, especially if you want to relax a bit, for example.

> "Generative music offers an opportunity to experience the joy of listening to music that will never be heard quite the same way again, combined with the convenience and portability of the system which generates it. Software-based generative music systems provide the ultimate experience: unique, original, endless music played from a computer or smartphone — the exact same devices used to listen to recorded music today" (Bainter, 2019).

In summary, one can say that the main characteristic of generative music is variability and the associated fact that the music is constantly changing and does not repeat itself. One advantage of this variability is that the music does not become tiring so quickly and can be enjoyed over a longer period of time. Especially in video games, you can make advantage of these benefits. Since people often play for longer periods of time, it makes sense to apply the concept of generative music in video games.

> *"Generative music in video games is composed of a collection of preexisting musical components such as melodic phrases, rhythms, and patterns whose content can be influenced by factors based on pure chance, the state of gameplay at any given time, and mathematical rules of probability" (Winifried, 2017, p. 188).*

> *"segments of musical material are constantly chosen at random and put together to form a piece of music. instead of a practising musician, the computer is now the interpreter. It is here a task for the music engine in the game to choose between a set of segments at random to make variation. The elements chosen by the computer could be anything from the velocity of a single note to the order of long precomposed musical pieces, and theoretically the technology behind it could be anything from simple random number generation to complex artificial intelligence. Most often the music system will at least offer some kind of weighted probability. In any case the music is in some way generated by the computer itself and the concept is therefore what we would call generative music" (Kaee, 2008, p. 84).*

Winifred (2014) also explains that generative music in video games is made up of a bunch of different musical parts like melodies, rhythms, and patterns. The content of these parts can be determined by things like chance, what's happening in the game at the time, and mathematical probability.

> *"Variability, or a kind of random sequencing, is used in games to increase the lifespan of a piece of music, which may need to be repeated hundreds of times for the player. By variably sequencing the music's component parts, the music becomes less repetitive" (Collins, 2008, p. 8).*

> *"Generative music offers the promise of certain specific advantages to video game development teams and publishers. If music is able to continuously generate new variations of itself by virtue of a mathematical algorithm combined with the element of chance, then it should be able to play for long periods while avoiding repetition fatigue. Also, the generative music system will theoretically produce vast quantities of music without the need for a music budget capable of supporting such a large amount of original content" (Winfried, 2017, P.189).*

---

[5] **The Grid** is a modular sound-design environment that powers a family of devices in Bitwig Studio

[6] **Pure Data** is an open source visual programming environment

[7] **Max** is a visual programming language for music and multimedia

[8] https://generative.fm

Wwise and FMOD offer both the possibility of enriching the music with certain random factors and probability triggers. Even though it would be possible to produce Generative Music in these middlewares environments that contains only a small percentage of composer-written music, a mixture of composed music and generative techniques is often used when it comes to gameaudio.

> *"The generative way of composing is as much an indexing of musical parameters as it is the actual writing of music. To keep some artistic control in the hands of the composer, a combination of generative music and composer-written music could be used" (Collins, 2008, p. 100).*

Since the music in games is often triggered by the player's actions and therefore the output is not always the same, the fundamental question arises whether dynamic video game music is always generative music.

> *"… in generative music a video game itself may assume the role of the musical performer and trigger predetermined musical phrases in unpredictable ways, often dictated by the state of gameplay. The resulting musical construct can be considered aleatoric because the element of chance was built into the composition as a fundamental principle of its design" (Phillips, 2014, p. 32).*

I don't think this view makes much sense, otherwise our tony hawk example of starting and stopping songs during a run would be generative music. I will explain in more detail how I use the term generative music in this thesis at the end of the next section. I am aware that in this chapter I have mainly spoken of generative music and have excluded the term audio. I will also come back to this at the end of the next section.

Procedural Audio

In the middleware FMOD there is an instrument called AudioWind[9]. As the name suggests, this instrument is capable to generate different types of wind sounds by means of synthesis[10]. The different frequencies and noise types can be controlled at the game engine's runtime[11] with the help of parameters. This means that the resulting wind sounds are being created in real time when playing the game. This process does not require any pre-recorded audio samples and is known as procedural audio. Thereby, sound assets are created in real time.

Sinclair (2020) is referring to this, when he points out, that audio models can accurately recreate a specific sound through scripting or programming. He also compares these techniques with procedurally generated terrains in game design:

> *"The term procedural assets, very specifically, refers to assets that are generated at runtime, based on models or algorithms whose parameters can be modified by data sent by the game engine in real time. Procedural asset generation is nothing new in gaming; for some time now textures, skies and even entire levels have been generated procedurally, yet audio applications to this technology have been very limited" (Sinclair, 2020, p. 239).*

> *"Procedural audio describes a broad array of systems that are able to produce a range of sound outputs for a range of inputs. This includes synthetic, generative, algorithmic, and Al driven systems, among others" (Hug, 2011, p. 388).*

In a way, you could also say that the earlier game consoles, which contained a sound chip, also processed procedural audio. Because they generated the sound effects and the music in real time and did not have the possibility to play back samples due to memory shortage.

> *"Early game consoles and personal computers had synthesiser chips that produced sound effects and music in real time, but once sample technology matured it quickly took over because of its perceived realism. Thus synthesised sound was relegated to the scrapheap" (Farnell, 2010, p. 318).*

More recent games such as Just Cause 4 (2020) also make use of procedural audio. The wind instrument mentioned above is used to create "whoosh" effects from passing cars.

> *"When driving a vehicle throughout the world in Just Cause 4, there's an extra layer of sound that comes in when passing a non-player vehicle in traffic. This creates a 'whoosh-by' effect that simulates the wind between two vehicles moving at a high speed in passing traffic. These layers of sounds are generated by a noise synth written for the audio middleware, FMOD.*
> *Using a combination of brown noise and white noise, the output of the synthesizer is controlled by a number of variables on the vehicle. The envelope of the sound, noise filtering, gain settings, and mix between noise types are controlled by factors like player vehicle speed, distance to non-player vehicles, and non-player vehicle speed" (Zender, 2021).*

While in just cause only a minimal part of the audio assets are generated in real time, there are other games that work one hundred percent with procedural audio technology. A game that relies only on procedural audio is Fract OSC (2014). The player controls his avatar in a first-person perspective in a Tron-like cyber-data world, while all interactions trigger different synthesised sounds. An evolving soundtrack is being created in realtime without any usage of audio samples. Such procedural audio techniques are especially possible when Pure Data (Pd) software is built into unity.

---

[9] https://lesound.io/product/audiowind-fmod/

[10] The electronic production of sound where no acoustic source is used.

[11] **Runtime** is the period of time when a program is running

> *"The generative audio capabilities of Pd and the adaptability of Unity combine to form the procedural interface of Fract as a whole—a game in which audiovisual content, as well as technical infrastructure, is defined by a real-time, process-oriented dialogue between player and system" (Errico, 2022, p. 206).*

> *"While procedural sound generation in games is often used to create musical variety or avoid bogging down the game's processor, the sounds produced as a result of player input in Fract are synthesized in real time by the game's sound engine" (Errico, 2022, p. 209).*

The concept of Fract OSC is built entirely on procedural audio, which has facilitated and determined the implementation and game design process. No Man's Sky (2016) is another video game where most of the graphics and 3D assets were procedurally rendered. Procedural audio is here only used to generate the sounds of the various species on the alien planets (which are also procedurally generated). In this example, the procedural audio system is closely deep linked to the data of the species. Various factors of the species, such as appearance and size, influence the sound. Everything is created in real time without any use of audio samples.

> *"Very little of the audio is procedurally created, only the creature vocals and background fauna. At the moment it's too expensive and risky to widely use this approach, although there are several tools in development that may help with this. Procedural audio is just one more option amongst more traditional approaches and the best approach as always is to use whatever combination best works for a particular project"* (Weir, 2017).

Another term we come across when talking about procedural audio is the technique of physical modelling. In this type of audio, a physical model of a sound source is accurately recreated. It therefore behaves physically correct.

> *"Physical modelling of sound is a technology that can be understood as a specific type of procedural audio" (Hug, 2011, p. 389).*

> *"To physically model an object means to simulate how it behaves physically and from an audio perspective this means simulating how an object vibrates in response to excitation and causes sound waves to be radiated from it" (Mullan, 2011, p. 343).*

Sinclair (2020) describes one possible implementation in game audio when he talks about the idea that physical modelling could generate the appropriate sound of a rolling barrel in real time, provided that the parameters of the game engine are linked to the model.

This technique could theoretically solve the problem of physical audible interactions in virtual worlds. A physically correct sounding interaction like the one mentioned above would require thousands of pre-recorded audio samples to cover all possible sound variations that the barrel could make. Therefore, it would make sense to simulate the exact sounds through physical modelling.

> *"One recording of a piece of wood being struck may be enough to provide realistic audio in a pre-determined scenario but it will be inadequate in a fully interactive environment.*
> *A partial solution to this problem is to use multiple recordings. We could, for example, record a block of wood being struck on various points with varying forces using different objects. When its virtual counterpart is excited, an algorithm can then determine the most suitable sample to playback or interpolate between the most appropriate samples. However, this approach can quickly become expensive in terms of both the memory and processing power required" (Mullan 2011, p. 341).*

At moments when these audio samples do not match the physically correct audible representation, we often perceive the audio as inappropriate, which can be often the case. Of course, this can also lead to funny and deliberately chosen comedy like situations. In the game Half Life 2 (2004) the player can use a gravity gun to fire objects or to move them around. Since the game uses a physics engine, the interaction with the objects is simulated more or less realistically. However, if you listen to the sound when certain objects such as

cupboards or wooden crates collide with each other, you will notice that the sound somehow does not resemble the physical representation and often seems too loud or inappropriate. Even newer games like Red Dead Redemption 2 (2018) are still struggling with the same problems almost 20 years later. The sounds that take place when the player collides with the boulder wall do not sound accurate and are also somewhat glitchy as the audio material repeats several times. This can be observed in the video of the youtube channel *Nemsk (2019, October 3) c*alled *Best Ragdolls EVER (Euphoria Physics)* at 1:00.

One advantage of physical modelling in a virtual world is, that such a model is highly flexible and can be changed at any time. It can change instantly and is therefore perfect for adapting to different gameplay situations.

> *"Procedural audio, on the other hand, is highly dynamic and flexible; it defers many decisions until run time. Data-driven audio uses prior assignment of polyphony limits or priorities for masking, but dynamic procedural audio can make more flexible choices at run time so long as we satisfy the problem of predicting execution cost" (Farnell, 2010, p. 321).*

> *"Further advantages of procedural audio are versatility, uniqueness, dynamic level of detail, and localised intelligence. The sound of flying bullets or airplane propellers can adapt to velocity in ways that are impossible with current resampling or pitch-shifting techniques" (Farnell, 2010, p. 322).*

Another advantage of procedural audio is that it does not always require the same amount of computing power. Samples, on the other hand, always need the same amount of memory, no matter what audio content they contain. Procedural is able to have a variable cost.

> *"In mixing a sound scene we may fade out distant or irrelevant sounds, usually by distance or fogging effects that work with a simple radius, or by zoning that attenuates sounds behind walls. Until a sampled sound drops below the hearing or masking threshold it consumes the same resources regardless of how much it is attenuated. With dynamic LOAD techniques a synthetic source can gracefully blend in and out of a sound scene producing a variable cost" (Farnell, 2010, p. 347).*

The question that arises now and has been discussed several times is why games don't rely more on procedural audio. One reason is that there are not enough easy accessible tools for implementing procedural audio when it comes to more complex concepts. That means you have to integrate pure data or Max into Unity as described above, this can be starting hurdle. However, there are a few providers who offer procedural gameaudio plugins, such as gamesynth[12]. I think that a recorded sample and the technique of sample layering still have a special quality that cannot yet be replaced with procedural audio plugins. I don't think it's necessarily about replacement, but procedural audio is just another tool for me to access. Especially more synthetic sounds like UI[13] sounds can be integrated very well with proceudral audio in my opinion.The point is also whether one really wants to aim for this high degree of realism in games. There are certainly games or installations that would benefit from a high degree of realism, especially in the virtual reality sector. For my personal process in game audio, the question is what problem can procedural audio solve in my game better than sample playback system.

Before concluding this chapter, I would like to point out that there seems to be a larger debate about the meaning of procedural audio and generative audio. While adaptive audio and interactive audio are relatively clear to separate, there seem to be a number of views on generative audio and procedural audio. During my readings and exploration of these two terms, it was relatively obvious that there is misconception in how these terms are being used. The lines between procedural and generative audio are quite blurry.

> *"The core philosophy of generative music is based on the idea of indeterminacy—the introduction of chance into the unfolding of a composition and the randomization of musical content for the*

[12] http://tsugi-studio.com/web/en/products-gamesynth.html

[13] User Interface

purposes of rendering something that is constantly unique. This interactive music system is sometimes also referred to as "algorithmic composition" or "procedural music," but the meaning is the same" (Phillips, 2014, p. 188).

"Generative music is music that is created via systemic automation, and is sometimes called procedural music, musical metacreation, or algorithmic music. These terms are mostly synonymous and can be used interchangeably, but we will use "generative music" for simplicity" (Plut & Pasquier, 2019, p. 2).

"In this book, we will tend to favor the stricter definition of the term procedural, that is, real time creation of audio assets, as opposed the real time manipulation of existing audio assets. The difference between procedural asset creation and advanced stochastic techniques are sometimes blurry, however. These more advanced random or stochastic techniques are certainly very important, and their usefulness should not be underestimated" (Sinclair, 2020, p. 19).

In a recent seminar talk (iGGi Seminars, 2021), Alessandro Coronas[14] and Paul Weir[15] discuss their understanding of the two terms. It can be seen that both have a completely different understanding of the terms. This also leads to some confusion later on in the seminar, because you simply don't know who is using the term in which context. Another question that emerges during the seminar is whether one distinguishes between music and audio when using these terms.

I think there is a multi-layered problem of conceptualisation in this discussion that moves back and forth between the term generative music, mostly popularised by Brian Eno and procedural generation that is often referred as technique when it comes to creating assets in real time in a game engine. Since Generative Music is kind of also referring to a music genre, it can also be an output of a procedurally generated system, which doesn't make things any easier. Perhaps it would make sense to replace the term with algorithmically generated variability music.

With the advance of artificial intelligence and the possibility of neural networks trained systems, new audio creation tools are emerging. With artificial intelligent voices such as Apple's Siri or Amazon's Alexa there seems to evolve a differentiation between the terminology of generative audio and generative music when it comes to speech synthesis, where a AI model is trained with a dataset of recorded voice samples. Google refers to a wikipedia entry on their Google Arts and Culture[16] webpage that also quotes an article published back in 2017 by the Economist. As this sort of technique is still in its early stages the terminology is not clearly defined yet. Other Google projects such as Nsynth[17] (Neural Audio Synthesis, 2017), who are experimenting with AI networks, are using the term neural audio instead of generative audio.
With these different examples I wanted to show that these terms are not ossified and that it might be necessary to establish new terminologies. Nevertheless, it is beyond the scope of this thesis to elaborate new ideas for terminology, and it is clearly not the aim of this work.

In summary for me generative audio describes a clever system that generates audio which is constantly changing and never repeats itself mainly through variability. Therefore I refer to generative music in this thesis when music or a sample is played with a certain chance in FMOD and when the music is constantly changing. The audio material can consist of samples or it can be generated in real time with synthesis. But also a system that uses, for example, different physical triggers to play a sound body can be called generative audio.

[14] Composer, Sound Designer of the game Mutazione (2019)

[15] Composer, Sound Designer & Audio Director of the game No man'Sky (2016)

[16] https://artsandculture.google.com/entity/generative-audio/g11d_88xn76?hl=en

[17] Neural Audio Synthesis

5. Music Systems

Since the beginning of our collaboration, it was clear that the developers wanted music in their game. So it was up to me to send them suggestions for music and to start composing little sketches. We agreed relatively quickly that we could go in the direction of Muzak[18] inspired content with rhythmic elements. The developers describe the tone of the game as light, funny, wholesome, harmon. The music should definitely support this mood, but it should also have its own character and thus give the game it's own identity. The Muzak genre is an interesting framework to start with, we agreed that the music could also be a bit more quirky, unpredictable and modern compared to the genre of Muzak. In the beginning, I did a little research to become more familiar with the musical aspects of it. I don't want to go into too much detail here because that is not part of the work. Nevertheless, I would like to briefly mention some characteristic of the genre.

> *"People have Muzak as the precursor to the all-encompassing musical soundtrack – to Napster and the iPod – and that's true, but only to a point. Insofar as it was a closed system, a curated system and a discretely distributed system, it was more like radio than a mixtape, and more like iTunes than bit-torrent: static, curated, non-transferable" (Baumgartner, 2012).*

Instruments	**Groove**	**Tone**	**Similar Genres**
organs piano strings woodwinds vibraphone brass guitars vowel based scatting drums percussion	swing bossa nova nambo chachacha	relaxing not too serious	lounge exotica space age pop ambient roots: cuban, latin

Figure 06 | Muzak

From the beginning it was also about how we could expand the game experience with a specially created music system. During the production phase, I invented and tested several ideas from scratch and discussed them with the developers. To test each system's concept I started with recording the music in sync with a gameplay video in Bitwig. The question may come up why not start integrating the assets directly in the game from the beginning. The answer is simple. The process of implementing the system directly into the game is much more work intensive and would not provide me with more information to evaluate the system's outcome in the early stages of its development. With this method I can simulate the system, with the help of a linear gameplay video. Working in such a way could be compared to developing a storyboard sketch in an animation movie. It provides enough information to assess the possibilities of the system for an initial evaluation. If a system surpassed the Bitwig video test successfully, it was tested in FMOD with all the parameter settings, that I initially have planned for the system A term I would like to explain briefly is called real time parameter or in short RTP. In computing, real time parameter refers to a parameter whose value is

[18] **Muzak** is a type of background music that is often played in retail stores and other public places. It is typically characterized as being bland and unobtrusive. Muzak was invented in the early 20th century.

measured as a function of time. Real time parameters are often used in feedback control systems. In game audio, an RTP environment allows us to change or automate different aspects of recorded audio in realtime. For example, an RTP can control the volume of a sound effect or it can be mapped to the volume of an audio track . When it comes to implementing audio, that should change in realtime to gameplay, an RTP setup is essential. Implementing such real time parameters in unity needs some scripting skills, therefore a good communication with the programmer is necessary, as he has to understand my needs and ideas and vice versa. Aaron from Lululu Entertainment is setting the more complex ones up in our prototype.

The following sections covers the exploration and creation of the different adaptive music systems. Each system is presented and the questions of the thesis will be discussed. The music composed for the systems is sometimes very rough and not fully elaborated, this is deliberately chosen because otherwise it would take too much time for a testing scenario.

System 1 [ambient loops]

The first system I came up with consists of a total of nine different loops, divided into three loops per room. While the player is progressing through the rooms, each loop's play-start is attached to a specific game event, that the player encounters during the playthrough. In the following section I explain at what point each loop starts and what kind of musical layer gets added. As long as the player not progresses, no loops are getting added or subtracted in this example. It is important to note, that this system was developed with a constant flow of ambient music in mind. During the design process I have already steered towards an ambient soundscape for the game, because I assumed that a pattern based rhythmic structure would not be beneficial for such a system. For the musical content of the system, I worked in Bitwig mainly with random modulators to achieve variability in each loop. Also the setting of different loop lengths contributes to the fact that the music changes constantly during play. This is a well-known technique for composing generative music.

Room	Event	Loop	Instrument
Bedroom	In the bedroom scene the first loop gets triggered with the initial game start.	1	Ambient distant bells
Bedroom	The player now has to solve the alarm clock riddle. If he manages to do that loop two starts	2	organ pad
Bedroom	The third loop in this scene gets added when the player reaches the door and opens it	3	"Na na na" vocals
Bathroom	gets played when Henry is possessing the sponge (synth pad).	4	Synth pad
Bathroom	gets added when Henry is jumping into the bathtub	5	String pad
Bathroom	gets added when Henry is drying himself with the towel	6	Clave
Kitchen	gets played when Henry is opens the door to the kitchen	7	Hihat
Kitchen	gets played when Henry possess an apple	8	Kick
Kitchen	Henry reads the papers	9	Hihat 2

Figure 07 | System 1 instrument chart

Room	Event	Loop	Instrument
Bedroom	the first loop gets triggered with the initial game start		organ pad
Bathroom	gets played when Henry is possessing the sponge		distant bells
Kitchen	gets played when Henry is possessing the apple		string pad

Figure 10 | Instrument chart three loops

In FMOD, each loop is part of a music event in FMOD. One could compare this event as a song in DAW clip launcher environment. The whole event gets initiated, when the game starts. If the player now reaches a certain progress in the level, the realtime parameter value of Loop 1 (at the top of the user interface) will be set from 0 (false) to 1 (true), which then triggers the loop 1 to start. The same can be applied for the rest of the loops. In unity, a script would have to be written that checks when these events take place in order to set the parameter to one in each case.

Figure 08 | System 1 nine loops parameter (top)

During the development of the first system I started to sketch a simpler idea of a loop based system that includes less loops, which leads us to an alternate version that I developed. I wanted to test the possibility of breaking down and simplifying the system a little bit more. The following example is only based on three different loops, that are all getting triggered in one of the rooms at a specific event. Each loop has a different length.

Figure 09 | System 1 three loops parameter (top)

Each loop is embedded in the same FMOD event. The real time parameters are identical to the first system, which means that each loop gets triggered individually, at the event mentioned in the chart above.

How technically complex is the system?
The implementation in unity would be pretty straight forward and it would not take an enormous amount of time to implement triggers in each room, when a certain event arrives. Implementing the code with help of a unity script for this kind of system should be easy to realise. It only would need to start the FMOD event in the beginning of the level and then change the parameters by code at a specific event. The technical complexity of this system I consider as pretty low. I think this as a big advantage when it comes to prototyping. But if development continues, the claiming of the key events that should trigger the audio clip, could be a problem, which I will discuss later in this chapter.

What kind of gameplay actions are linked to the system?
With its linearity approach this system is closely linked to the progress state of the game. Only when the player progresses in the game, the loops getting added and the music will change. In further production, it would then have to be decided exactly which events should trigger the loops during the player's progression. In our example, these are on the one hand invisible spatial collision triggers, that are placed at certain points in the game world (when Henry collides with them, the loop gets triggered). On the other hand, they are events that take place, when Henry possesses a specific object. The last one is directly linked to the main mechanic of the game, the ability to possess and control any object in the game world. In summary, the system provides a functional core that is interwoven with Henry's game mechanics and specifically with the player's progress.

Can the system adjust to different levels?
In each new room, the key events, who should trigger the next loop, must be established first. The problem is that when defining these key events, their timing must be taken into account. Otherwise, loops could be triggered one after the other and there is a risk that no loop is triggered for a longer period of time. It must also be taken into account whether Henry triggers these events in the game at all. Since the level is relatively open and there are several ways to reach another room, this is a difficulty. Also to mention here is, that not every object has to be possessed for completing the level. All the trigger events would have to be planned precisely and this would have to be correlated with the player's freedom of choice. There is also the danger that these key events would have to be constantly adjusted if a small change was made in the level during development. The question that also arises here is, what if the next level has for example eight rooms? Another crucial question that we can ask ourselves now is: What kind of system can we think of, that solves the problem if the player does not advance in the exact way the programmers originally planned? For these reasons, I don't consider this system to be very adjustable to new levels.

What are the benefits of the system?
A benefit of this system is that it can play audio at any point in time given by an event. Because the player triggers these events at random intervals and the loops have different lengths, the system is able to play generative music without any pulse. The system is also relatively easy to realise for the prototype. The main feature of it is that it can adapt the music to the progress of the player.

What can be improved and how?
While testing both systems, new answers and ideas crossed my mind. Because only one loop is being triggered in each room, the player will rarely notice a change in music related to his actions. I am not claiming that the player should consciously perceive a change in the music, but I am sure that this system does not improve the experience in a drastic way. I could just as easily record a 10min ambience track and have it play in the background. I think the difference would be barely noticeable. It's obvious that the system with only one loop in each room offers less interactivity on the musical level, than the system that activates three loops in each room. Especially for the three loop version, I would say that this system does not enhance the overall gameplay experience, because the players will not recognise themselves, that their actions have an impact in the music.

I noticed that the system desperately needs the option to de-stack loops. If I add more and more loops, while the player is progressing though the levels, each level could end up in a very dense and maybe chaotic

soundscape, which can be very distracting. At the moment the system is completely linear, it only can add content. That being said, there should be the option to mute or replace the loops during the gameplay at a specific time. But here the question comes up, what triggers the command to de-stack a loop? Without a specific implementation, the idea is pointless. I will later refer to this section of the thesis, because it will lead us to an interesting approach of another system. But for now let's move on. The other system's weakness is, that it doesn't allow a time synced loop concept. Obviously the player will not trigger these loops in a quantised way. If I decide to use loops that should be in sync with each other, for examples rhythmic patterns, the setup in FMOD has to be designed in a different way. For that reason we address, that this system can't support pulse oriented music. It works well with ambient music, because it has no pulse in general, and the tracks don't have to be in sync with each other.

Feedback from the developers

When I showed this first concept to the developers, I received following feedback from them. They liked the idea of the system, but the music was not intense enough. They could imagine the music very well in another level, but for this level they wanted music, that is a bit more rhythmic.

I understood well, what the developers meant. Although I liked the mixture of the rather calm ambient sound and the gameplay, I can understand that they want something slightly more upbeat. From this we decided, that I would design something pattern based. With the feedback from the developers and my own experience in creating System One, I have continued to develop the system. What has changed is discussed in the next section.

System 2 [stacked loops]

As I mentioned in the previous chapter, the current system is not able to play music in sync. So the next step is to find a solution for that issue. There are multiple ways to achieve this goal in FMOD. One way to would be to set the parameter of a Loop to false to deactivate it. When we set its parameter to false in a unity script, the loop will be played to its end, which dependant on the loop length, could take a long time to stop. For stoping the loop instantly, we need to map the parameter value to the volume automation of the loop. Through this way we can easily fade out the loop when the parameter is set to false, without adding code. With the knowledge of automating the volume parameter, a modification of the system can be realised for its benefits. If we start each loop at beginning of the bedroom scene, all loops should be in sync with each other. For this purpose we need to move all audio clips in FMOD from the parameter page to the timeline page, because all clips should be synchronised. All tracks are now placed under each other, as well as in a DAW. Now we can simply automate each loop's volume with our real time parameter setup. This gives us instant control over playing a loop and disabling a loop. In reality though, we only change the volume of the loop. Because we now have the possibility to play audio content in sync with each other, I can better implement a pattern based composition. So let's replace all of system one's audio clips with a pattern based four bar loop and see what happens. For this I recorded a pattern based four bar loop composition with nine tracks. The same game events as in the previous system will start each loop's playback in following order.

Room	Event	Loop	Instrument
Bedroom	In the bedroom scene the first loop gets triggered with the initial game start.	1	Bass
Bedroom	The player now has to solve the alarm clock riddle. If he manages to do that loop two starts	2	Kick
Bedroom	The third loop in this scene gets added when the player reaches the door and opens it	3	Clave
Bathroom	gets played when Henry is possessing the sponge (synth pad).	4	Hihat
Bathroom	gets added when Henry is jumping into the bathtub	5	Trumpets
Bathroom	gets added when Henry is drying himself with the towel	6	Tom
Kitchen	gets played when Henry is opens the door to the kitchen	7	Arp
Kitchen	gets played when Henry possess an apple	8	Snare
Kitchen	Henry reads the papers	9	Piano

Figure 11 | System two instrument chart

How technically complex is the system?
I consider its design as not complex and it could be implemented relatively quickly when it comes to a prototype. In FMOD, the individual automations for the volume of each track have to be set carefully, which can be a bit time-consuming with a large number of tracks. But as in the first system, there is a danger that

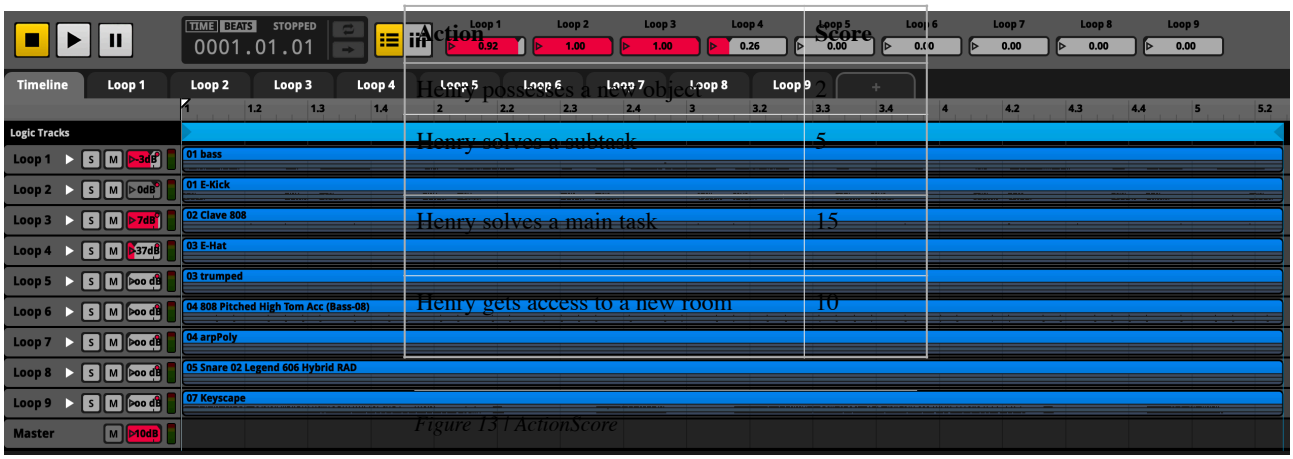


Figure 12 | System with nine loop parameter (top).

the system will become much more time-consuming when it comes to determining new trigger events in future levels.

What kind of gameplay actions are linked to the system?
Just as with the first system, the new one serves its purpose if the player solves the key events fluidly. It is linked to the possess mechanic of the game and the progress that the players will make during their playthough. Therefore nothing has changed compared to system one.

Can the system adjust to different levels?
This system shares the same problem as the first system. When it comes to determining which events will affect the parameters, the events would have to be set up from scratch for each level. The only thing where this system can adjust a bit more than the previous one, is in a linear level environment where a certain tension would be the essence of the experience.

What are the benefits of the system?
To my surprise it works way better than I first thought. The music Is repetitive but in a way not too repetitive that I instantly would like to turn it off. This due to multiple reasons. One is certainly that there is a dialog on top of the music that can transform the music into a kind of musical poem. Another reason is that the game events still trigger new loops, which builds tension while the player progresses. A third reason is connected to the arrangement of the music itself. Its a very simple arrangement with a lot of room for other sounds. The sounds of the player's action are going well together with the music for that reason. There is enough space in the mix and the arrangement, for these kind of sounds. At some points in the gameplay even new rhythmic structures emerged from the player actions. In these moments, the music and the sound effects really merge.

Due to its properties of playing music in sync with each other, it can be used for music with or without a pulse. I have also observed that the linearity of this system creates some tension all the way to the kitchen. This tension generating mechanism could be useful for a certain level or moment in the game. In my opinion, it could provide an enhancement to the game experience. Unlike the first system, it offers the possibility to reduce the number of loops that are being played immediately. Since the real-time parameters are assigned to the volume controls of the individual loops, we can switch the loops on and off immediately, and the loops keep in sync with each other. But we have not yet decided what kind of game events controls this action. We can conclude that we have not yet solved this problem. This system is less linear as the first one when we consider its design. It can adapt to the players progress and it could decrease the musical content if we would set up specific events for it.

What can be improved and how?
Let's imagine that the player lingers longer in a room without triggering a key event. In such a scenario, the system would fail because the next loop would not start without triggering a specific event related to the

player's progress. The music would remain in an endlessly repeating loop. This is especially annoying when it involves pattern based music that does not change. This problem would be a real bummer, even though the player is enjoying exploring the room and he might rearrange the whole scene to create something funny.

Feedback from the developers

The new music I composed for the system went more in the direction that the developers envisioned. So we agreed that I would continue to work in that style. During the conversations with the developers, I went through different features of the gameplay. Thereby we discussed which game mechanics the music system could be linked to. The question is, what would be a way to link the system more directly to the player's actions in order to influence the music more directly? We could track the virtual position of Henry and thus read out a value that could change the music. We could determine in which room henry is and this indicator could change the music. We could measure the input rate of the player and use this value for changing the music. Eventually, we agreed that the music should somehow fade out when nothing is happening in the game. As I mentioned in previous sections, we concluded that the system should be able to respond to Henry when he is not progressing.

System 3 [action score]

Aaron came up with the idea, that we could measure Henry's activity with an action score. Over time, the score should decrease if Henry doesn't do anything. I think it’s a brilliant idea. If we take a closer look at the gameplay, we can measure the following tasks and activities of Henry and assign different scores to each action. As these are variables, we can change them in unity. For the implementation in FMOD we need a

parameter, that adds up these points. In the unity script we need also the option that the score is decreasing over time (Action Score Decay Speed) and the possibility to set the action score range (Action Score Range). In my first approach to using this system, the Action Score parameter affects several aspects in FMOD. It changes various audio effects and the volume of the individual tracks that are displayed in this list.

How technically complex is the system?
If you look at the FMOD implementation, the system is relatively simple. There is only the action score parameter that you have to set up. Since all of the chosen actions (possessing an object, main tasks, sub tasks, room reveal) are already part of the core game mechanics, it wasn’t that difficult for the programmer to unify all these existing mechanics into a script, which was then able to communicate between the engine and FMOD. Composing on the other hand for such a system is a more complex challenge. Because it is not a linear system, the music can increase and decrease at any time. A low action score should decrease the music, while a high action score should bring more hectic atmosphere to the music. The difficulty lies in combining these two worlds in a seamless transition without losing tension while navigating from to another extreme. All in all, the system is on the more complex side when we consider its overall design. Unity, FMOD and the Music are closely and in a more complex way intertwined than in the systems before.

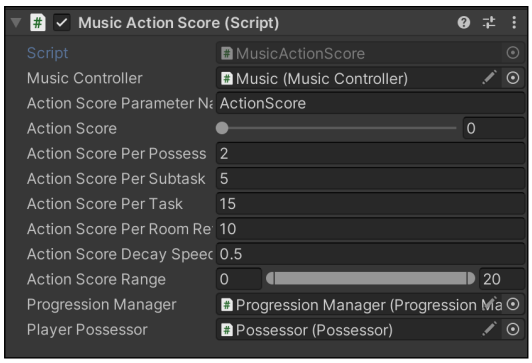


Figure 14 | Variable set up in Unity

What kind of gameplay actions are linked to the system?
The system is directly linked to the action when Henry becomes a new object. This mechanic occurs in every level and will be a constant throughout the whole game. By differentiating between main and sub tasks the system is connected with the task mechanics of the game. Also the event when a new room gets revealed is implemented in the system. Therefore we address that the system tracks the progress of the players and their activities given by the core mechanics of possessing objects.

Can the system adjust to different levels?
Because the system is closely linked to the main mechanics of the game, it can be used independently of the level architecture. Even if, the second level will look completely different and there may be more than three rooms, the game mechanics and thus the reaction of the system will remain the same. Since the system also offers the possibility to adjust certain variables in the action score script, it can even be fine tuned for a new level.

Loop	affects	affects	affects	affects	affects
Bass	Volume	Equalizer			
Kick	Volume	Equalizer			
Clave	Volume	Pitch Shifter			
Hi-Hat	Volume	Chorus			
Trumpet	Volume	Equalizer	Reverb		
Synth-Arp	Volume	Equalizer	Chorus	Delay	Reverb
Snare	Volume	Equalizer			
Keys	Volume	Equalizer	Modulating EQ	Reverb	

Figure 15 | System 3 FMOD set up

What are the benefits of the system?

The ability of tracking the players progress in the one hand and measure how often the players becomes a new objects on the other hand, solves an important problem, where the other systems didn't provide a

FIGURE 17 | ACTION SCORE SYSTEM

solution for. Player activity and progress are both rewarded. As mentioned above, the foundation of the system is the already existing game mechanics. For this reason, despite its complexity, we were able to create it relatively quickly (it took Aaron only an hour to write the action script). With the help of the variables in the script, the system can also be changed and adapted, which can be a big advantage in my opinion. While playing the game and testing the system with the given action score values it felt rewarding that music was

able to increase and decrease. Especially the room reveal action score felt great to me. Because of this experience I consider that the system is able to enhance the game play experience through its adaptability.

What can be improved and how?
During testing, I noticed that besides the action score settings in unity, there is another important factor that influences the build up and decay of the music. If the player completes a task that earns a lot of points, the action score jumps from, let's say for example ten to fifteen points within a short amount of time. This has the consequence that the parameter also jumps to this value and the music therefore changes rapidly. To solve this problem, there is a so-called seek speed option in FMOD, which can be found on the parameter sheet. This allows us to set the speed of the playhead on the parameter sheet. This example proofs that a balance must be found between the automation tracks in FMOD and the action score values for making the transition from one to another extrem more seamless.

Figure 16 | seek speed option in FMOD

Another problem that I have encountered was that constant increasing and decreasing of the music resulted in a tiring experience after a while. The contrast of literally no music and a lot music felt not right. A solution for this problem would be, that music would not decrease but It could change its style to a more laid back arrangement, when the action score is low. A threshold should be set, at what time the music decreases completely when Henry is doing nothing. It is important to mention here, that the sub tasks and main tasks must be defined first. In the current production phase, this is still a bit vague. I also noticed that the music needs more variability. This can be achieved with an additional parameter that indicates in which room the player is, this leads us to the next system.

Object	**Music Event**
Alarm clock	percussive rumble
Pillow 1	piano chord 1
Pillow 2	piano chord 2
Pillow 3	piano chord 3
Plant	piano pattern
Lamp	chip sounding synthesiser sequence

FIGURE 18 | TASK CHART

System 4 [room indicator]

In this system, everything is exactly the same as in the previous one, except that there are now three additional parameters that indicate in which room Henry is in.

How technically complex is the system?
With a few additional lines of code, the script can now detect which room Henry is in. In FMOD a parameter must be created for each room so that the script can communicate with it. Because of this added function, I rate the complexity a little bit higher than previous system.

What kind of gameplay actions are linked to the system?
In addition to the previous system, this one is connected to the different rooms of the levels.
Solving a puzzle and moving on to another room is a core mechanic of the game.

Can the system adjust to different levels?
It should be noted that levels with more than three rooms are a potential drawback for this system. One would then have to decide whether to introduce a fourth parameter that captures the fourth room, or to trigger the music of the first room again, when the fourth room is being entered. Otherwise, the system can be adjusted well.

What are the benefits of the system?
With the ability to indicate which room Henry is in, the system can provide this information to FMOD. There is now an additional layer in the system that is interwoven with the core mechanics of the game. This gives the possibility to change the music when entering a new room. In this way, different musical themes can be linked to the particular rooms. It also allows the possibility to change the music only slightly, so that there is a certain variability in it. All this can be realised with the so-called horizontal composition approach, which was described in chapter four. With the help of the room indicator, you could theoretically create different reverb presets for each room.

What can be improved and how?
With the new possibilities that this system offers, there are new options to adapt the music. For this reason, the first thing that needs to be done is to expand and enrich the musical content. Melodies and themes could be changed or played by other instruments in the different rooms. Even musical transitions can be now realised in FMOD when switching between rooms. In order to increase the variability of the music more, certain audio clips could be given a chance factor in FMOD. That being said, the system can be further improved, especially in terms of music and composition. Otherwise, the action score variables in Unity and the automation lines in FMOD should be adjusted when testing the system.

The system allows you to create a composition for each of the three rooms. As described above, this becomes a problem if there are more than three rooms, because then a new parameter would have to be introduced and the script would also have to be updated. Even though this would not be too much work, it is important to mention this here. One way to prevent this would be to measure when a new room is entered in general, regardless of which room it is. This in turn prevents the possibility of writing music specifically for a room. Another idea would be to create three virtual invisible zones in unity. One zone could therefore also contain two or more rooms.

System 5 [possessor trigger]

The initial idea of this new system is that world objects, that Henry can possess could be categorised and each category could be connected to a different parameter in FMOD. This parameter switch would then influence the audio effects or the content of the music itself. The player would be the unconscious director of the music. His actions would thus generate the music. So all the interactions with the objects would serve as the foundation of a music machine.

How technically complex is the system?
The main complexity of this system is to first categorise all the objects in the game world. A script would then have to determine which type of object Henry has just possessed. Within FMOD, a parameter would have to be made for each of the different categories. The question to be answered is what the criterias of each category would be. Is it material, size or colour? So there seem to be many possibilities that would have to be resolved first.

What kind of gameplay actions are linked to the system?
The main mechanic of possessing a new object is the only action that is linked to this system.

Can the system adjust to different levels?
The mechanics of the system could easily be applied to other levels, but we encounter the same problem as with systems 1 and 2. Instead of having to determine which events change the parameters, we would have to generate a new object class in a new level and categorise objects from the beginning each time, which makes the system more complicated in the long term.

What are the benefits of the system?
The main advantage of the system is that it continuously measures when a new object is possessed. The value does not decrease as it does with the action score. It simply gives us a trigger when a new object is possessed, which can be very useful. I'll go into more detail about this feature in the final system 7.

What can be improved and how?
In a first attempt I effects on and off, depending on the object you play. While editing the video I noticed that I already have established certain rules for specific objects. For example, when Henry becomes a pillow or a blanket, I activated a lowpass filter to give the feeling of something soft. The fact that i was looking for a specific rule in the beginning reinforces my feeling that this system has the tendency to become slightly chaotic. I think it runs the risk of crossing a threshold where the change is so drastic that, the player ascribes too much importance to it and would consider it feedback.

The more I looked into the idea, the more i realised that the whole object categorisation is not necessary. It would actually be enough to simply measure when Henry becomes a new object and then link these actions in FMOD. In one experiment I made a simple music with an arp and only changed the filter LFO speed when henry becomes a new object. This worked very well, although it is only a subtle change in the music.

It is also clear that this system can only use the FMOD effects. This means that you can mainly automate with eq, delay, reverb, filter, flanger, distortion, chorus tremolo and not real time a synthesizer lfo.

Another improvement that could be made would be the following.
You could create 5 object classes (A,B,C,D,E). Detached from their properties, you could then divide all objects equally into these classes per room. This would avoid the problem of categorising.

System 6 [tasks]

This system differs substantially from the others in terms of its musical concept. It would be considered part of a concept where music is only partially played. For introducing this system we will focus only on the bedroom scene. As this system treats some sounds slightly differently, a precise explanation is needed. Each room has specific tasks that the player has to solve for progressing. In the bedroom there are two of these key quests. The first one is that the player has to stop the alarm, and the second one is that the bed has to be done. To reinforce the feeling of successfully solving the puzzle of "making the bed", a chord sequence is played that contains three different chords that are to be played one after the other.

How technically complex is the system?
The main and sub tasks in the prototype are predefined. For each room, these tasks would have to be linked with FMOD and some connected parameters. When an object is possessed, either effects can be controlled or a musical layer can be played. One does not exclude the other. The chord sequences must be played in a certain order, regardless of which object becomes the Henry. In order to be able to realise this, you have to make some things ready in FMOD and in a unity script.

What kind of gameplay actions are linked to the system?
This system is primarily linked to the various tasks in the rooms. Therefore, it can be said that the system is oriented towards the success and progress of the player.

Can the system adjust to different levels?
With the current idea, it would be difficult to transfer the system to a new level. For that, you would have to determine exactly which tasks are solved how in the new level and you would also have to determine exactly which musical ideas are triggered. Since the gameplay around solving the task can certainly become more complex in future levels, it is difficult to create a general formula for such a system at the moment.

What are the benefits of the system?
The essence of this system is that it focuses on solving the different tasks and puts them in the spotlight musically. But the system also allows individual objects to be musically presented when Henry possesses them. I think this could improve the overall playing experience because it seems very rewarding. Since music would only be played sometimes in such a system, it also leaves enough room for ambience and general interaction sounds during the gameplay.

What can be improved and how?
Because I decided relatively early on that I could not continue to implement this system due to time and resource constraints, there is not much to write about at the moment in terms of what improvements could be made. The decision not to continue with the system and the related experience will be discussed in the conclusion.

System 7 [final system]

In this section I will explain which final system the prototype uses in the game. Since this system is a mixture of the above mentioned systems, I will not go into detail on every point. Certain aspects will also be dealt with in the conclusion chapter.

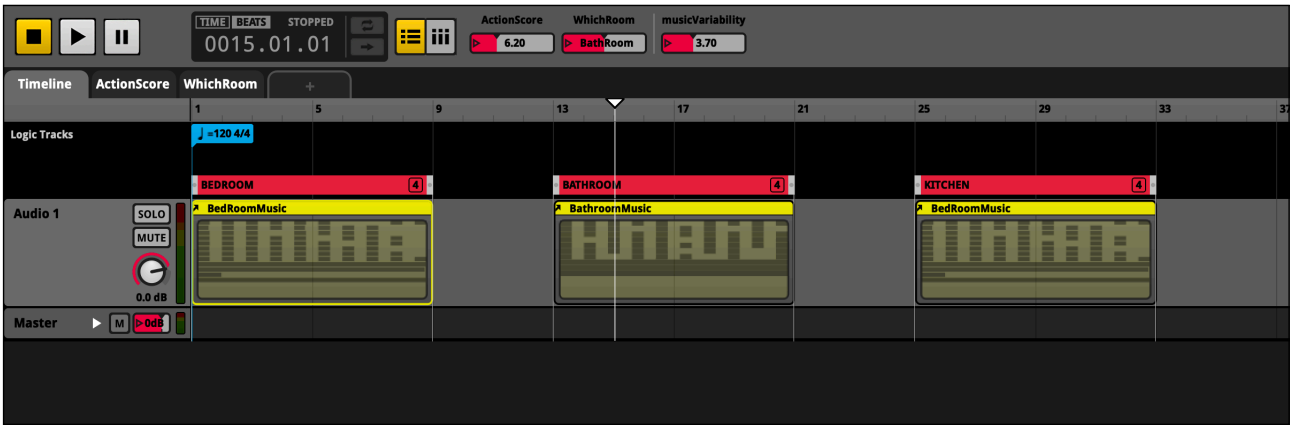


Figure 19 | main music Event in FMOD. Multiple nested events included.

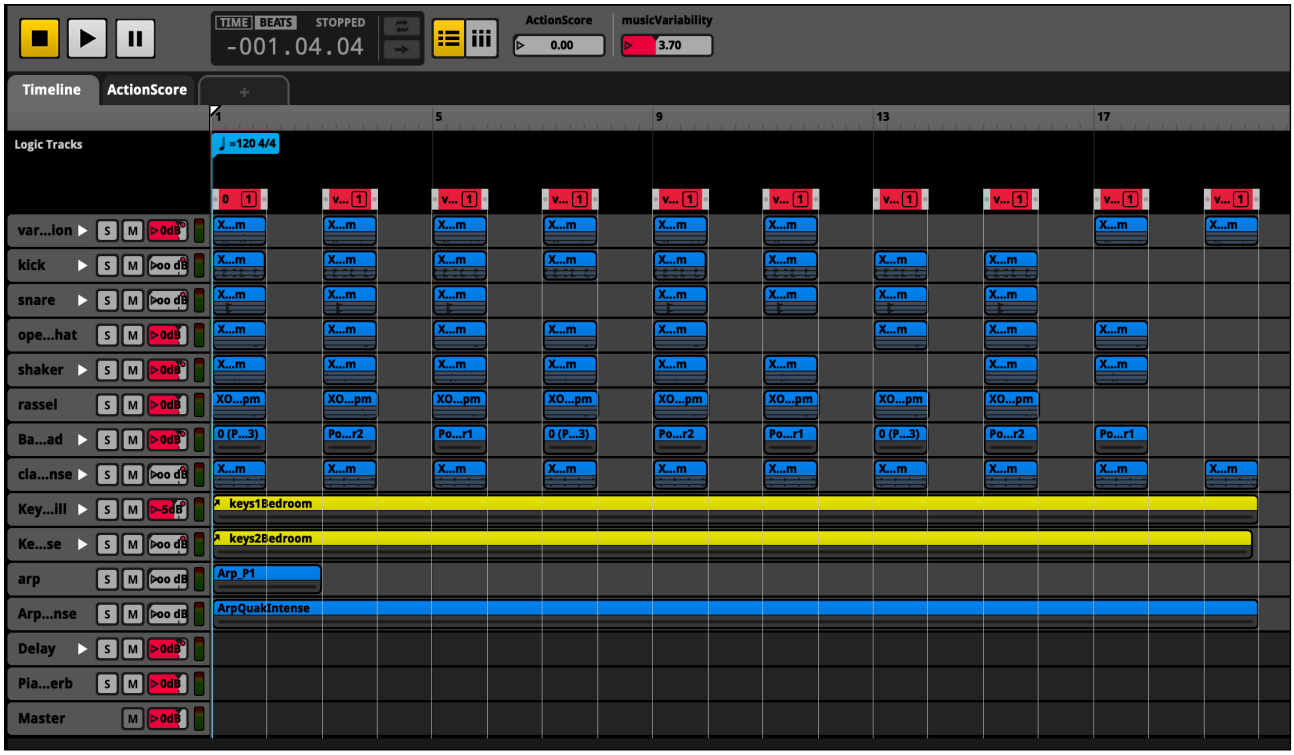
How technically complex is the system?
The system currently integrated into the game features the action score technique, the room indicator technique and the possessor trigger technique. The latter technique is mainly used to create variability in the music. For this I have written a script that generates a random number between 0-10 every time Henry possesses a new object. I have linked this number to an FMOD parameter. If Henry now possesses a new object, the playhead can jump back and forth between different sections in the corresponding events in FMOD. At the moment this mainly affects the rhythmic patterns of the drum kit (figure 20).

Can the system adjust to different levels?
The system can be easily adapted to new levels. However, there is still the problem I described in system four. But the solutions already discussed could also be applied to this system.

What are the benefits of the system?
The main advantage of this system over the others is that it combines different techniques and can therefore be used across the board.

What kind of gameplay actions are linked to the system?
Possess mechanic, room progressing, overall activity of the player

What can be improved and how?
The groundwork for the system has been completed, however, the composition and the variables in unity need to be improved and adjusted. Since this always depends somewhat on the composition and level, you have to test this several times. It would also make sense to sort all instrument groups into separate events and then integrate them as nested events. On the one hand, this creates a better structure and on the other hand, one remains more flexible in FMOD. Especially with larger sections and automated tracks it can get very confusing. I think a next step would also be to compose different transitions when a room change takes effect.



Figur 20 | Bedroom Music Event with the possessor trigger sections (red bars on top of track 1) controlled by the musicVariabaility-Parameter (top).

6. Ambience

During the development of the action score system I came up with another idea of how the action score can influence the audio. In this chapter I describe how I designed the ambience and how it is connected to the music system.

One question I asked myself at the beginning was how much ambience[19] the game actually needs, especially if music is to run constantly, a dense ambience could get too tight in the spectrum of the music and vice versa. It would almost certainly create a slightly chaotic soundscape and would also be a big mixing challenge. I discussed this with the developers from the beginning. We agreed that we would focus more on the music first, because it was more in their interests to have the music first for the overall feel of the game. That made sense to me, although I think that ambience can also play a big role in evoking a certain feeling for a game in general. While I was experimenting with the music system, I realised that we should somehow

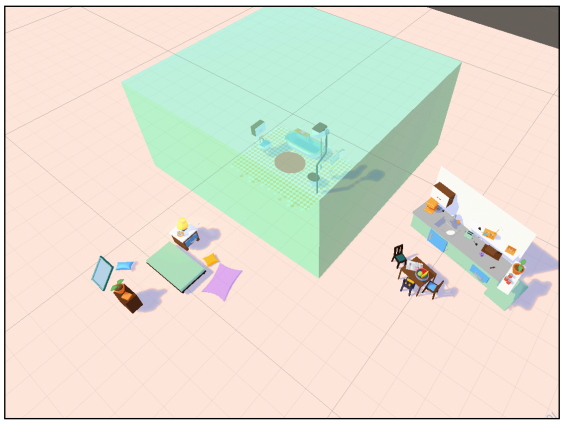


Figure 21 | collision trigger in Unity

link the ambience system with the music system. Or perhaps more precisely, we should link it directly to the action score. In the last section of this chapter I will explain how the ambience system is intertwined with the music system. But first it's important to understand how the ambience system works.

For the ambience in each room, I created an FMOD event. There are three main ambience events in FMOD, that are being triggered, when the player enters the associated room. In Unity I have created trigger boxes in each room for this procedure. Since the rooms are open and the player does not necessarily have to enter the room through the door, a trigger box in a door makes little sense in our scenario. The trigger box starts the FMOD event when Henry collides with it. When Henry leaves the trigger box, the event is stopped. As we don't want an abrupt start and stop situation I have modulated the volume of each ambience event with an ADRS[20] modulator. The ambience fades slowly in when the event gets started and it fades slowly out when the event is stopped. With this technique a seamless transition from one ambience to another can be realised. After testing my system I realised that die ADRS modulator needs some fine tuning, because some of the ambiences distinguished a bit too fast in my opinion. The player should definitely not feel an empty void when crossing from one room to another. So therefore the transition should be really precise.

[19] **Ambience** in gaming refers to the background sound of a particular space. It can be used to set the mood or atmosphere of a scene and create a sense of immersion for the player.

[20] **ADSR** stands for Attack, Decay, Sustain, and Release. It is a common envelope generator used to shape the amplitude envelope of a sound.

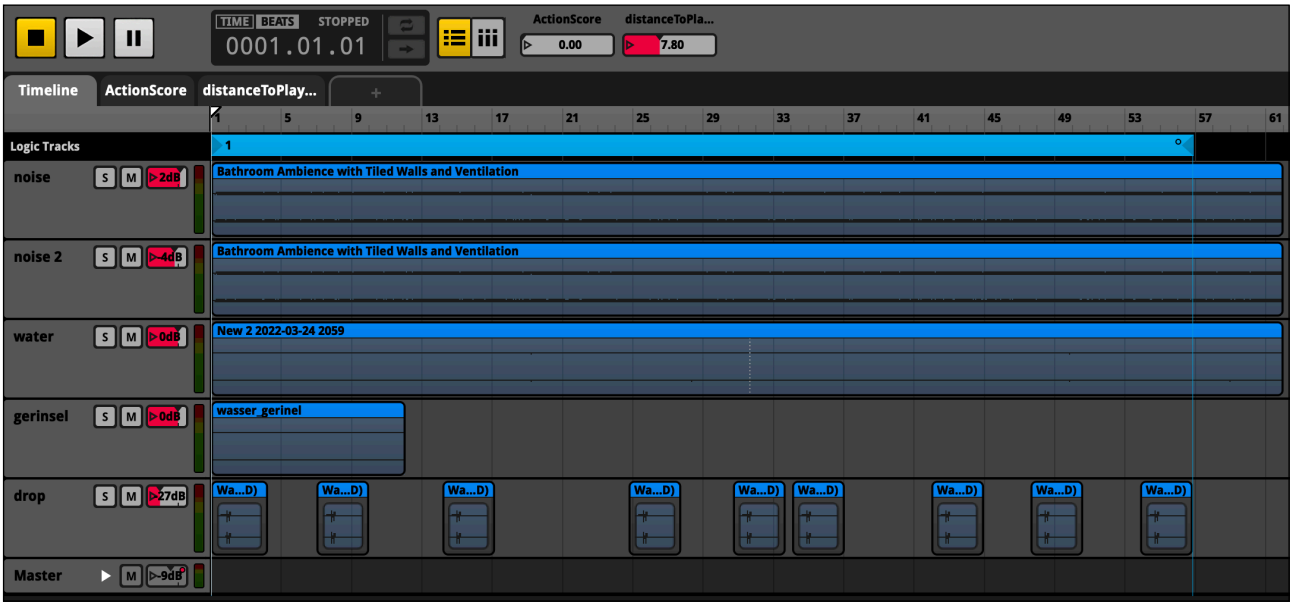


Figure 22 | bathroom ambience event with distanceToPlayer parameter (top).

However, after the first implementation, I noticed several problems with the system. I will briefly describe them in the following. The first problem was that the trigger boxes in Unity work with so-called tags. This means that the trigger is only activated when an object collides with the appropriate tag. In our case I tagged the game object Henry as “player”. Unfortunately, the problem is that you can also play as a plant or an alarm clock in the game, so you would have to tag these objects as well. In summary, all objects would have to be tagged, which is a greater effort and therefore makes little sense. The second problem that came up was that the FMOD listener[21] was pinned to the camera. As a result, when you zoomed out or panned the camera, you heard the ambience of a room that was not visible, but was in the back of the camera. This was very confusing, because we actually want to hear what Henry is perceiving, he is the centre. All in all, I realised that the system was not usable at the moment and that other implementation techniques had to be found.

I then had the idea of rebuilding the system with the help of a simple script. In unity, I placed an invisible game object in the middle of the bedroom. After, I used the script to measure the distance from the game object to Henry's current position. The distance value is assigned to my FMOD real-time parameter called distanceToPlayer, which I created earlier in FMOD. Now I was able to automate the volume of the bedroom ambience in FMOD related to the player distance. With the help of the automation curve, I was able to define the audible radius of the ambience. I then applied the same procedure to the other rooms. With a bit of fine tuning the ambiences are blending really good into each other now.

While implementing the ambiences, another important challenge presented itself. The question originates as to how far I could make specific objects in the room audible and locate them in the virtual space as 3D sound sources. This would mean that sounds such as the humming of the refrigerator would change in the stereo image when changing the camera angle or it would become quieter or louder as the player zooms in on the object with the camera. I had to explain somehow to the developers that we don't get an ambience when I place two or three objects in the room as 3D sound sources. If we do so, we would have three sounds in a vacuum. A challenge was to explain this concept of different ambiences to the developers. For the reasons mentioned above I decided to treat the ambiences as a stereo soundscape. I think this is the quickest and safest way to give the game an ambience and it does not exclude the possibility of placing individual objects as 3D sound sources at a later stage on top of it. In the following section I would like to go into more detail about each ambience and explain what the difficulties were.

[21] **FMOD Listener** is a unity component, that you can attach to any game object, which then perceives the audio in the 3D space. You can think of it as virtual ears.

Bedroom

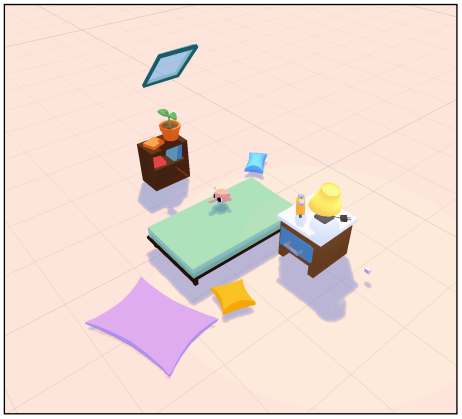


Figure 23 | Bedroom

If we take a closer look at the bedroom, we notice that actually only the alarm clock is an object that can make a constant sound. So here we could create an ambience that includes the ticking of the alarm clock. However, in this case we are facing two issues. The first is that the alarm clock is also an object that can be played. This means that the alarm clock can walk around the house and leave our ambience zone from the bedroom. The ticking would stop if the player leaves the bedroom as an alarm clock if we decide to implement the ticking in the ambience. As a player you instantly connect the ticking sound to the alarm clock that you are playing. It would feel very strange to witness the stopping of the ticking in such a moment. The second challenge is kind of similar but the other way around. Imagine unpossessing the alarm clock in the bedroom and still able to hear the ticking sound, even if you now play as Henry's plant (the one next to the window in the bedroom). Wouldn't that feel awkward too? I tested both possibilities and I can definitely answer this with a yes. But here we can keep in mind, that we might be able to implement the ticking of the alarm clock as a 3D sound source at a later stage.

For the bedroom scene I created a very simple and subtle ambience. It consists only of a low mid range noise and a distant bird chirp and a distant rooster wake up call. I could add some more ambient sounds that would tell us a bit more about the location of the house. For example, leaves rustling from the threes in Henry's garden, or dogs barking or just the occasional car driving by in the distance from Henry's neighbourhood. For this stage of the production the ambience I have created provides enough content for not feeling in an empty space, when no music is being played.

Bathroom

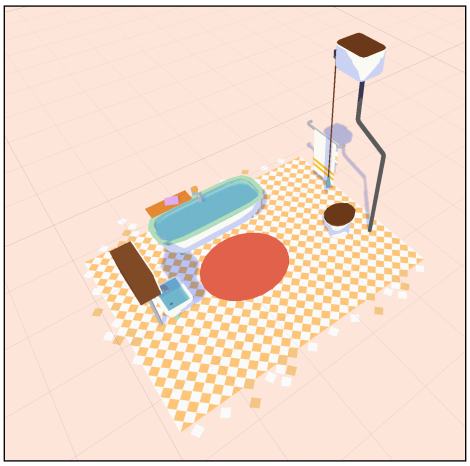


Figure 24 | Bathroom

For the bathroom, the challenge was to design an ambience that doesn't sound like a stalactite cave but doesn't sound like a spa bath either, but still evokes the feeling of a certain moisture and humidity. The ambience includes a noise with a low pass filter. I tried to recreate the sound of water going through a thin pipe with adding some sounds of a small creek in the background of the noise. In addition to that you can hear some water drips. There is a rubber duck that hides above the toilet, which makes a squeaking sound. This sound I would place as a 3D source in the room, as the player can interact with it. This means I will not include the squeaking sound in the room tone.

Kitchen

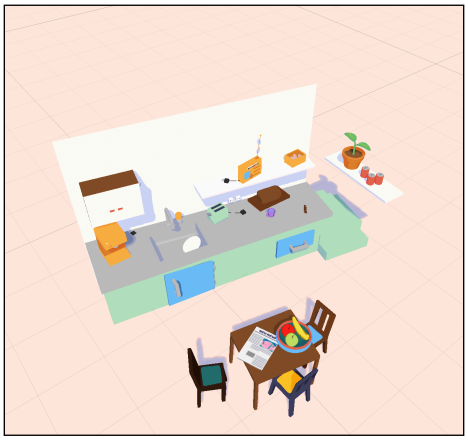


Figure 25 | Kitchen

In the kitchen, I went for a similar ambience as in the bedroom. There is a noise track with a low and high pass filter that cuts off the high frequency and the low frequency content. Too much content in the low frequency range would be too disturbing and it would fit more for a horror space game than in an ordinary kitchen. Here, too, the question arises as to which objects in the space should be implemented as 3D sound sources. Most of the objects in the kitchen must first be connected to the power supply by the player, which means, that they are not making any sound without this action.

These examples show that a crucial factor in the further development will be which sound sources I place as 3D objects and which sounds I implement in the ambience for the specific room. At the moment I can already establish a certain rule. Objects that you can't see in the room work well integrated in the ambience soundscape, because they can help adding additional information about the space and they don’t interfere with the sounds of an actual object that the player can interact with. On the other hand, objects that are obviously located in the world itself and objects that are able to make a certain sound itself can be used as a 3D sound source, but then they should be detached from the ambience. The ambience should be treated as a separate layer of the sounds coming from the object interactions. I think it’s really important to make a strict distinction in this case for the further development of the game.

Action Score and Ambience

As I described above, I had the idea to link the ambience system to the action score. The initial idea behind this thought was to adapt not only the music to the gameplay but also the ambience. The central question that kept popping up was what happens if Henry does nothing or instead decides to engage in smaller actions, that are not so relevant and do not earn many action points. In this case the complexity of the music decreases over time, until the player can barely notice any music. Here we would have the option that the ambience system could play a more important role. As we already have an RTP set up in FMOD that is measuring the action score, it was easy to map the exact same parameter to the master fader[22] of each ambience event. In FMOD I automated then the volume related to the action score parameter. This results in increasing the volume of the ambience when the action score is very low and it decreases the ambience, when the action score is high. Such a set up gives us the advantage that the music and ambience are not constantly overlapping each other. Another possibility that opens up is, that we now can add additional ambient sound sources, that you can only hear at a certain action score level. We don't want to hear the neighbour's dog barking during Henry's speedrun, but we might want to hear the barking sound when Henry is relaxing and sipping a cup of coffee in the kitchen. At this stage of the production I only lower and increase the volume of the ambiences. But it is good to know, that such a setup can be scaled in its complexity.

A major question that will have to be answered in the further development is at what threshold the ambience should become louder. At the moment, this still depends heavily on the music and the distributed action scores, which is why there is no definitive answer to this question yet.

[22] A **master fader** is a fader on a mixing console used to control the overall level of the main mix.

7. Conclusion & Next Steps

After joint consultation with the developers, we came to the conclusion that system number seven can best fulfil the requirements for the prototype. It can be used as a foundation for the upcoming production and meets the quality requirements for the prototype. The main reason for this is that the system is closely linked to the core game mechanics. These mechanics run like a common thread through the entire game. In the game, this consists of the possess mechanics and the moment when a new room is being revealed. It therefore makes sense that the adaptive system is built around this seed and that the actions associated with it are linked to the system in some way.

One of the sub questions of the thesis was, what gameplay mechanics the system could be linked to. Through this question I had to repeatedly examine the gameplay. Over time I realised that we would have to attach something close to the DNA of the game. During the production of the various systems, this realisation began to manifest itself as more and more. Because the action score system is so interwoven with the gameplay, it also opened up the possibility for me to apply it to the ambient system. The ambient system and the music system interact with each other, both can adapt to the gameplay, therefore one could also define this as adaptive mixing. Through this idea of using the action score also for the ambience system, another layer has been created, that makes it possible to tell more about the environment, when the action score is low. The louder the ambience gets, the more attention can be drawn to something in the room. This interaction between ambience and music was not possible with the other systems tested. I believe for this reason this system can expand the game experience a little more than the other systems that only track action points. The music gives the player feedback about his activities and the ambience can communicate something about the world to the player, which can immerse him more in the action. All in all, it can be noted, that the selected system has a higher adaptivity than the other systems. But the final system still needs further improvements. Especially fine tuning with the action scores and balance between ambience and music needs to be adjusted. In my opinion, the music could also be much more subtle and reduced. At the moment I have the feeling that the music changes too often.

It is important to mention here that the other systems also had certain advantages. I can well imagine that in certain gameplay situations I will use one system or the other in future production. There is also the possibility of mixing the systems, which has already happened in one way. The system we have implemented in the current state of development builds on the different systems created during this work. The crucial element in this process was the knowledge I was able to gain from the individual tests of the different systems.

During one point of the project, I realised that I could test an extremely large number of systems. But that would have been not the main idea of my undertaking. Rather, I have become aware that the question arises as to what exactly the system should support in the game. Is it rather the progress in the level that is to be emphasised or perhaps the activity of the player? Should it reward the player's action, even he does not make any important progress in the game, or should the system reward the progress of the player related to the puzzles in the room? Or is there an option for kind of both scenarios? Or should the system not reward any actions at all and only be used during specific moments in the narration? Since this also depends very much on how the developers want their game to end up, it was clear to me, that we would have to discuss this together. In a new project, I would probably ask this question in the beginning, not only to myself, but also to the development team.

Another question I posed for each system was, how complex it was. In this work, it turned out that a more complex system was initially more time consuming to create, but had the advantage of being more adaptable as production progressed. On the other hand, the systems that were initially easy to realise turned into complicated and chaotic systems when they were tried to be used for a new level. Instead of speaking of complexity, one could perhaps rather speak of how effective a system is and define what exactly would contribute to the effectiveness of a system.

During production, I realised that certain concepts were too complex and would take a lot of time to develop and build. Recourses play an important role, and they are usually limited in terms of time and money. Throughout the thesis I recognised that in such a production you always have to manage this balancing act. To give an example, I would like to explore system six [tasks] in more detail, but that would take an enormous amount of time and the resources are not available at the moment.

I realised relatively early on that composing and building the system are closely connected and can affect each other during the process of creation. Each system has required a new approach to composing the music. Composing adaptive music requires a completely different approach than composing linear music. This presented me with a greater challenge at first. I think that the composition and the music still have a lot of headroom. One of the biggest challenges was that the adaptive system adapted but repeated itself in its adaptivity, which was tiring. From this I concluded that the music in an adaptive system has to be variable so that it does not become tiresome. So adaptability alone does not necessarily solve the problem of ear fatigue. Only with the variability factor from system five [trigger] was it possible, to integrate more variability into the system.

One thing that has plagued me throughout the process has been the tedious process of audio bouncing in Bitwig. Bitwig is an enormously powerful DAW for designing sounds in my opinion. But when it comes to exporting, naming and grouping large amounts of audio assets, it can get a bit confusing and complicated. On the other hand, FMOD can be very glitchy and buggy. Audio clips often suddenly have different lengths when imported and loopsections can be attached to the wrong gridpoint without you even noticing. Only when you zoom in a hundred times, you discover that the end of the loop section is not exactly on the grid. Basic functions such as editing multiple audio clips are in vain. All these annoying little things partly reduced my motivation when it came to getting back to work. I think that in the future I will learn the middleware Wwise and also want to have a look at reaper, as Wwise and FMOD offer reaper integration.

Also, the audio effects in FMOD are not up to the standard I am used to in a DAW. I realised that I was often faced with the decision of bouncing a track with the effects included and thus missing the possibility of changing the effect during gameplay, or bouncing the audio asset without the effect and using the FMOD effect instead, which doesn't sound as good and offers fewer options, but can be automated in the game. Actually, I don't find either option entirely satisfying.

Especially with system five [trigger], I noticed that the effects in FMOD are simply not good enough to be able to work with them precisely. The delay effect, for example, lacks filter cut-offs. I would have to create whole effect tracks and route them into the main mix, which is very time-consuming on the one hand, and on the other hand the workflow in FMOD is extremely slow. In addition, they simply don't sound as good as in usual effect plug-ins.

When composing and testing, I have found that smaller changes in the music, such as changing an LFO or minimally changing a pattern, have a better effect than larger changes in the music. So the change should be musically subtle in some way, without the basic structure falling apart. If one moves too far out of this context, these changes can easily be interpreted as feedback. Over time, I have also completely changed the approach in FMOD and started nesting events. The music I composed does not yet correspond to my ideas and is not yet so firmly connected to the muzak genre. I will definitely explore this further in the coming months.

Through the theoretical work in chapter four, I was able to deal more deeply with the terminologies and their meanings. This has given me a more conscious approach and I have also been able to expand my vocabulary. Through this thesis, I was able to gain valuable experience and develop myself in various areas of game audio.

8. References

Bainter, A. (2019, June 5).
Introduction to Generative Music. Medium. https://medium.com/@alexbainter/introduction-to-generative-music-91e00e4dba11

Baumgartner, L. (2012, September 27)
Elevator Going Down: The Story Of Muzak. Red Bull Music Academy. Retrieved February 21, 2022, from https://daily.redbullmusicacademy.com/2012/09/history-of-muzak

Collins, K. (2008)
Game sound : an introduction to the history, theory, and practice of video game music and sound design. The MIT Press

Collins, K. (2017)
From Pac-Man to pop music : interactive audio in games and new media. Routledge.

Carr, D. (2019, July 11). Deconstructing Brian Eno's Music for Airports [Review of Deconstructing Brian Eno's Music for Airports]. Reverb Machine. https://reverbmachine.com/blog/deconstructing-brian-eno-music-for-airports/

Errico M. (2022)
Push. Software Design and the Cultural Politics of Music Production. Oxford University Press.

Farnell, A. (2010)
Designing sound. The MIT Press

iGGi PhD. (2021)
iGGi Seminars - no4 "Procedural Audio in Narrative Game Design" [Review of *iGGi Seminars no4 "Procedural Audio in Narrative Game Design"*].
https://www.youtube.com/watch?v=AllYuKKxks8&t=1s

Grimshaw, M. (2011)
Game sound technology and player interaction: concepts and developments. Hershey, Pa ; New York Information Science Reference Cop.

Jarman C. (2021, May 03)
The Best Video Game Music on Bandcamp: March/April 2021. Retrieved February 02, 2022, from https://daily.bandcamp.com/high-scores/the-best-video-game-music-on-bandcamp-march-april-2021

Hug D. (2011)
New Wine in New Skins. In Grimshaw, M. (Ed.), Game sound technology and player interaction: Concepts and developments (p. 384-415). Hershey, PA: IGI Global.

Droumeva M. & Fraser S. (2011)
An Acoustic Communication Framework for Game Sound: Fidelity, Verisimilitude, Ecology. In Grimshaw, M. (Ed.), Game sound technology and player interaction: Concepts and developments (p. 132-153). Hershey, PA: IGI Global.

Horowitz, S. (2017)
ESSENTIAL GUIDE TO GAME AUDIO : the theory and practice of sound for games. Focal Press Taylor & Francis Group.

Kaee J. (2011)
Theoretical approaches to composing dynamic music for video games. In Collins, K. (ed.). From Pac Man to Pop Music (p. 75-93). Ashgate.

Lopez Duarte, A. E. (2020)
Algorithmic interactive music generation in videogames. SoundEffects - an Interdisciplinary Journal of Sound and Sound Experience, 9(1), 38–59. https://doi.org/10.7146/se.v9i1.118245

Mongeau A. (2017, March 28)
Behind the sounds of No Man's Sky: A Q& A with Paul Weir on procedural Audio. Retrieved April 18, 2020 from https://www.asoundeffect.com/no-mans-sky-sound-procedural-audio/

Mullan, E. (2011)
Physical modelling for soundsynthesis. In Grimshaw, M. (Ed.), Game sound technology and player interaction: Concepts and developments (p. 340-360). Hershey, PA: IGI Global.

nathanidiothend. (2007, October 29)
Brian Eno - Music For Airports Interview [Video]. YouTube. https://www.youtube.com/watch?v=ykJg-vE3k-E

Nemsk. (2019, October 3)
Red Dead Redemption 2 - Best Ragdolls EVER (Euphoria Physics) [Video]. Youtube. https://www.youtube.com/watch?v=-uQ3DvctIlY

Phillips, W. (2017)
A composer's guide to game music. The Mit Press.

Plut, C., & Pasquier, P. (2019)
Generative Music in Video Games: State of the Art, Challenges, and Prospects. Entertainment Computing, 100337. https://doi.org/10.1016/j.entcom.2019.100337

Sinclair J. (2020)
Principles of game audio and sound design sound design and audio implementation for interactive and immersive media. New York, Ny London Routledge.

Stevens, R. (2017).
Game Audio Tutorial: a practical guide to sound and music for interactive games. Routledge.

Sweet, M. (2014)
Writing interactive music for video games : a composer's guide. Addison-Wesley.

Whitemore G. (2014)
Teasing Apart the Terms Interactive Music and Adaptive Music. In Horowitz S. & Looney S. (ed). The Essential Guide to Game Audio (P. 100-120) Focal Press Taylor & Francis Group

Zender D. (2021, May 13)
How procedural audio brings sounds to life in video games. (Last visited 24.04.2022) https://splice.com/blog/procedural-audio-video-games/

Cited works

Bamerang (2021). [Video Game] Lululu.

Beat Saber (v1.4.2). (2019). [Video Game]. Beat Games.

Eine Laune der Natur (2021). [Video Game]. L. Montandon.
https://www.youtube.com/watch?v=BJsc8aGOyKw

Fishery (2022, early access). [Video Game]. Lilou Studios.

Fract osc (2014) [Video Game]. Phosfiend Systems.

In pieces (2021). [Video Game]. Aalto Game Project.
https://rekon.itch.io/in-pieces

Just Cause 4 (2018). [Video Game]. Avlanche Studios. Sqaure Enix.

Half Life 2 (2004) [Video Game] Valve.

Klavierstück XI (1957). Karlheinz Stockhausen.

Marble Madness (1984). [Video Game]. Tengen.

Marble Sadness (2021). [Video Game]. Aalto Game Project.
https://aalto-gamedesign.itch.io/marble-sadness

Music for Airports (1978). Brian Eno.

Musikalisches Würfelspiel (1787). Wolfgang Amadeus Mozart.

Mutazione (2019). [Video Game]. Die gute Fabrik. Akupara game.

No man's sky (2016) [Video Game] Hello Games.

NSynth: Neural Audio Synthesis (2017)
https://magenta.tensorflow.org/nsynth

Red dead Redemption (2010) [Video Game]. Rockstar Games.

REZ Infinite (2017). [Video Game]. Monstars Inc. Resonair.

Skyrim (2011) [Video Game]. Bethesda Game Studios.

Soul Reaver 2 (2001) [Video Game]. Crystal Dynamics.

Super Mario Bros (1983) [Video Game]. Nintendo.

The legend of Zelda: Ocarina of time (1998). [Video Game]. Nintendo.

Tony Hawk's pro Skater 1+ 2 [Video Game]. Activions.

9. Appendix

The attached materials are available at this link and include:

https://drive.google.com/drive/folders/1mkifWS2COq-XaEp1zI6cAhDXRZPLwCm-?usp=sharing

AmbienceScript_distanceToPlayer.cs

system1ambientLoops] nineLoops.mov

system1ambientLoops] nineLoopsGameplay.mp4

system1[ambientLoops] threeLoops.mov

system2[stackedLoops].mov

system2[stackedLoops]gamePlay.mp4

system3 [actionScore].mov

system3[actionScore]gamePlay.mov

system4[roomlndicator].mov

system4[roomlndicator]slowBeatComposition.mov

system5[possessor trigger].mp4

ssystem5[possessor trigger]OnlySynthLFOChanges.mp4

system6[tasks].mov

system6[tasks]GamePlay.mp4

system7[final System].mov

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig ohne fremde Hilfe angefertigt habe. Alle Stellen, die ich wörtlich oder sinngemäss aus öffentlichen oder nicht öffentlichen Schriften übernommen habe, habe ich als solche kenntlich gemacht

Helsinki, 2.05.2022